Fragmentando o DNA de Ferramentas de Alinhamento Progressivo: uma Metaferramenta Eficiente

Mario João Jr.[1,3], **Alexandre C. Sena**[2], **e Vinod E.F. Rebello**[3]

[1] Laboratório Médico de Pesquisas Avançadas, UERJ – Rio de Janeiro – RJ – Brasil

[2]Instituto de Matemática e Estatística, UERJ – Rio de Janeiro – RJ – Brasil

[3]Instituto de Computação, UFF – Niterói – RJ – Brasil

junior@lampada.uerj.br, asena@ime.uerj.br, vinod@ic.uff.br

Resumo. *O Alinhamento Múltiplo de Sequências genéticas é essencial para a área de bioinformática. Devido à sua complexidade exponencial, heurísticas são utilizadas. A mais popular é o Alinhamento Progressivo, com inúmeras ferramentas desenvolvidas ao longo dos anos. Entretanto, nenhuma consegue gerar sempre o melhor alinhamento, nem se sobressair. Assim, os cientistas são obrigados a escolher e utilizar mais de uma ferramenta. Ao invés de desenvolver uma nova heurística, este trabalho apresenta uma metaferramenta que avalia novas combinações de técnicas extraídas de outras ferramentas e coordena suas execuções eficientemente. A abordagem é capaz de alcançar speedups superlineares, mantendo, e por vezes melhorando, a qualidade dos alinhamentos.*

1. Introdução

Ao processo de alinhamento de mais de duas sequências genéticas (DNA, RNA ou proteínas) para identificar as similaridades que refletem aspectos relacionados à evolução das espécies e às funcionalidades das estruturas orgânicas [Edgar and Batzoglou 2006] dá-se o nome de Alinhamento Múltiplo de Sequências (*Multiple Sequence Alignment* - MSA). O alinhamento múltiplo é um passo essencial utilizado em diferentes áreas de pesquisa da bioinformática, como por exemplo, evolução das espécies [Thompson et al. 2011], análise de domínios [Thompson et al. 2011], análise de co-evolução [Goh and Cohen 2002], inferência filogenética [Mirarab and Warnow 2011], e predição de funcionalidades das proteínas [Kemena and Notredame 2009] ou de suas estruturas tridimensionais [Przybylski and Rost 2002].

Uma grande quantidade de ferramentas de alinhamento múltiplo de sequências foram propostas nas últimas quatro décadas [Gotoh 2014], algumas das quais continuaram a evoluir ao longo dos anos [Katoh and Toh 2008]. Curiosamente, a maioria das ferramentas contemporâneas ainda são baseadas em algoritmos progressivos. O Alinhamento Progressivo é considerado uma das primeiras heurísticas de alinhamento múltiplo [Feng and Doolittle 1987], sendo seu fluxo de trabalho composto por três etapas: Na primeira etapa, todas as sequências são comparadas entre si, em pares, gerando uma matriz triangular onde cada elemento é o resultado de cada comparação representado por um *score* de similaridade; A segunda etapa cria uma Árvore Guia binária que identifica a ordem em que as sequências devem ser alinhadas na última etapa; Esta segue a Árvore Guia, das folhas para a raiz, realizando os alinhamentos par a par. Ao chegar a raiz, todas as sequências terão sido alinhadas, gerando o alinhamento múltiplo.

As ferramentas de Alinhamento Progressivo diferem entre si pela combinação de técnicas adotadas na implementação das três etapas. Avaliações destas ferramentas têm mostrado que os tempos de execução e as qualidades dos alinhamentos podem variar consideravelmente dependendo das técnicas empregadas e das sequências sendo alinhadas [Edgar and Batzoglou 2006]. Por ser uma heurística, o alinhamento ótimo não é garantido. Assim, cientistas têm que fazer uso de uma ou mais ferramentas disponíveis, esperando que pelo menos um dos alinhamentos gerados tenha uma qualidade satisfatória.

Nesse contexto, o objetivo deste trabalho é propor e implementar uma *metaferramenta* – uma ferramenta que integra as técnicas empregadas por ferramentas existentes de forma unificada. Sua execução é coordenada por um *workflow* que efetivamente permite avaliar combinações novas e existentes de técnicas usadas para gerar alinhamentos progressivos. Três vantagens principais podem ser destacadas nesta abordagem: (i) produzir múltiplas soluções que não somente contemplem as produzidas pelas principais ferramentas existentes, mas também gerar novas soluções com potencial de ser a mais adequada; (ii) evitar repetir a execução de uma mesma técnica desnecessariamente, o que pode acontecer quando se usa múltiplas ferramentas distintas; (iii) permitir a execução paralela de estágios independentes da metaferramenta controlada por meio da criação de um *workflow*. Dentre estas vantagens, destaca-se o item (ii), uma vez que a primeira etapa do Alinhamento Progressivo é a mais demorada e deixará de ser executada, reduzindo os custos computacionais. Além disso, os resultados obtidos mostram um ganho de quase 9 vezes quando a metaferramenta, se aproveitando das características descritas no itens (ii) e (iii), é executada em 8 processadores.

O restante deste trabalho está dividido da seguinte forma: a próxima seção descreve o Alinhamento Progressivo e três das principais ferramentas de alinhamento múltiplo que utilizam essa heurística; a Seção 3 descreve a metaferramenta proposta; em seguida, na Seção 4, são apresentados e analisados alguns resultados iniciais obtidos; por fim, são apresentadas algumas conclusões, bem como trabalhos futuros.

2. Referencial Teórico

Para contextualizar o leitor a cerca do tema que envolve este trabalho, esta seção apresenta um resumo sobre Alinhamento Progressivo (subseção 2.1), relacionando as técnicas mais utilizadas em cada etapa. O Alinhamento Progressivo, além de ser uma classe de heurísticas utilizada até hoje, ainda é a base para outras heurísticas para alinhamento múltiplo, como o Alinhamento Iterativo e o Alinhamento baseado em Consistência. Em seguida, são apresentadas as ferramentas mais utilizadas que implementam essa heurística.

2.1. Alinhamento Progressivo

O método heurístico de Alinhamento Progressivo foi criado por Feng e Doolittle [Feng and Doolittle 1987]. Nele, os autores apresentam um método heurístico dividido em três etapas (como ilustrado na Figura 1) para realizar o alinhamento de múltiplas sequências genéticas baseado no alinhamento par a par das sequências.

Na primeira etapa do método, Feng e Doolittle utilizam o algoritmo de Needleman e Wunsch [Needleman and Wunsch 1970] para alinhar as n sequências, duas a duas (totalizando $\frac{n\times(n-1)}{2}$ alinhamentos par a par), gerando um *Score* de Similaridade para

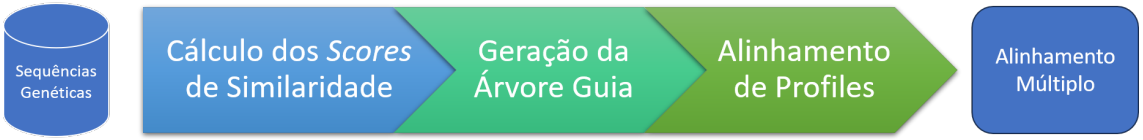

Figura 1. Etapas do Alinhamento Progressivo

cada alinhamento, que são armazenados numa matriz triangular denominada Matriz de Similaridades. Com o passar do tempo, novas técnicas para a geração desta Matriz de Similaridades foram desenvolvidas [Thompson et al. 1994] [Katoh and Toh 2008].

A partir da Matriz de Similaridades, a ordem para a realização dos alinhamentos é dada por uma árvore filogenética binária, gerada por um algoritmo de agrupamento hierárquico, que servirá de guia para o alinhamento final. A ideia geral é que sequências mais semelhantes entre si, ou seja, com os maiores *scores*, sejam alinhadas primeiro. As técnicas para a geração da Matriz de Similaridades e os algoritmos de agrupamento hierárquico implementados neste trabalho serão apresentados na subseção 3.2.

Durante a etapa final de Alinhamento de *Profiles*, a Árvore Guia é percorrida das folhas para a raiz, gerando um *Profile* [Eddy 1998] para cada nó não folha, que representa o alinhamento dos dois ramos (cada um podendo ser um outro *profile* ou uma sequência), até que o alinhamento completo seja produzido.

2.2. Trabalhos Relacionados

Desde a proposta inicial de Feng e Doolittle, as ferramentas de Alinhamento Progressivo estão em constante evolução, muitas vezes acrescentando novas etapas. Dentre as ferramentas mais populares encontram-se o CLUSTAL [Higgins and Sharp 1988], o CLUSTAL W [Thompson et al. 1994] e o MAFFT [Katoh et al. 2002], nas opções FFT-NS1 e NW-NS1.

Na primeira etapa, o CLUSTAL utiliza o algoritmo descrito em [Bashford et al. 1987] para o Cálculo dos *Scores* de Similaridade. Para a geração da Árvore Guia, a ferramenta utiliza o *Unweighted Pair Group Method using Arithmetic averages* (UPGMA) [Sokal and Michener 1958]. Por fim, o algoritmo de Wilbur e Lipman [Wilbur and Lipman 1984] é utilizado para realizar o Alinhamento dos *Profiles*, seguindo a ordem estabelecida pela Árvore Guia.

Além das técnicas implementadas no CLUSTAL, o CLUSTAL W [Thompson et al. 1994], uma segunda versão do CLUSTAL, permite utilizar, na primeira etapa, o algoritmo de Myers e Miller [Myers and Miller 1988] para o Cálculo dos *Scores* de Similaridade e o Neighbor-Joining (NJ) [Saitou and Nei 1987] foi adicionado na segunda etapa do Alinhamento Progressivo. Por fim, o algoritmo de Myers e Miller passou a ser usado também como método para alinhamento de *profiles*.

Desenvolvido em 2002 por Katoh *et al.* [Katoh et al. 2002], o *Multiple Alignment using Fast Fourier Transform* (MAFFT) converte as sequências em vetores de volume e polaridade de cada resíduo da sequência genética. A partir da conversão, é calculada a correlação entre as transformadas rápidas de Fourier dos vetores para cada par de sequências (FFT-NS1). Usando a correlação como *score* de similaridade, a Árvore

Guia é gerada pelo UPGMA e o alinhamento de *profiles* é feito utilizando o algoritmo de Myers e Miller. Ele possui ainda a opção de utilizar o algoritmo de Myers e Miller para a geração dos *scores* de similaridade (NW-NS1). Também existem versões mais recentes que tentam melhorar o alinhamento final iterativamente.

O trabalho de [João Jr et al. 2022] apresentou um arcabouço monolítico e sequencial para gerar múltiplos alinhamentos (inclusive progressivos) para um mesmo conjunto de sequências, combinando técnicas existentes nas ferramentas mais populares e utilizadas pela comunidade científica. Os resultados apresentados mostram a alta qualidade dos alinhamentos gerados pelo arcabouço, quando comparado com duas das principais ferramentas de alinhamento múltiplo da literatura (o T-COFFEE [Notredame et al. 2000] e o ProbCons [Do et al. 2005], ambos da classe de Alinhamento baseado em Consistência). Porém, o arcabouço não é capaz de aproveitar os dados de técnicas já executadas e não permite a execução paralela das diferentes etapas do alinhamento.

Diferentemente das propostas anteriores, que somente utilizam um conjunto de técnicas, este trabalho adota uma estratégia de aproveitar múltiplas técnicas em cada uma das etapas. A metaferramenta proposta realiza múltiplos alinhamentos progressivos de uma forma mais eficiente na tentativa de aumentar a probabilidade de construir alinhamentos de qualidade. Esta abordagem baseada na agregação de técnicas mostrou que a antiga classe de Alinhamento Progressivo voltou a ser competitiva em relação as classes de Alinhamento Iterativo e de Alinhamento baseado em Consistência. Uma vez que a capacidade da metaferramenta gerar alinhamentos de alta qualidade já foi detalhadamente analisada em [João Jr et al. 2022] e [João Jr et al. 2023], o objetivo deste trabalho é avaliar o desempenho da metaferramenta. Implementando a proposta por meio de um *workflow* tem-se as seguintes vantagens: evitar a reexecução de uma mesma técnica em mais de um alinhamento, e; permitir a execução paralela de etapas independentes da metaferramenta.

3. Metaferramenta para Alinhamento Progressivo

A metaferramenta apresentada neste trabalho se baseia nas etapas do Alinhamento Progressivo (subseção 2.1). Porém, ao invés de gerar apenas um Alinhamento Progressivo, a metaferramenta gera múltiplos alinhamentos para permitir ao pesquisador escolher a melhor opção. A subseção 3.1 descreve o *framework* para Alinhamento Progressivo que serviu de base para a implementação da metaferramenta. Em seguida, é apresentada a arquitetura da metaferramenta proposta (subseção 3.2). Por fim, a subseção 3.3 descreve a implementação da solução proposta por meio de um *workflow*.

3.1. Arcabouço Monolítico para Alinhamento Progressivo

A metaferramenta apresentada neste trabalho se baseia nas etapas do Alinhamento Progressivo, sendo utilizadas, para as duas primeiras etapas, as técnicas do arcabouço monolítico avaliadas em [João Jr et al. 2022]. Tal arcabouço foi implementado com base no código fonte em C++ do CLUSTAL W [Thompson et al. 1994], de onde foram utilizadas as implementações das técnicas FULL, QUICK, NJ e UPGMA, além do alinhamento de *profiles* e da manipulação dos arquivos lidos e gerados, seguindo formatos adotados pela comunidade, como por exemplo, o formato de arquivo FASTA para sequências genéticas.

Para a primeira etapa (Cálculo dos *Scores* de Similaridade), foram implementadas as seguintes técnicas presentes em ferramentas tradicionais:

- **FULL:** cada *Score* de Similaridade é calculado como a porcentagem de identidade do alinhamento obtido usando o algoritmo de Myers and Miller [Myers and Miller 1988];
- **QUICK:** calcula os *scores* utilizando o método rápido da primeira versão do CLUSTAL, que conta as tuplas correspondentes em cada par de sequências;
- **LCS:** define como *Score* de Similaridade o tamanho da maior subsequência comum (*Longest Common Subsequence* - LCS) entre duas sequências, dividido pelo tamanho da menor sequência. O LCS é gerado usando o algoritmo de Hirschberg implementado em [João Jr et al. 2019];
- **KMERS:** os *scores* são o número de *k-mers* (subsequências de tamanho k que duas sequências tem em comum), dividido pelo tamanho da menor sequência.

Para a segunda etapa (Geração da Árvore Guia), além dos dois algoritmos mais comuns, disponíveis no CLUSTAL W, o UPGMA e o NJ, o arcabouço também implementa duas outras técnicas denominadas SLMIN e SLMAX, descritas em [João Jr et al. 2023]. Por sua vez, a última etapa utiliza o Alinhamento de *Profiles* proveniente do CLUSTAL W. É importante destacar que as técnicas implementadas representam o que há de mais relevante na atualidade. Ainda mais no caso da segunda etapa, que implementa todas as 4 técnicas encontradas na literatura para a geração da árvore guia.

3.2. Arquitetura

A arquitetura atual da metaferramenta, ilustrada na Figura 2, segue as etapas do Alinhamento Progressivo da Figura 1, a saber: Cálculo dos *Scores* de Similaridade, Geração da Árvore Guia e Alinhamento de *Profiles*. Para cada etapa, as técnicas descritas na subseção 3.1 são implementadas como programas distintos baseados no código fonte do CLUSTAL W, mas executados como processos independentes.

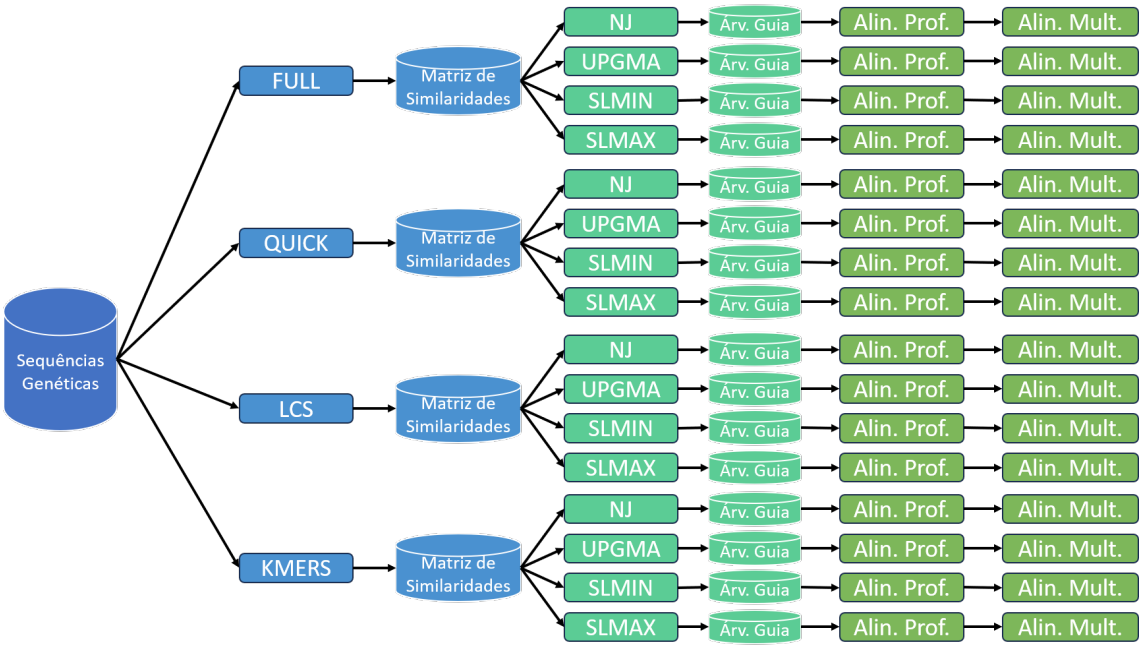

Figura 2. Arquitetura da metaferramenta para Alinhamento Progressivo

A conexão entre as etapas é feita pela comunicação de informações por arquivos específicos. Para que os *Scores* de Similaridade possam ser utilizados na Geração da

Árvore Guia, um arquivo com a Matriz de Similaridade é gerado pelo processo que executa a técnica utilizada na primeira etapa do Alinhamento Progressivo. Posteriormente, este aquivo é lido pelo processo que implementa a técnica utilizada na segunda etapa. Analogamente, para que o Alinhamento de *Profiles* possa seguir a ordem com que as sequências serão alinhadas, um arquivo com a Árvore Guia é gerado pelo processo que executa a técnica utilizada na segunda etapa para, posteriormente, ser lido pelo processo que fará o Alinhamento de *Profiles*.

Com esta arquitetura atualmente são gerados 16 alinhamentos (4 técnicas da primeira etapa $\times$ 4 técnicas da segunda etapa) para uma única entrada (instância de alinhamento múltiplo). Se estes alinhamentos fossem gerados utilizando-se o fluxo tradicional do Alinhamento Progressivo e com cada etapa sendo executada por um processo, ter-se-ia um total de 48 processos executados (4 técnicas da primeira etapa $\times$ 4 técnicas da segunda etapa $\times$ 3 etapas). Entretanto, a metaferramenta descrita neste trabalho foi desenvolvida para reutilizar as Matrizes de Similaridade geradas na primeira etapa, evitando assim, repetição de trabalho (conforme mostra a Figura 2). Com a reutilização, cada processo relativo a cada técnica da primeira etapa só precisa ser executado uma única vez, evitando a execução de 12 processos. É importante ressaltar que, como será mostrado nos experimentos realizados na Seção 4, esses processos, no Alinhamento Progressivo, são os que tendem a demorar mais tempo, o que aumenta a eficiência da solução proposta.

3.3. *Workflow* para Alinhamento Progressivo

A criação do *workflow* da metaferramenta apresentada neste trabalho, foi realizada utilizando o Nextflow [Di Tommaso et al. 2017]. Desenvolvido para gerenciar *workflows* na área de bioinformática, o Nextflow pretende prover a execução de aplicações de forma paralela, eficiente e tolerante a falhas, além de oferecer proveniência e rastreabilidade. Ele possui suporte nativo para a execução de *containers* em ambientes de Nuvens Computacionais, além de seguir os padrões estabelecidos pela *Workflows Community Initiative* [1].

O Nextflow segue o paradigma de *dataflow* e é composto, basicamente, por duas entidades: processos e canais. Os processos seguem a filosofia dos processos UNIX, possuindo entradas e saídas, e são responsáveis pela execução propriamente dita dos processos que compõem os *workflows*. As entradas e saídas podem ser valores, como a quantidade de sequências ou a opção a ser passada para o processo de alinhamento, ou podem ser arquivos. Os canais seguem a filosofia FIFO dos PIPES UNIX e servem para realizar a comunicação entre processos.

No *workflow* desenvolvido para este trabalho, exibido na Figura 3, foram criados três processos, um para cada etapa do Alinhamento Progressivo, conectados por canais com os arquivos gerados nas etapas anteriores (*pipes* horizontais, ou cilindros, conectando os processos, representados por retângulos). Para que todas as técnicas de cada processo possam ser executadas, um canal artificial com os nomes das técnicas foi criado e associado a entrada dos processos que executam as duas primeiras etapas (*pipes* horizontais com os nomes das técnicas).

[1] https://workflows.community/

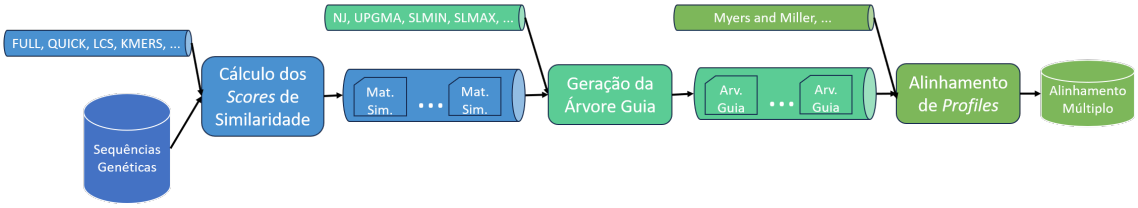

Figura 3. *Workflow* para Alinhamento Progressivo

4. Análise Experimental

Para analisar o desempenho da metaferramenta, foram usadas sequências de proteínas da família PF00005 [Hung et al. 1998] pertencentes ao *benchmark* PFAM [Finn et al. 2014]. Tais proteínas são transportadores ABC responsáveis pela translocação de uma variedade de compostos através das membranas biológicas, sendo a maior família de proteínas em muitas bactérias completamente sequenciadas. Uma outra razão para a escolha deste *benchmark*, foi a significativa quantidade de sequências disponíveis para analisar alinhamentos múltiplos com diferentes quantidades e tamanhos de sequências.

Desta família, foram selecionadas sequências pertencentes a um dos três grupos a seguir: tamanhos menores que 100 resíduos; entre 100, inclusive, e 200, e; entre 200, inclusive, e 300. A quantidade total de sequências que compõem estes três grupos representa 70% da família PF00005. Das milhares de sequências que compõem estes grupos, foram selecionadas amostras de 100, 200, 300, 400, 500 e 1000 sequências para avaliar o impacto do número de sequências no desempenho da metaferramenta. Esta combinação de números e tamanhos de sequências permite avaliar a eficácia da proposta em relação a diferentes tamanhos de problemas de alinhamento múltiplo, especialmente para proteínas. As médias dos tempos de três execuções dos cenários utilizados para essa avaliação são exibidas na Tabela 1. Os experimentos foram executados em instâncias da família C7g da AWS EC2 equipadas com processadores AWS Graviton 3 com frequência de 2,6 GHz de clock. Nestas instâncias, a quantidade de memória alocada é de 2 GiB para cada CPU. Neste trabalho, foram utilizadas instâncias de máquinas virtuais com 1, 2, 4 e 8 CPUs, cuja utilização é gerenciada pelo Nextflow.

O arcabouço monolítico que originou a metaferramenta apresentada neste trabalho teve a qualidade dos alinhamentos gerados avaliada em [João Jr et al. 2023], onde os resultados mostraram que a qualidade é, no mínimo, mantida e muitas vezes melhorada em relação às ferramentas existentes. O arcabouço foi implementado com base no código fonte do CLUSTAL W e para avaliar a sobrecarga da sua implementação em relação à implementação original do CLUSTAL W, ambos foram executados com as mesmas opções de técnicas (as originais do CLUSTAL W) para o alinhamento de 500 sequências de cada um dos 3 grupos descritos anteriormente. As médias dos tempos de 3 execuções para os três grupos foram, respectivamente, $9,46s$, $29,54s$ e $84,55s$ para o CLUSTAL W e $9,47s$, $29,65s$ e $84,70s$ para o arcabouço. Tais tempos mostram um acréscimo que varia de $0,11\%$ a $0,36\%$, indicando que o código utilizado no arcabouço, baseado no CLUSTAL W e com a adição de outras técnicas, não altera significativamente o tempo de execução para a geração de alinhamentos múltiplos.

Para a construção do *workflow* descrito na Seção 3.3, o código do arcabouço foi particionado, gerando programas separados para implementar cada técnica utilizada em

Tabela 1. Tempos de execução (em segundos), sobrecargas, fatores de ganho e *Speedups* obtidos na execução dos experimentos com a metaferramenta

Tamanho das Sequências	Até 100					
Número de Sequências:	100	200	300	400	500	1000
Tamanho Médio das sequências:	78,8	77,4	77,4	77,8	77,3	78,0
Arcabouço monolítico (s)	2,65	8,88	18,59	32,35	49,40	193,06
Sequencial sem reaproveitamento (s)	3,01	9,98	20,95	36,41	55,78	219,22
Sobrecarga das etapas separadas	13,60%	12,39%	12,71%	12,54%	12,93%	13,55%
Sequencial com reaproveitamento (s)	1,66	4,88	9,67	16,25	24,40	92,50
Ganho do reaproveitamento	1,81	2,04	2,17	2,24	2,29	2,37
Nextflow com 1 processo (s)	14,62	18,18	24,11	31,41	40,06	116,08
Nextflow com 2 processos (s)	8,65	10,21	13,02	16,94	21,51	59,99
Nextflow com 4 processos (s)	5,30	6,31	7,81	9,66	11,79	31,67
Nextflow com 8 processos (s)	4,79	5,53	5,97	7,55	9,28	26,37
Sobrecarga do Nextflow	778,96%	272,29%	149,33%	93,27%	64,14%	25,50%
Speedup Nextflow com 2 CPUs	1,69	1,78	1,85	1,85	1,86	1,94
Speedup Nextflow com 4 CPUs	2,76	2,88	3,09	3,25	3,40	3,67
Speedup Nextflow com 8 CPUs	3,05	3,29	4,04	4,16	4,32	4,40
Ganho total Nextflow com 2 CPUs	0,31	0,87	1,43	1,91	2,30	3,22
Ganho total Nextflow com 4 CPUs	0,50	1,41	2,38	3,35	4,19	6,10
Ganho total Nextflow com 8 CPUs	0,55	1,60	3,12	4,28	5,32	7,32
Tamanho das Sequências	100 a 200					
Número de Sequências:	100	200	300	400	500	1000
Tamanho Médio das sequências:	158,7	161,3	160,9	157,1	156,5	156,1
Arcabouço monolítico (s)	8,43	29,68	63,02	104,68	161,46	617,86
Sequencial sem reaproveitamento (s)	8,88	30,91	65,62	109,15	165,94	644,88
Sobrecarga das etapas separadas	5,34%	4,16%	4,12%	4,27%	2,77%	4,37%
Sequencial com reaproveitamento (s)	4,47	12,99	25,63	41,34	61,26	223,52
Ganho do reaproveitamento	1,99	2,38	2,56	2,64	2,71	2,89
Nextflow com 1 processo (s)	18,75	28,13	42,01	59,38	81,04	254,94
Nextflow com 2 processos (s)	9,82	14,51	21,57	30,10	40,99	127,29
Nextflow com 4 processos (s)	6,04	8,37	12,17	16,70	23,43	77,57
Nextflow com 8 processos (s)	4,80	6,82	10,51	14,84	20,71	70,81
Sobrecarga do Nextflow	319,54%	116,58%	63,92%	43,63%	32,28%	14,06%
Speedup Nextflow com 2 CPUs	1,91	1,94	1,95	1,97	1,98	2,00
Speedup Nextflow com 4 CPUs	3,10	3,36	3,45	3,56	3,46	3,29
Speedup Nextflow com 8 CPUs	3,91	4,12	4,00	4,00	3,91	3,60
Ganho total Nextflow com 2 CPUs	0,86	2,05	2,92	3,48	3,94	4,85
Ganho total Nextflow com 4 CPUs	1,39	3,55	5,18	6,27	6,89	7,97
Ganho total Nextflow com 8 CPUs	1,76	4,35	6,00	7,05	7,80	8,73
Tamanho das Sequências	200 a 300					
Número de Sequências:	100	200	300	400	500	1000
Tamanho Médio das sequências:	250,8	250,3	252,0	251,2	251,5	253,0
Arcabouço monolítico (s)	20,49	74,72	165,78	286,83	444,17	1748,92
Sequencial sem reaproveitamento (s)	21,00	76,21	167,95	291,90	451,89	1776,20
Sobrecarga das etapas separadas	2,52%	2,00%	1,31%	1,77%	1,74%	1,56%
Sequencial com reaproveitamento (s)	8,69	27,23	56,43	93,79	141,15	523,72
Ganho do reaproveitamento	2,42	2,80	2,98	3,11	3,20	3,39
Nextflow com 1 processo (s)	23,80	43,94	75,51	114,25	164,22	563,28
Nextflow com 2 processos (s)	12,49	22,62	38,29	57,90	82,34	282,80
Nextflow com 4 processos (s)	7,39	14,57	27,64	45,71	68,63	262,63
Nextflow com 8 processos (s)	6,33	13,84	26,93	44,24	66,88	258,49
Sobrecarga do Nextflow	173,92%	61,38%	33,82%	21,81%	16,56%	7,55%
Speedup Nextflow com 2 CPUs	1,91	1,94	1,97	1,97	1,99	1,99
Speedup Nextflow com 4 CPUs	3,22	3,02	2,73	2,50	2,39	2,14
Speedup Nextflow com 8 CPUs	3,76	3,18	2,80	2,58	2,46	2,18
Ganho total Nextflow com 2 CPUs	1,64	3,30	4,33	4,95	5,39	6,18
Ganho total Nextflow com 4 CPUs	2,77	5,13	6,00	6,28	6,47	6,66
Ganho total Nextflow com 8 CPUs	3,24	5,40	6,16	6,48	6,64	6,77

cada uma das três etapas do Alinhamento Progressivo. Com o intuito de avaliar o impacto dessa divisão em relação a execução do arcabouço monolítico, os programas foram executados sequencialmente na ordem em que geram os alinhamentos, combinados de forma a gerar todos os 16 alinhamentos possíveis para a combinação das 4 técnicas da etapa de Cálculo dos *Scores* de Similaridade com as 4 técnicas da Geração da Árvore Guia. Os tempos para a execução do arcabouço monolítico e dos processos separados executados sequencialmente podem ser observados nas linhas **Arcabouço monolítico** e **Sequencial sem reaproveitamento** da Tabela 1. As linhas **Sobrecarga das etapas separadas** mostram percentualmente $(\frac{Sequencial\ sem\ reaproveitamento - Arcabouco\ monolitico}{Arcabouco\ monolitico} \times 100)$ o quanto a execução sequencial em separado das etapas do Alinhamento Múltiplo apresenta de sobrecarga. Como era de se esperar, todos os percentuais são positivos, indicando que a utilização de arquivos temporários para o envio dos dados entre as etapas gera atraso na execução. Tal atraso é proporcionalmente maior para o primeiro grupo com sequências de tamanho menores que 100 e diminui conforme o tamanho das sequências aumenta, chegando a menos de 2% para os maiores casos (tamanho entre 200 e 300, com 300 sequências ou mais).

Com a possibilidade de executar cada técnica de cada etapa separadamente, surge a possibilidade de reaproveitamento dos dados intermediários gerados ao final da primeira etapa. Para medir o ganho desse reaproveitamento, todos os alinhamentos gerados pelas 16 possíveis combinações de técnicas foram executados utilizando o reaproveitamento dos dados gerados pela primeira etapa. Os tempos dessas execuções e os ganhos obtidos com o reaproveitamento estão, respectivamente, nas linhas **Sequencial com reaproveitamento** e **Ganho do reaproveitamento** $(\frac{Sequencial\ sem\ reaproveitamento}{Sequencial\ com\ reaproveitamento})$. Esse ganho atinge $3,39$ vezes para o maior exemplo, aumentando com o tamanho das sequências e a sua quantidade.

Na Figura 2, observa-se que o fluxo dos dados gerados pelas etapas permite a execução concorrente entre si das técnicas da primeira etapa e entre as técnicas da segunda etapa. Para implementar tal solução, utilizou-se o *workflow* descrito na Seção 3.3. Os tempos para as execuções da metaferramenta com $N = 1$, 2, 4 e 8 processos concorrentes, seus ganhos em relação à execução com apenas um processo concorrente e em relação ao arcabouço monolítico são mostrados, respectivamente, nas linhas **Nextflow com** N **processos**, **Speedup Nextflow com** N **CPUs** e **Ganho total Nextflow com** N **CPUs**. A **Sobrecarga do Nextflow** $(\frac{Nextflow\ com\ 1\ processo - Sequencial\ com\ reaproveitamento}{Sequencial\ com\ reaproveitamento} \times 100)$ ilustra o custo do gerenciamento realizado pelo Nextflow em relação ao tempo de execução necessário para a geração dos alinhamentos. Apesar de ser alto para problemas pequenos, diminui significativamente quanto maior o número e tamanhos das sequências.

O *Speedup* obtido pela metaferramenta com mais de um processo, quando comparado a execução com apenas um processo, é de no máximo 4, e, para o grupo com as maiores sequências, diminui com o aumento do número das mesmas. Para compreender esse comportamento, faz-se necessário analisar os tempos de execução de cada técnica de cada etapa do Alinhamento Progressivo. Na Tabela 2, são apresentados os percentuais do tempo gasto com cada técnica durante a execução da metaferramenta para a geração de 16 alinhamentos diferentes. Os percentuais relativos às técnicas da segunda etapa (NJ, UPGMA, SLMIN e SLMAX) são os obtidos pela execução de 4 vezes cada uma dessas técnicas, visto que cada técnica da segunda etapa é executada uma vez para cada técnica

da primeira etapa. Similarmente, o percentual obtido para o Alinhamento de Profiles equivale à execução do mesmo 16 vezes.

Tabela 2. Percentual do tempo que cada técnica gasta para os 16 alinhamentos

FULL	QUICK	LCS	KMERS	UPGMA	SLMIN	SLMAX	NJ	Alin. de Profiles
47.8%	7.0%	26.1%	0.3%	0.2%	0.2%	0.2%	1.5%	16.7%

Ao observar-se a Tabela 2, percebe-se que os tempos necessários para a execução das técnicas da primeira etapa equivalem à $81,2\%$ do tempo total, sendo a técnica FULL a mais lenta, responsável por $47,8\%$ do tempo, seguida pela técnica LCS ($26,1\%$). A diferença entre o tempo de execução da técnica FULL para as demais acaba por limitar o *speedup* quando se comparam os tempos de execução da metaferramenta com mais de um processo concorrente em relação a um único processo. Outra razão para a limitação do *speedup* para 8 processos concorrentes é o fato de existirem implementadas na metaferramenta, no momento, 4 técnicas para a segunda etapa e uma para a terceira etapa do Alinhamento Progressivo.

Porém, tal diferença nos tempos de execução da primeira etapa também é o fator que permite um maior ganho total da metaferramenta com as técnicas implementadas atualmente, apresentado nas 3 últimas linhas da Tabela 1. Com o uso da metaferramenta, as técnicas da primeira etapa só precisam ser executadas uma única vez, ao invés de 4 (o número de técnicas da segunda etapa). Desta forma, o tempo é reduzido em $(N2E-1)\times(TF+TQ+TL+TK)$, onde N2E é o número de técnicas da segunda etapa do Alinhamento Progressivo e TF, TQ, TL e TK são os tempos para executar as técnicas da primeira etapa (FULL, QUICK, LCS e KMERS, respectivamente). A combinação do reaproveitamento em conjunto com a execução orquestrada pelo *workflow* neste experimento permitiu um ganho total, em relação a versão monolítica, de até ≈ 9 vezes, executando em 8 processadores. Tal ganho será maior quanto maior for o reaproveitamento. Por fim, a separação e execução em paralelo das técnicas permitiram ganho de desempenho para todos os alinhamentos múltiplos executados em relação a versão monolítica, com exceção do menor tamanho de problema estudado (i.e. 100 sequências com até 100 resíduos), quando executado com 4 e 8 CPUs.

5. Conclusões e Trabalhos Futuros

Por ser uma heurística, o método de Alinhamento Progressivo para solucionar o problema de Alinhamento Múltiplo de Sequências genéticas não garante a geração do alinhamento ótimo. Assim sendo, cientistas se veem obrigados a escolher e executar diversas ferramentas para identificar o alinhamento mais adequado às suas necessidades.

O presente trabalho apresentou uma metaferramenta que implementa técnicas utilizadas em ferramentas existentes, combinou-as (por vezes de maneira inédita) e obteve resultados com ganho de desempenho de quase 9 vezes, mesmo quando utilizando 8 processos concorrentes. Tal metaferramenta possibilita a continuidade da pesquisa em diversos aspectos. A começar com a implementação de outras técnicas para as etapas do Alinhamento Progressivo, como por exemplo, acrescentar a técnica utilizada pelo ProbCons [Do et al. 2005] para o Cálculo dos *Scores* de Similaridade. Ainda mais interessante é ampliar as etapas do Alinhamento Progressivo com a inclusão de técnicas de

refinamento do Alinhamento Progressivo como, por exemplo, o Alinhamento Baseado em Consistência e o Alinhamento Iterativo, conforme sugerido em [João Jr et al. 2023], o que vai aumentar ainda mais o reaproveitamento e o desempenho, além de reduzir os custos computacionais.

Em relação ao desempenho, sabendo-se que a primeira etapa do Alinhamento Progressivo é a mais custosa, a paralelização dessa etapa, para a geração dos *Scores* de Similaridade com o FULL e o LCS, apresenta um grande potencial. Ainda mais se combinada essa paralelização com a utilização de múltiplas máquinas virtuais. Por fim, ainda poderiam ser analisadas outras soluções para o gerenciamento da metaferramenta além do Nextflow.

Agradecimentos

Os autores agradecem o apoio do CNPq aos projetos Universal 404087/2021-3 e CNPq/AWS 421828/2022-6.

Referências

Bashford, D., Chothia, C., and Lesk, A. M. (1987). Determinants of a protein fold: Unique features of the globin amino acid sequences. *Journal of Molecular Biology*, 196(1):199–216.

Di Tommaso, P., Chatzou, M., Floden, E. W., Barja, P. P., Palumbo, E., and Notredame, C. (2017). Nextflow enables reproducible computational workflows. *Nature Biotechnology*, 35(4):316–319.

Do, C. B., Mahabhashyam, M. S. P., Brudno, M., and Batzoglou, S. (2005). ProbCons: Probabilistic consistency-based multiple sequence alignment. *Genome research*, 15(2):330–40.

Eddy, S. R. (1998). Profile hidden Markov models. *Bioinformatics*, 14(9):755–763.

Edgar, R. C. and Batzoglou, S. (2006). Multiple sequence alignment. *Current Opinion in Structural Biology*, 16(3):368–373.

Feng, D.-F. and Doolittle, R. F. (1987). Progressive Sequence Alignment as a Prerequisite to Correct Phylogenetic Trees. *Journal of Molecular Evolution*, 25:351–360.

Finn, R. D., Bateman, A., Clements, J., Coggill, P., Eberhardt, R. Y., Eddy, S. R., Heger, A., Hetherington, K., Holm, L., Mistry, J., Sonnhammer, E. L., Tate, J., and Punta, M. (2014). Pfam: the protein families database. *Nucleic acids research*, 42.

Goh, C.-S. and Cohen, F. E. (2002). Co-evolutionary analysis reveals insights into protein–protein interactions. *Journal of Molecular Biology*, 324(1):177–192.

Gotoh, O. (2014). *Heuristic Alignment Methods*, chapter 2, pages 29–43. Humana Press, Totowa, NJ.

Higgins, D. G. and Sharp, P. M. (1988). CLUSTAL: A package for performing multiple sequence alignment on a microcomputer. *Gene*, 73(1):237 – 244.

Hung, L.-W., Wang, I. X., Nikaido, K., Liu, P.-Q., Ames, G. F.-L., and Kim, S.-H. (1998). Crystal structure of the ATP-binding subunit of an ABC transporter. *Nature*, 396(6712):703–707.

João Jr, M., Sena, A. C., and Rebello, V. E. F. (2019). On the parallelization of Hirschberg's algorithm for multi-core and many-core systems. *Concurrency and Computation: Practice and Experience*, 31(18):e5174.

João Jr, M., Sena, A. C., and Rebello, V. E. F. (2022). On using consistency consistently in multiple sequence alignments. In *2022 IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW)*, pages 152–161.

João Jr, M., Sena, A. C., and Rebello, V. E. F. (2023). On closing the inopportune gap with consistency transformation and iterative refinement. *PLoS ONE*, 18(7):1–24.

Katoh, K., Misawa, K., Kuma, K.-i., and Miyata, T. (2002). MAFFT: A novel method for rapid multiple sequence alignment based on fast fourier transform. *Nucleic Acids Research*, 30(14):3059–3066.

Katoh, K. and Toh, H. (2008). Recent developments in the MAFFT multiple sequence alignment program. *Briefings in Bioinformatics*, 9(4):286–298.

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

Mirarab, S. and Warnow, T. (2011). FastSP: linear time calculation of alignment accuracy. *Bioinformatics*, 27(23):3250–3258.

Myers, E. W. and Miller, W. (1988). Optimal alignments in linear space. *Bioinformatics*, 4(1):11–17.

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

Notredame, C., Higgins, D. G., and Heringa, J. (2000). T-Coffee: A Novel Method for Fast and Accurate Multiple Sequence Alignment. *Journal of Molecular Biology*, 302(1):205 – 217.

Przybylski, D. and Rost, B. (2002). Alignments grow, secondary structure prediction improves. *Proteins*, 46(2):197–205.

Saitou, N. and Nei, M. (1987). The Neighbor-joining Method: A New Method for Reconstructing Phylogenetic Trees. *Molecular Biology and Evolution*, 4(4):406–425.

Sokal, R. R. and Michener, C. D. (1958). A statistical method for evaluating systematic relationships. *The University of Kansas Science Bulletin*, 38(22):1409–1438.

Thompson, J. D., Higgins, D. G., and Gibson, T. J. (1994). CLUSTAL W: Improving the sensitivity of progressive multiple sequence alignment through sequence weighting, position-specific gap penalties and weight matrix choice. *Nucleic Acids Research*, 22(22):4673–4680.

Thompson, J. D., Linard, B., Lecompte, O., and Poch, O. (2011). A Comprehensive Benchmark Study of Multiple Sequence Alignment Methods: Current Challenges and Future Perspectives. *PLoS ONE*, 6(3).

Wilbur, W. J. and Lipman, D. J. (1984). The context dependent comparison of biological sequences. *SIAM Journal on Applied Mathematics*, 44(3):557–567.