UNIVERSIDADE FEDERAL DE SÃO CARLOS
CENTRO DE CIÊNCIAS BIOLÓGICAS E DA SAÚDE
PROGRAMA DE PÓS-GRADUAÇÃO EM GENÉTICA E EVOLUÇÃO

ALINHAMENTO DE SEQÜÊNCIAS BIOLÓGICAS COM O USO DE ALGORITMOS GENÉTICOS

Sabrina Oliveira Ogata

SÃO CARLOS – SP
2005

UNIVERSIDADE FEDERAL DE SÃO CARLOS
CENTRO DE CIÊNCIAS BIOLÓGICAS E DA SAÚDE
PROGRAMA DE PÓS-GRADUAÇÃO EM GENÉTICA E EVOLUÇÃO

ALINHAMENTO DE SEQÜÊNCIAS BIOLÓGICAS COM O USO DE ALGORITMOS GENÉTICOS

Sabrina Oliveira Ogata

Dissertação apresentada ao Programa de Pós-Graduação em Genética e Evolução do Centro de Ciências Biológicas e da Saúde da Universidade Federal de São Carlos, como parte dos requisitos para obtenção do título de Mestre em Genética e Evolução, área de concentração: Genética e Evolução.

SÃO CARLOS – SP
2005

Ficha catalográfica elaborada pelo DePT da Biblioteca Comunitária da UFSCar

O34as

Ogata, Sabrina Oliveira.
Alinhamento de seqüências biológicas com o uso de algoritmos genéticos / Sabrina Oliveira Ogata. -- São Carlos : UFSCar, 2005.
77 p.

Dissertação (Mestrado) -- Universidade Federal de São Carlos, 2005.

1. Genética – processamento de dados. 2. Bioinformática. 3. Algoritmos genéticos. 4. Alinhamento de seqüências. I. Título.

CDD: 575.10285 (20ª)

Orientador

Prof. Dr. Pedro Manoel Galetti Junior

“Nada na Biologia faz sentido exceto à luz da Evolução”
Theodosius Dobzhansky

Dedico este trabalho aos meus pais e ao meu esposo por sempre apoiarem-me e incentivarem nos momentos difíceis.

AGRADECIMENTOS

Ao Prof. Dr. Pedro Manoel Galetti Júnior, por me dar a oportunidade de continuar minhas pesquisas na área de Bioinformática.

Ao Prof. Dr. André Ponce de Leon F. de Carvalho, pela paciência, amizade e apoio que me deu durante todo o projeto. Além de todos os conhecimentos que me passou ao longo desses dois anos.

Aos meus pais, Oduvaldo e Silvia, que tiveram que aguentar as saudades de uma filha que não conseguiu visitá-los muitas vezes durante esses dois anos, e apesar disso sempre estavam me incentivando e apoiando.

Ao meu esposo Adriano, pelo seu carinho, compreensão e por sempre estar ao meu lado, ajudar-me muito e suportar-me nos momentos mais frágeis. Aishiteru, ooji-sama.

Aos meus irmãos, João e Beto, minhas cunhadas Rosana e Cláudia, e minha pequena sobrinha Alessandra, pelos poucos, mas proveitosos momentos de alegria que passamos juntos nesses dois anos.

A todos os meus amigos, mesmo aqueles que eu conversava apenas via email ou via icq (ou mesmo via Ragnarok Online), pelos momentos de “espairecimento da mente”.

Ao pessoal da secretaria da Genética, Regiane, Rose e Tatiane, por sempre me auxiliar e solucionar os problemas burocráticos do mestrado.

A todos aqueles que colaboraram direta ou indiretamente para a realização deste trabalho.

E por fim agradeço ao CNPq pelo apoio financeiro concedido aos meus estudos.

RESUMO

A comparação de seqüências de genomas de organismos é uma das aplicações computacionais mais utilizadas por biólogos moleculares. Esta operação serve de suporte para outros processos como, por exemplo, a determinação da estrutura tridimensional de proteínas. Durante os últimos anos, tem-se observado uma grande expansão na utilização de Algoritmos Evolutivos, especialmente Algoritmos Genéticos (AGs), para problemas de otimização, como os de alinhamento de seqüências de dna e proteínas. Os AGs são técnicas de busca inspiradas em mecanismos de seleção natural e da genética. Neste trabalho, um AG foi utilizado para o desenvolvimento de uma ferramenta computacional para o problema de alinhamento de pares de seqüências. Uma comparação foi realizada com o uso da ferramenta bl2seq (do pacote de ferramentas Blast) afim de se checar a desempenho do AG. Utilizou-se para isso seqüências de diversos tamanhos e graus de similaridade. Os resultados demonstraram alguns casos específicos onde o AG superou a ferramenta bl2seq. A principal contribuição deste trabalho está na área de Bioinformática, com o emprego de uma nova metodologia, que posteriormente possa ser utilizada como mais uma opção para análises de seqüências em projetos genomas e para refinamento dos resultados obtidos por outras ferramentas usualmente utilizadas, como o Blast.

Palavras-chave: Algoritmos Genéticos; Alinhamentos de seqüências; Bioinformática.

ABSTRACT

The comparison of genome sequences from different organisms is one of the computational application most frequently used by molecular biologists. This operation serves as support for other processes as, for instance, the determination of the three-dimensional structure of the proteins. During the last years, a significative increase has been observed in the use of Genetic Algorithms (GA) for optimization problems as, for example, dna or protein sequence alignment. GAs are search techniques inspired in mechanisms from Natural Selection and Genetics. In this work, a GA was used to development a computacional tool for the sequence alignment problem. A benchmark comparison of this program with the bl2seq tool (from Blast package) is showed using some sequences, varying in similarity and length. In some cases the GA overhelmed the bl2seq tool. This work contributes to the Bioinformatic field by the use of a novel methodology that can use as an option for sequence analysis in genome projects and to enhance the results of other tools such like Blast.

Keywords: Genetic Algorithms; Pairwise Alignment; Bioinformatics.

Sumário

Lista de Figuras

Lista de Tabelas

1. INTRODUÇÃO

O cromossomo inteiro do fago X174 com 5.387 nucleotídeos foi seqüenciado na década de 70. No mesmo período, Sanger e seus colaboradores desenvolveram a técnica de terminação de cadeia. Isso motivou o desenvolvimento de outras técnicas que, com o auxílio da automação e dos computadores, otimizaram o tempo de seqüenciamento e melhoraram a análise de dados para a obtenção de resultados mais confiáveis [1].

A descoberta de enzimas de restrição (usadas na preparação de segmentos específicos de cromossomos); o aperfeiçoamento da eletroforese em gel até o ponto em que fragmentos de DNA que diferem em comprimento por um único nucleotídeo pudessem ser separados e o desenvolvimento de técnicas de clonagem de genes (preparação de grandes quantidades de um determinado gene ou seqüência de DNA de interesse) viabilizaram o seqüenciamento de genomas inteiros de vários organismos, seja animal ou vegetal.

Os esforços para seqüenciar o genoma humano geraram projetos de seqüenciamento de genoma de outros organismos menos complexos, muitos dos quais já foram completados [1]. Isso trouxe como conseqüência um acúmulo de dados muito rápido, dificultando sua interpretação.

Em paralelo com o desenvolvimento da Biologia molecular, por volta de 1980, os computadores começaram a ter uma capacidade cada vez

maior em termos de poder de processamento e velocidade. Assim, a crescente geração de dados biológicos, aliada ao desenvolvimento de computadores cada vez mais poderosos, fez com que uma nova área de pesquisa multidisciplinar surgisse, a Bioinformática.

1.1. A Bioinformatica

A Bioinformática diz respeito à utilização de técnicas e ferramentas da computação para a resolução de problemas da biologia [2]. As aplicações da área englobam a resolução de problemas que vão desde a comparação de seqüências e montagem de fragmentos até a identificação e análise da expressão de genes e determinação da estrutura de proteínas. Uma área importante da Biologia que necessita da aplicação da computação é a Biologia Molecular [3].

Diversas áreas da Computação estão inclusas nessas aplicações como, por exemplo, a Teoria da Computação, principalmente por meio da formulação de algoritmos para diversos novos problemas que surgiram em anos recentes nessa área, e Bancos de Dados, necessários para lidar com as gigantescas massas de informação oriundas dos avanços em técnicas laboratoriais [3].

Dentre as ferramentas utilizadas na área de Bioinformática, diversas têm sido propostas para alinhamento de seqüências. As mais utilizadas empregam a chamada programação dinâmica para a obtenção de um alinhamento entre pares de seqüências [3]. O pacote de ferramentas

BLAST [4] enquadra-se nesse tipo de ferramenta, sendo a mais utilizada pelos bioinformatas.

Observa-se que pesquisas recentes em alinhamento de sequências incluem a utilização de técnicas alternativas, como Algoritmos Genéticos (AGs) [5],[6],[7],[8] com o objetivo de obtenção de soluções melhores que aqueles resultados por métodos tradicionais.

1.2. Algoritmos Genéticos

Os AGs são técnicas de otimização e busca inspiradas em mecanismos de Seleção Natural e Genética [9]. A utilização de AGs em aplicações geralmente começa com a definição de conjunto de soluções iniciais, chamado de população, definida em geral de forma aleatória. Esta passa por um processo iterativo até que um determinado critério de parada seja satisfeito. A cada iteração, são aplicados operadores genéticos de seleção, crossover e mutação.

1.3. Ferramenta computacional desenvolvida

Neste trabalho, utilizou-se a técnica de AGs para o desenvolvimento de uma ferramenta computacional que realizasse o alinhamento de pares de seqüências biológicas.

Para validar o desempenho da ferramenta, os resultados obtidos foram comparados com aqueles produzidos pela ferramenta *bl2seq* (disponível no pacote de ferramentas do Blast), utilizando bases de dados de domínio

público. Isso permitiu a validação dos resultados obtidos pelo AG em relação àqueles obtidos pelo método de programação dinâmica encontrado na ferramenta *bl2seq*.

1.4. Organização da Dissertação

A dissertação está organizada da seguinte maneira: o Capítulo 2 apresenta uma breve descrição da área de Bioinformática com ênfase às ferramentas de alinhamentos de seqüências biológicas. O Capítulo 3 apresenta os principais aspectos da Computação Evolutiva, principalmente dos AGs, técnica utilizada para a criação da ferramenta deste trabalho. O Capítulo 4 descreve a metodologia utilizada no desenvolvimento da ferramenta para alinhamento de pares de seqüências com o uso de Ags. No Capítulo 5 são apresentados os resultados experimentais da ferramenta, com diversos casos de teste. Também são apresentadas comparações com a ferramenta *bl2seq* do pacote de ferramentas *Blast* para checar a desempenho do AG proposto. O Capítulo 6 apresenta as conclusões acerca do trabalho, sugerindo perspectivas futuras para a pesquisa.

A principal contribuição do trabalho desenvolvido está na área de pesquisas em Bioinformática, com o emprego de uma nova metodologia, que posteriormente possa ser utilizada como mais uma opção para análises de seqüências em projetos genomas e para refinamento dos resultados obtidos por outras ferramentas.

2. BIOINFORMÁTICA

Os esforços direcionados para o seqüenciamento do genoma humano geraram projetos de seqüenciamento de genoma de outros organismos menos complexos, muitos dos quais já foram finalizados [1]. Estes projetos trouxeram como conseqüência um grande acúmulo de dados, dificultando sua análise e interpretação.

Esta quantidade crescente de dados biológicos tem levado a necessidade de utilização de técnicas computacionais de apoio à análise e interpretação dos dados gerados. A necessidade de desenvolvimento de tais ferramentas contribuiu para o surgimento de uma nova área de pesquisa, a Bioinformática.

A Bioinformática, diz respeito à utilização de técnicas e ferramentas da computação para a resolução de problemas da biologia [2]. Uma área importante da biologia que necessita da aplicação da computação é a biologia molecular [3]. A Bioinformática engloba a resolução de problemas em áreas que vão desde a comparação de seqüências e montagem de fragmentos até a identificação e análise das etiquetas de expressão gênica (EST) e determinação da estrutura de proteínas.

A aplicação da tecnologia de etiquetas de expressão gênica (EST) foi demonstrada ser uma ferramenta eficiente para descoberta de genes, mapeamento gênico e geração de perfis de expressão [10].

Os projetos EST envolvem, inicialmente, a preparação de bibliotecas de cDNA, isolamento e seqüenciamento de clones, análise dos dados

gerados e, finalmente, a submissão destes dados ao GeneBank [11], o repositório mundial de seqüências de organismos.

Uma grande vantagem de projetos EST é o acesso mais fácil às informações de espécies que possuem um genoma complexo. Tal acesso é mais difícil se forem utilizadas técnicas convencionais da Genética, como foi o caso do Projeto Genoma da Cana-de-açúcar [12].

Atualmente, existe um grande número de ferramentas de Bioinformática que têm sido desenvolvidas para atender fases específicas de processos envolvidos em projetos EST. A criação de um Ambiente de Bioinformática, onde ocorra a integração de todas essas ferramentas facilita o acesso a elas. Dentre estas ferramentas, diversas têm sido propostas para alinhamento de seqüências.

2.1. Alinhamentos de Seqüências

A comparação de seqüências de genomas de organismos é uma das aplicações computacionais mais utilizadas por biólogos moleculares [11]. A descoberta de homologia entre seqüências de uma proteína ou famílias de proteínas freqüentemente traz as primeiras pistas a respeito da função de um novo gene seqüenciado [4].

O alinhamento de seqüências biológicas é uma operação básica em Bioinformática, por servir de suporte para outros processos como, por exemplo, a determinação da estrutura tridimensional das proteínas e análises filogenéticas [5]

De acordo com [3], a definição de alinhamento de seqüências pode ser dada como segue: dado um alfabeto finito *A*, sejam *s* e *t* duas cadeias sobre A de comprimento *m* e *n,* respectivamente. Seja *A** = *A* È {-} um alfabeto finito onde "-" representa o caracter espaço (*espaço*). A tupla (*s**, *t**) representa um alinhamento de *s* e *t* se as seguintes propriedades são satisfeitas:

- As cadeias *s** e *t** possuem o mesmo comprimento, $|s^*|=|t^*|$;
- s_i^* e $t_i^* \neq -, \forall i, 0 \leq i \leq |s^*|$. Ou seja, não existem dois espaços na mesma posição em s^* e t^*;
- Eliminando os espaços, s^* é reduzida para s;
- Eliminando os espaços, t^* é reduzida para t.

Para as seqüências de DNA, o alfabeto é *A** = {A,G,C,T,-}, dado que existem apenas 4 nucleotídeos além do caracter de espaço. O alfabeto A* para proteínas tem 21 elementos (incluindo o espaço).

Em relação à região analisada, o alinhamento pode ser definido em 2 tipos distintos: alinhamento local e alinhamento global (figura 1).

No alinhamento local procura-se alinhar apenas as regiões mais conservadas entre seqüências, independente da localização relativa de cada região em sua seqüência. Dessa forma, o alinhamento resulta uma ou mais regiões conservadas entre as seqüências. Já no alinhamento global, as seqüências devem ser alinhadas de um extremo ao outro, dando origem a apenas um resultado [13].

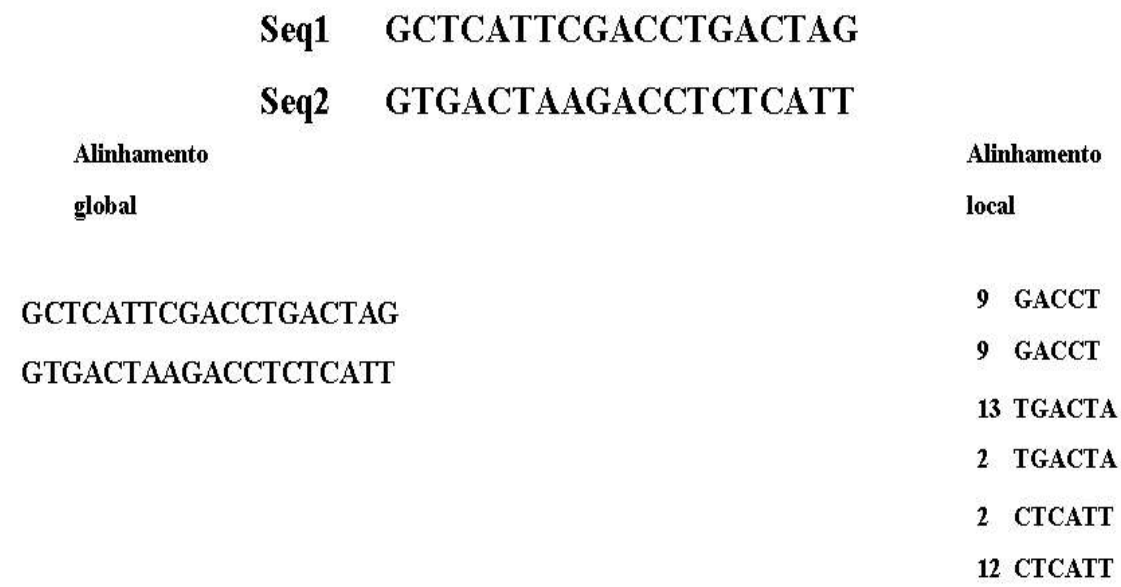


Figura 1: Exemplo de alinhamento local e global.

Tradicionalmente, o problema de alinhamento de seqüências é formulado como um problema de otimização com objetivo simples, em que, para cada par de seqüências, é procurado o alinhamento de maior semelhança, conforme um esquema de pontuação.

A pontuação de seqüências pode ser atribuída utilizando um método em que cada coluna do alinhamento recebe uma nota ou pontuação. O somatório destas notas contabiliza a pontuação total do alinhamento. A seguir, um exemplo de política de pontuação é apresentado: se uma coluna apresentar duas bases iguais, recebe pontuação *1*. Se apresentar bases diferentes, recebe a pontuação *– 1*. Finalmente, se essa coluna tiver uma base e um espaço, ela recebe a pontuação *– 2 (*figura *2)*.

Políticas mais complexas de pontuação de alinhamento de seqüências de nucleotídeos e aminoácidos têm sido desenvolvidas, como matrizes de substituição PAM e BLOSUM [14] .

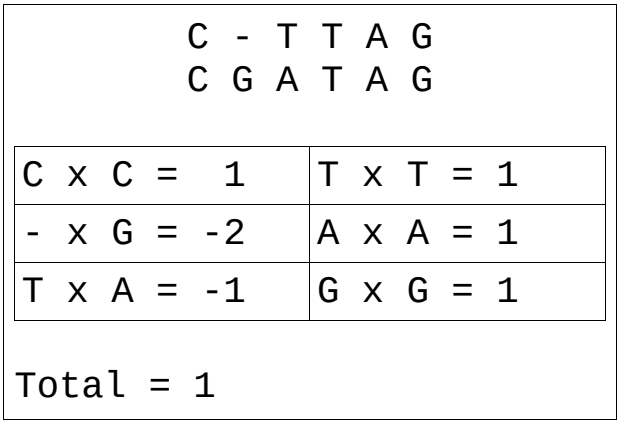


Figura 2: Exemplo de pontuação de um alinhamento. Nesse caso, o valor do alinhamento seria 1.

O alinhamento ótimo será aquele que tiver pontuação mais alta, que é definida pela similaridade entre duas seqüências [3]. A seguir, serão apresentadas algumas das técnicas utilizadas para alinhamentos de seqüências.

2.2. *Dot Plot*

Uma das primeiras técnicas utilizadas para comparação de seqüências, popularizada por Maizel e Lenk [15], foi a técnica de "plotagem de pontos" (Dot Plot). Esta técnica permite visualizar os possíveis alinhamentos que ocorrem entre duas seqüências.

Quando a técnica Dot Plot é utilizada, as seqüências são comparadas através de uma matriz. Em cada região onde as duas seqüências pareiam, um ponto é colocado. Assim, uma diagonal de pontos indica onde as seqüências são similares (figura 3).

A desvantagem desse tipo de comparação é o elevado número de janelas quando as seqüências são longas. Por exemplo, se uma seqüência

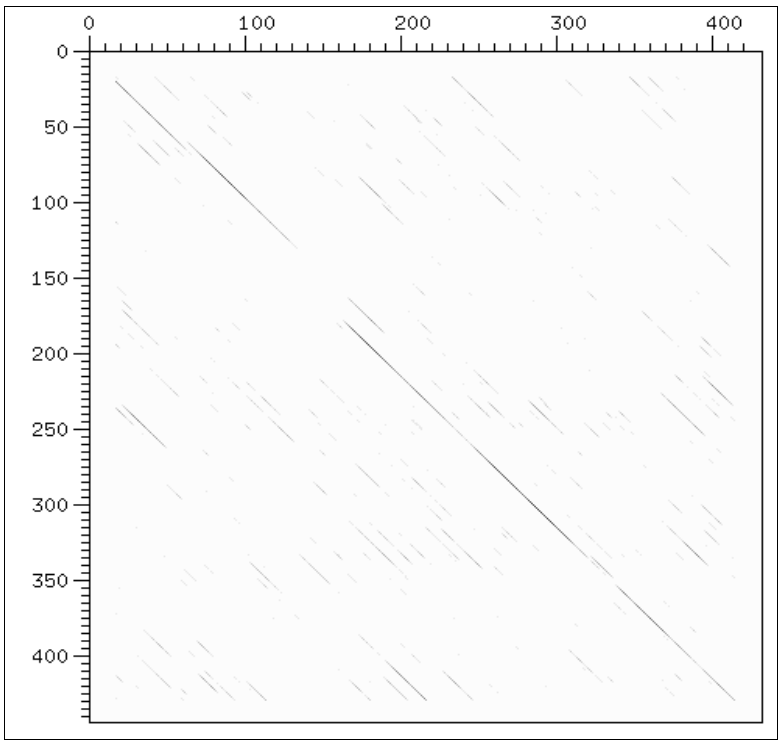


Figura 3: Dot Plot entre duas seqüências de DNA: a cadeia alpha da hemoglobina humana está no eixo horizontal e a cadeia beta está no eixo vertical.

apresentar tamanho N e a outra tamanho M, o número de janelas será igual a N x M, o que torna o processamento computacional pesado e lento.

2.3. O algoritmo de Needleman e Wunsch

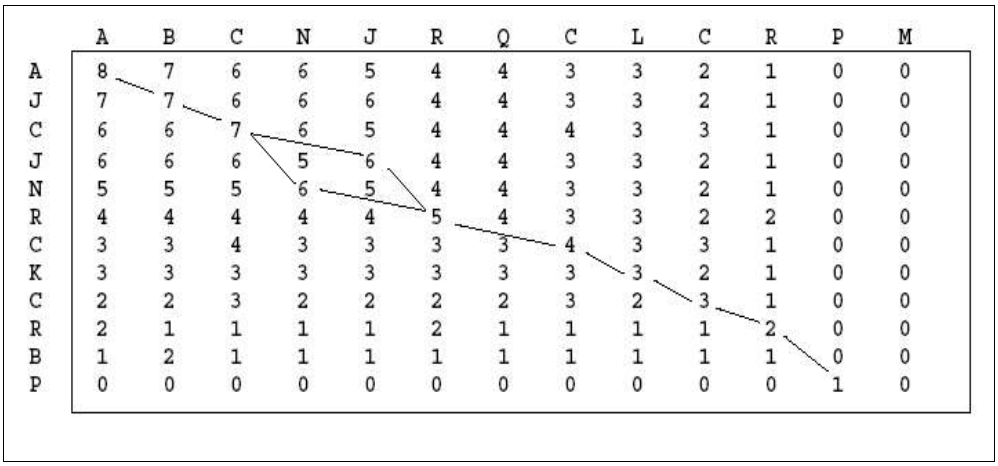
O algoritmo mais básico para alinhamento de seqüências foi desenvolvido por Needleman e Wunsch [16]. Seus estágios iniciais se assemelham ao Dot Plot, pois colocam as seqüências a serem comparadas à margem da matriz. Nesta matriz, a cada pareamento correto (*match*) é associado o valor *1* e a cada erro (*mismatch*) é associado o valor *0* (figura 4).

Quando o processo é finalizado, ao invés de seguir por uma única

diagonal para checar as semelhanças, os autores propuseram a modificação da diagonal central, inserindo buracos (*espaços*), para permitir a obtenção do melhor alinhamento (ou rota pelas diagonais da matriz).

O método proposto consiste em pontuar todos os alinhamentos. Quando todos os alinhamentos estiverem pontuados, é escolhido aquele que tem a maior pontuação.

Esse algoritmo leva em conta a similaridade das seqüências como um todo, maximizando o grau de similaridade ou alinhamento global. A desvantagem nesse tipo de comparação é que podem ser perdidas seqüências curtas e de alta similaridade.



	A	B	C	N	J	R	Q	C	L	C	R	P	M
A	8	7	6	6	5	4	4	3	3	2	1	0	0
J	7	7	6	6	6	4	4	3	3	2	1	0	0
C	6	6	7	6	5	4	4	4	3	3	1	0	0
J	6	6	6	5	6	4	4	3	3	2	1	0	0
N	5	5	5	6	5	4	4	3	3	2	1	0	0
R	4	4	4	4	4	5	4	3	3	2	2	0	0
C	3	3	4	3	3	3	3	4	3	3	1	0	0
K	3	3	3	3	3	3	3	3	3	2	1	0	0
C	2	2	3	2	2	2	2	3	2	3	1	0	0
R	2	1	1	1	1	2	1	1	1	1	2	0	0
B	1	2	1	1	1	1	1	1	1	1	1	0	0
P	0	0	0	0	0	0	0	0	0	0	0	1	0

Figura 4: Matriz de alinhamento baseada no algoritmo de Needleman e Wunsch. Nesse caso em particular, espaços e pareamentos incorretos têm a mesma penalidade.

2.4. Algoritmo de Smith-Waterman

Conceitos similares aos algoritmos de Needleman-Wunsch foram utilizados no algoritmo de similaridade local de Smith e Waterman [17].

Um alinhamento local permite mostrar padrões conservados entre

duas seqüências de nucleotídeos, sendo mais apropriado que um alinhamento global por evitar a perda de regiões conservadas.

Entre as características principais desse algoritmo está a determinação da subseqüência mais longa e que apresenta o maior grau de similaridade entre duas seqüências.

Para que isso seja realizado, poucas mudanças no algoritmo de Needleman e Wunsch são feitas, como:

- um sistema negativo de escore é associado a erros;
- espaços são penalizados;
- o começo e o final de um caminho ótimo podem ser encontrados em qualquer lugar da matriz.

A seguir serão descritas as matrizes de substituição PAM e BLOSUM. Tais matrizes são utilizadas comumente para pontuar alinhamentos entre seqüências.

2.5. Matrizes de substituição PAM e BLOSUM

Uma matriz de substituição é uma tabela que descreve a probabilidade de um par de resíduos (aminoácidos ou nucleotídeos) ocorrer em um alinhamento. Os valores associados aos elementos em uma matriz de substituição são logaritmos das razões de duas probabilidades. Uma das probabilidades é a de ocorrência aleatória de um aminoácido em um alinhamento de seqüências. Este valor é simplesmente o produto das freqüências independentes de cada aminoácido. A outra é a

probabilidade de ocorrência significativa de um par de resíduos em um alinhamento de seqüências. Estas probabilidades são derivadas de alinhamentos de seqüências reais que são sabidamente válidas [11].

Para se pontuar um alinhamento, o programa de alinhamento precisa saber se é mais provável que um dado par de aminoácidos tenha ocorrido ao acaso ou tenha ocorrido como resultado de um evento evolutivo. O logaritmo da razão da probabilidade de ocorrência significativa pela probabilidade de ocorrência ao acaso é positivo se a probabilidade de ocorrência significativa for maior, e negativo se a probabilidade de ocorrência ao acaso for maior. Por serem logaritmos de razões de probabilidade, as pontuações podem ser significativamente somadas para dar uma pontuação para toda a seqüência. Quanto mais positiva a pontuação, maior a probabilidade que o alinhamento seja significativo.

As matrizes de substituição para bases do DNA ou RNA são muito simples. Por via de regra, o programa de alinhamento de seqüências *BLAST* usa o esquema de associar uma recompensa padrão a um pareamento e uma penalidade padrão a um não-pareamento, sem considerar as freqüências totais das bases.

Para alinhamento de proteínas, o método de pontuação simples aplicado ao DNA não é suficiente. Matrizes de substituição para aminoácidos são complicadas, pois elas refletem a natureza química e a freqüência de ocorrência dos aminoácidos. Os aminoácidos que compõem as proteínas possuem propriedades bioquímicas que determinam como eles são substituídos durante a evolução.

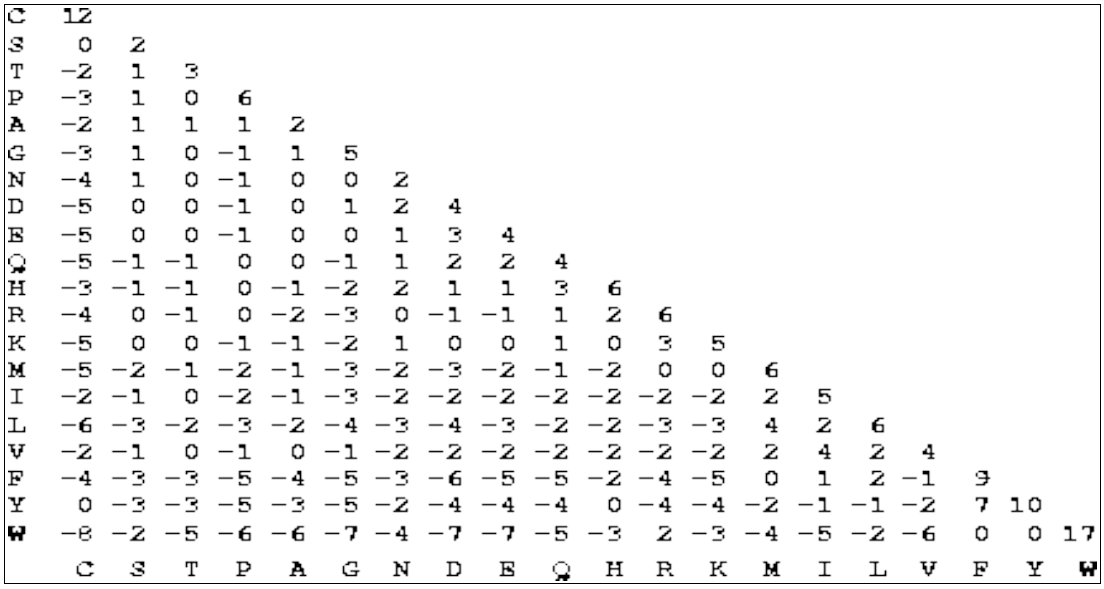
Dado que a comparação de proteínas é feita freqüentemente com critérios evolutivos, é necessário um esquema de pontuação que leve em conta estes critérios. Isto é realizado associando, na pontuação, diferentes probabilidades a diferentes substituições [3].

Uma forma de pontuação muito utilizada atualmente envolve as chamadas matrizes de substituição, as quais pertencem a duas famílias: PAM e BLOSUM.

As matrizes de substituição da família PAM (*Point Accepted Mutation*) foram construídas por meio de alinhamentos de um amplo conjunto de proteínas intimamente relacionadas, que sofreram uma certa divergência evolutiva (figura 5).

Diz-se que duas seqüências *s* e *t* divergem em uma unidade PAM, se a série de mutações aceitas para converter *s* em *t* estão na ordem de uma mutação para cada 100 aminoácidos (1%). Uma mutação diz-se aceita quando não se altera o funcionamento da proteína ou quando a mudança na proteína é benéfica para o organismo.

Schwartz e Dayhoff descobriram que 250 unidades PAM (PAM250) era um tempo apropriado para estudar relacionamentos evolutivos entre proteínas que fossem muito distantes [18].



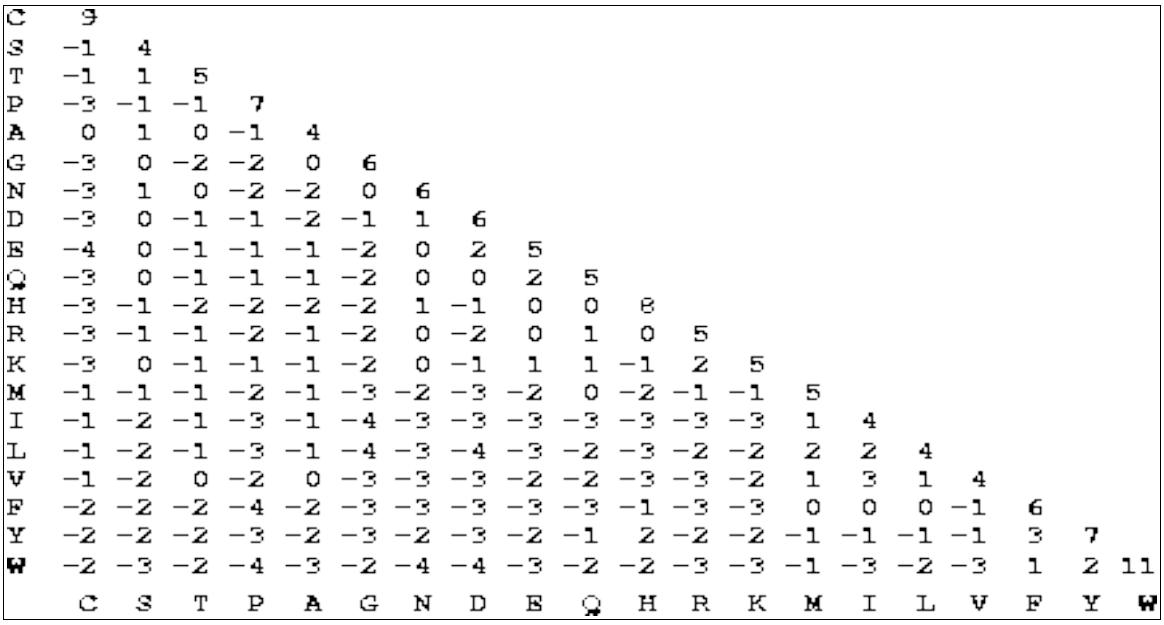
	C	S	T	P	A	G	N	D	E	Q	H	R	K	M	I	L	V	F	Y	W
C	12																			
S	0	2																		
T	-2	1	3																	
P	-3	1	0	6																
A	-2	1	1	1	2															
G	-3	1	0	-1	1	5														
N	-4	1	0	-1	0	0	2													
D	-5	0	0	-1	0	1	2	4												
E	-5	0	0	-1	0	0	1	3	4											
Q	-5	-1	-1	0	0	-1	1	2	2	4										
H	-3	-1	-1	0	-1	-2	2	1	1	3	6									
R	-4	0	-1	0	-2	-3	0	-1	-1	1	2	6								
K	-5	0	0	-1	-1	-2	1	0	0	1	0	3	5							
M	-5	-2	-1	-2	-1	-3	-2	-3	-2	-1	-2	0	0	6						
I	-2	-1	0	-2	-1	-3	-2	-2	-2	-2	-2	-2	-2	2	5					
L	-6	-3	-2	-3	-2	-4	-3	-4	-3	-2	-2	-3	-3	4	2	6				
V	-2	-1	0	-1	0	-1	-2	-2	-2	-2	-2	-2	-2	2	4	2	4			
F	-4	-3	-3	-5	-4	-5	-3	-6	-5	-5	-2	-4	-5	0	1	2	-1	9		
Y	0	-3	-3	-5	-3	-5	-2	-4	-4	-4	0	-4	-4	-2	-1	-1	-2	7	10	
W	-8	-2	-5	-6	-6	-7	-4	-7	-7	-5	-3	2	-3	-4	-5	-2	-6	0	0	17

Figura 5: Matriz de substituição PAM 250.

Um método alternativo de cálculo resultou nas matrizes de substituição da família BLOSUM (*Blocks Substitution Matrices*). Nesse método, foram considerados blocos, ou regiões altamente conservadas, no alinhamento das seqüências proteicas (figura 6).

As matrizes BLOSUM têm apresentado desempenho superior às matrizes PAM nos casos em que as seqüências são moderadamente divergentes. Matrizes BLOSUM são derivadas da base de dados Blocks, um conjunto de alinhamentos sem espaços de regiões de seqüências de famílias proteicas relacionadas. Um algoritmo de agrupamento (clustering) arranja as seqüências em cada bloco em grupos proximamente relacionados. As freqüências de substituições entre estes grupos dentro de uma família deriva a probabilidade de uma substituição significativa.

O valor numérico associado com cada elemento de uma matriz BLOSUM representa o valor de corte para o passo agregador. Assim, uma matriz BLOSUM62 indica que as seqüências foram colocadas no mesmo



	C	S	T	P	A	G	N	D	E	Q	H	R	K	M	I	L	V	F	Y	W
C	9																			
S	-1	4																		
T	-1	1	5																	
P	-3	-1	-1	7																
A	0	1	0	-1	4															
G	-3	0	-2	-2	0	6														
N	-3	1	0	-2	-2	0	6													
D	-3	0	-1	-1	-2	-1	1	6												
E	-4	0	-1	-1	-1	-2	0	2	5											
Q	-3	0	-1	-1	-1	-2	0	0	2	5										
H	-3	-1	-2	-2	-2	-2	1	-1	0	0	8									
R	-3	-1	-1	-2	-1	-2	0	-2	0	1	0	5								
K	-3	0	-1	-1	-1	-2	0	-1	1	1	-1	2	5							
M	-1	-1	-1	-2	-1	-3	-2	-3	-2	0	-2	-1	-1	5						
I	-1	-2	-1	-3	-1	-4	-3	-3	-3	-3	-3	-3	-3	1	4					
L	-1	-2	-1	-3	-1	-4	-3	-4	-3	-2	-3	-2	-2	2	2	4				
V	-1	-2	0	-2	0	-3	-3	-3	-2	-2	-3	-3	-2	1	3	1	4			
F	-2	-2	-2	-4	-2	-3	-3	-3	-3	-3	-1	-3	-3	0	0	0	-1	6		
Y	-2	-2	-2	-3	-2	-3	-2	-3	-2	-1	2	-2	-2	-1	-1	-1	-1	3	7	
W	-2	-3	-2	-4	-3	-2	-4	-4	-3	-2	-2	-3	-3	-1	-3	-2	-3	1	2	11

Figura 6: Matriz de substituição BLOSUM62.

bloco somente se elas fossem mais que 62% idênticas. Ao permitir que seqüências mais diversas sejam incluídas em cada agrupamento, valores de corte mais baixos representam escalas de tempo evolutivas maiores. Assim, matrizes com valores de corte baixos são apropriadas para se buscar relações mais distantes.

Dependendo do tamanho das seqüências que estão sendo comparadas, matrizes de um dado tipo são mais apropriadas.

As seqüências de DNA não mudam apenas por mutação pontual. Mudam também por inserção e deleção de resíduos. Conseqüentemente, muitas vezes é necessário introduzir espaços em uma ou ambas as seqüências que estão sendo alinhadas para produzir um alinhamento significativo entre elas. A maioria dos algoritmos usa uma penalidade por espaços para representar a validade de adicionar um espaço em um alinhamento.

2.6. FASTA e BLAST

O primeiros programa de busca por similaridades em bancos de dados largamente utilizado foi o *FASTA* [13].

O princípio básico do programa *FASTA* é comparar cada seqüência do banco de dados com a seqüência procurada e reportar aquelas que tenham semelhança significativa [3]. Contudo, o *FASTA* possui alguns problemas relacionados à perda de regiões da seqüência analisada.

O algoritmo *FASTA* busca, primeiramente, seqüências curtas (chamadas de k-tuplas), que ocorrem tanto na seqüência de busca quanto na base de dados de seqüências. Usando então a matriz *BLOSUM50*, o algoritmo pontua os 10 alinhamentos sem espaços que contêm os k-tuplas mais idênticas. Estes alinhamentos sem espaços são testados por sua habilidade de serem fundidos a um alinhamento com espaços sem reduzir a pontuação para abaixo de um dado limiar. Para aqueles alinhamentos fundidos que pontuam acima do limiar, um alinhamento local ótimo daquela região é então computado e a pontuação para aquele alinhamento (chamada de pontuação otimizada) é relatada [11].

Exemplos de programas *FASTA* atualmente disponíveis:

- [fasta] Compara uma seqüência protéica contra uma base de dados de proteínas (ou uma seqüência de DNA contra uma base de dados de DNA) usando o algoritmo *FASTA*.;
- [ssearch] Compara uma seqüência protéica contra uma base de dados de proteínas (ou uma seqüência de DNA contra uma base

de dados de DNA) usando o algortimo Smith-Waterman;

- [align] Computa o alinhamento global entre duas seqüências de DNA ou proteína;
- [lalign] Computa o alinhamento local entre duas seqüências de DNA ou proteína.

Um algoritmo melhorado e muito utilizado atualmente pertence ao grupo de programas *BLAST (Basic Local Alignment Search Tool)* [4], que possuem um rigoroso método para obter as probabilidades de pareamento. O *BLAST* é uma ferramenta de busca rápida que procura por semelhanças entre uma seqüência qualquer e as seqüências já armazenadas em um banco de dados.

O princípio básico da família de programas *BLAST* é que os melhores alinhamentos provavelmente incluem segmentos pareados de alta pontuação (*High-scoring Segment Pairs*). Um segmento é uma sub-cadeia de uma seqüência. Dadas duas seqüências, um segmento pareado entre elas é um par de segmentos do mesmo comprimento, um de cada seqüência.

O algoritmo *BLAST* tem três passos básicos. Primeiro, ele cria uma lista de todas as seqüências curtas (chamadas de palavras (*words*) na terminologia empregada pelo *BLAST*) que pontuam acima de um dado valor de limiar quando alinhados com a seqüência de busca. A seguir, a base de dados é vasculhada para ocorrências destas palavras. Como o comprimento das palavras é curto (3 resíduos para proteínas, 11 resíduos para ácidos nucléicos), é possível vasculhar uma tabela pré-computada de

todas as palavras e suas posições nas seqüências para melhorar a velocidade de processamento. Estas palavras pareadas são então estendidas em alinhamentos locais sem espaços entre a seqüência de busca e a seqüência da bases de dados. Extensões são continuadas até que a pontuação do alinhamento caia abaixo de um dado valor de limiar. Os alinhamentos com as melhores pontuações são combinados, nas posições que for possível em alinhamentos locais. Originalmente, o BLAST procurava somente alinhamentos sem espaços. No entanto, já existem novas adições ao pacote BLAST que realizam buscas por alinhamentos com espaços [11]

3. COMPUTAÇÃO EVOLUTIVA

3.1. Teoria da Evolução

Até meados do século XIX, a Teoria do Criacionismo ou Fixismo era aceita no mundo científico. Nessa teoria, todas as espécies seriam criadas a partir de um plano divino e nunca sofreriam qualquer alteração.

Muitos cientistas tentavam discutir a possibilidade das espécies evoluírem para outras espécies no decorrer de gerações. O naturalista francês Jean-Baptiste Lamarck ficou famoso em sua época por formular sua teoria evolutiva no periódico *Philosophie Zoologique,* em 1809 [19].

A idéia de Lamarck, também chamada de transformismo, assumia que as linhagens de espécies persistiam indefinidamente, mudando de uma forma para outra. Nessa teoria, não há extinção, nem ramificações entre linhagens.

O mecanismo que Lamarck sugeriu era o da herança de caracteres adquiridos e “força interior”. As mudanças de geração para geração seriam acumuladas por desejo do organismo em evoluir. Com tal acúmulo no decorrer do tempo, a espécie poderia se transformar o bastante para se tornar uma nova [19].

Uma revolução no pensamento científico teve início em 1858. Charles Robert Darwin e Alfred Russel Wallace demonstraram suas evidências para a teoria da evolução das espécies. Ambos defendiam, independentemente, o princípio da Seleção Natural, mas como Darwin já

havia escrito "*On the Origin of Species"*, apenas o seu trabalho foi considerado [19].

A teoria de Darwin diferia em vários aspectos em relação do transformismo de Lamarck. Uma das principais diferenças seria a da ancestralidade comum e extinção que poderia ocorrer nas espécies. Além disso, por meio da Seleção Natural, Darwin tentou descrever o processo responsável pela ocorrência da Evolução.

Apesar de sua teoria acerca da evolução ter sido aceita, o mecanismo de Seleção Natural foi rejeitado. Darwin não conseguiu criar uma teoria satisfatória de hereditariedade que embasasse tal mecanismo [19].

Entretanto, quando a teoria de hereditariedade de Mendel foi redescoberta em 1920, a teoria de evolução de Charles Darwin foi reestudada. Fisher, Haldane e Wright demonstraram que a hereditariedade mendeliana e a Seleção Natural seriam compatíveis [19]. Surgia dessa forma a teoria sintética de evolução, ou neodarwinismo, que a partir de 1940 foi se espalhando e sendo aceito amplamente no mundo científico.

3.2. A Seleção Natural

Durante a reprodução dos organismos, a recombinação, permutação (*crossing-over*) e possíveis mutações, alteram o genoma do novo indivíduo. Isso traz como conseqüência toda a diversidade atualmente encontrada. Sobre essa diversidade, atua novamente o ambiente que, por

meio da Seleção Natural, seleciona os indivíduos mais aptos para se reproduzirem.

A Seleção Natural faz com que os traços hereditários correlacionados com o sucesso reprodutivo aumente de freqüência e sejam mantidos. A sobrevivência é apenas um meio de alcançar o sucesso reprodutivo [20].

3.3. A computação evolutiva

Entre 1950 e 1960, muitos cientistas de computação, independentemente, estudaram sistemas evolucionários com o objetivo de utilizar conceitos da evolução como uma ferramenta de otimização [21].

A otimização é a busca da melhor solução para um dado problema. Em termos matemáticos, esta consiste em encontrar a solução que corresponde ao ponto de máximo ou mínimo de uma função objetivo (também chamada de função de aptidão).

A idéia por trás dos sistemas baseados no processo evolutivo seria a de evoluir uma população de possíveis soluções para um determinado problema, usando operadores inspirados na biologia, como a Seleção Natural [21]. Dessa forma, o processo evolucionário pode ser modelado no computador e aplicado a problemas onde heurísticas não estão disponíveis, os resultados até então não são satisfatórios ou o espaço de busca requer uma busca exaustiva para encontrar a solução ótima [22].

Os primeiros trabalhos desenvolvidos com algoritmos inspirados na evolução e aprendizado de máquina para otimização foram os de Box

(1957), Friedman (1959), Bledsoe (1961), Bremermann (1962), Reed, Tombs e Baricelli (1967) [21].

Outros pesquisadores investigaram como utilizar a evolução para otimizar sistemas específicos. Fogel *et al.* Propuseram o uso da evolução para desenvolver Inteligência Artificial (IA) por meio de programação evolucionária [22]. Outra abordagem proposta foi o uso de estratégias evolucionárias para problemas de otimização [21]. Uma terceira abordagem fazia uso de AGs para simular evolução [22].

A mais recente área de pesquisa em computação evolucionária é a programação genética, que se distingue dos outros métodos por se focar na evolução de programas de computador.

Entretanto, apesar apesar das diferentes abordagens, todos os métodos de otimização evolucionariamente inspirados utilizam a mesma estrutura básica, descrita a seguir:

Representação da solução

Em todas as abordagens, a representação da solução (também chamada de indivíduo) é uma das chaves para que o método resulte em uma solução satisfatória. É necessario que se crie uma representação boa o suficiente para o problema afim de que os operadores responsáveis para a evolução da população de indivíduos alcance o maior espaço de busca possível.

Fonte de variação

Um fator fundamental para que a evolução ocorra em sistemas naturais é a quantidade de variação na população. Analogamente, em

sistemas computacionais inspirados na evolução são necessários operadores que promovam tal variabilidade. Tais operadores serão a fonte da nova variação em cada geração do processo evolucionário [22].

Seleção

Em todos os sistemas computacionais inspirados na evolução, a seleção será a responsável por promover a remoção das soluções menos apropriadas para o problema em questão. Aqueles indivíduos que sobreviverem servirão como pais dos indivíduos da próxima geração, produzindo sua prole com grande diferença em relação aos pais deles, dependendo da metodologia de seleção utilizada.

Função de Aptidão

A seleção requer um meio de associar um valor a cada uma das soluções para o problema e, para que isso ocorra, cria-se a chamada função de aptidão. Por meio dessa função, cada solução terá um valor que significará o quão adequada é a solução para o problema. Baseado nesse valor, a seleção irá escolher os indivíduos mais ou menos aptos para o problema.

A função de aptidão é considerada o passo mais desafiador em algortimos evolucionários. O valor de aptidão de cada solução deve ser um reflexo exato do problema, caso contrário o processo evolucionário não encontrará a solução satisfatória [22].

A seguir, será descrita com mais profundidade uma das abordagens da Computação Evolutiva, os Algoritmos Genéticos (AGs). Tal abordagem foi a utilizada na metodologia da ferramenta desenvolvida durante esta

pesquisa.

3.4. Algoritmos Genéticos

Como dito anteriormente, os *AGs* representam uma das abordagens de computação evolucionária onde, por meio de métodos adaptativos, pode-se solucionar problemas de otimização [9]. O método foi desenvolvido por John Holland, seus colegas e estudantes na Universidade de Michigan e popularizado por David Goldberg [9].

Na Biologia, a teoria da Evolução, formulada por Charles Darwin, diz que o meio ambiente seleciona, em cada geração, os indivíduos de uma espécie mais aptos para a reprodução. Dessa forma, os mais aptos são os que se reproduzem e levam adiante seus genes, fazendo com que os genes de indivíduos menos aptos sejam eliminados.

Um *AG* é uma metáfora computacional desses fenômenos e, por isso, utiliza muitos termos originados da Biologia. Os principais conceitos, seguindo as definições de Goldberg, são:

- Cromossomo: na Biologia, genoma é o conjunto completo de genes de um organismo. Esses genes estão localizados em cromossomos. Nos AGs, os cromossomos representam uma estrutura de dados que codifica uma solução para um problema, como, por exemplo, um número de vetores binários.
- Gene: na Biologia, o gene é a unidade da hereditariedade,

transmitida pelos cromossomos e que controla todas as características do organismo. Nos AGs, é um parâmetro codificado em um cromossomo.

- Indivíduo: um membro da população. Nos AGs, o indivíduo é representado pelo cromossomo e sua aptidão.
- Genótipo: o que na Biologia representa a composição genética contida no genoma, nos AGs representa a informação codificada no cromossomo.
- Fenótipo: na biologia representa a expressão ou manifestação física de um gene no organismo. Nos AGs, representa a informação reconstituída a partir dos dados do genótipo pelo valor de sua aptidão.

Os AGs são robustos para a busca de soluções em uma grande variedade de problemas, pois não impõem muitas das limitações encontradas nos métodos de busca tradicionais [9]. Uma das características mais importantes desses algoritmos é que possibilitam encontrar soluções ótimas ou quase ótimas para um problema sem necessidade de informação extra, como cálculo de derivadas de funções [9].

Nas próximas subseções deste capítulo, serão descritas as principais características dos *AGs.*

3.5. Operadores Genéticos

A forma mais simples de *AGs* envolvem 3 tipos de operadores: crossover, mutação e seleção [21].

- **Crossover**: é o operador responsável pelo cruzamento de informações entre indivíduos de uma população. Aleatoriamente o operador escolhe um ponto ao longo de dois cromossomos e faz a troca entre os trechos correspondentes, gerando dois novos filhos (Figura 7).

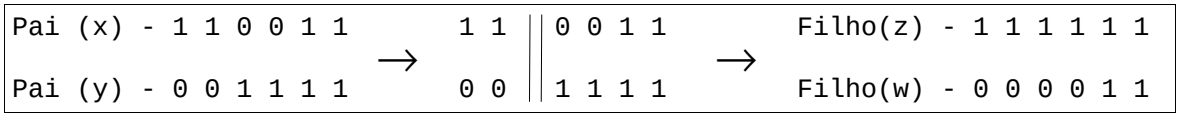


Figura 7: Funcionamento do operador de Crossover. Assim que os pais são selecionados, um ponto em seu vetor é escolhido aleatoriamente. Os trechos entre os dois cromossomos a partir desse ponto de crossover são trocados e assim novos filhos são gerados.

- **Mutação:** é o operador responsável pela variabilidade do AG. O seu funcionamento envolve escolher aleatoriamente genes no cromossomo e trocar o seu valor. Em um vetor binário isso significaria trocar um *0* por *1* e vice-e-versa (Figura 8).

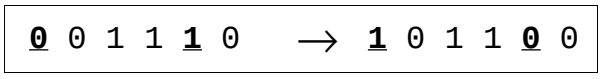


Figura 8: Funcionamento do operador de Mutação. Neste exemplo, o cromossomo sofreu duas mutações, no primeiro e quinto gene.

- **Seleção:** é o operador responsável por escolher os cromossomos na população. A seleção ocorre em dois momentos distintos: na escolha dos pais para gerar novos indivíduos e na escolha dos indivíduos que estarão presentes na próxima geração. Quanto maior o valor de aptidão do cromossomo, maiores as chances dele ser escolhido.

Existem vários métodos descritos para o operador de seleção. Os mais conhecidos são o método da roleta e o método do torneio [9].

O método da roleta é o mais utilizado. Nesse método os indivíduos de uma população são escolhidos para a próxima geração utilizando uma roleta, similar àquelas de jogos de azar. Nessa roleta, os indivíduos têm representada uma fatia proporcional ao seu valor de aptidão. Assim, quanto maior é a aptidão do indivíduo, maior seria sua faixa representada na roleta [9] (Figura 9).

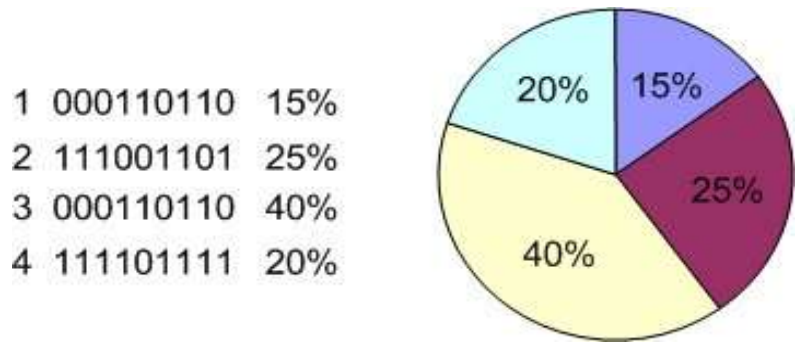


Figura 9: Método de Seleção por roleta. Os cromossomos que tiverem maior aptidão terão mais chances de serem escolhidos.

No método de torneio, os indivíduos são escohidos aleatoriamente, com a mesma probabilidade e farão parte de uma população intermediária. Então, ocorre uma disputa entre os indivíduos escolhidos. O indivíduo com maior aptidão, será aquele que permanecerá na geração seguinte, ou será o escohido para ser um dos pais.

Um outro mecanismo muito utilizado em *AGs* é o *Elitismo*, que força o *AG* a reter um número de melhores indivíduos em cada geração. Sem esse operador, os melhores indivíduos de uma população poderiam ser perdidos, caso não fossem selecionados ou sofressem a ação dos

operadores de crossover e mutação. Diversos estudos mostram que o desempenho do *AG* com o uso de elitismo é, em geral, melhorado [21].

3.6. Funcionamento

Tipicamente, a utilização de um AG para a solução de um problema de otimização tem como primeiro passo a geração de uma população inicial. Esta é geralmente formada por um conjunto de indivíduos de composição aleatória, que representam possíveis soluções para o problema a ser resolvido.

Essa população é avaliada durante o processo evolutivo: cada cromossomo recebe uma nota (aptidão) que reflete a qualidade desta solução. De acordo com sua aptidão, os cromossomos são escolhidos, tornando-se os progenitores da próxima geração de indivíduos. Operadores de crossover e mutação são aplicados sobre esses indivíduos alterando as características dos filhos, produzindo uma melhora ou piora nos seus valores de aptidão.

Em seguida, o operador de seleção escolhe os melhores indivíduos e forma a nova população, que geralmente mantém a quantidade de indivíduos iniciais. Para manter essa quantidade de indivíduos iniciais, pode-se utilizar um parâmetro no *AG* chamado de *taxa de substituição*. Esta taxa poderá fazer tanto a população original inteira ser substituída por uma nova quanto apenas uma certa quantidade de indivíduos (normalmente os piores) serem substituídos pelo mesmo número de novos

indivíduos gerados.

Essa iteração continua até que um critério de parada seja satisfeito como, por exemplo, número de gerações máximas (figura 10).

```
/* Seja P( t ) a população de cromossomos na geração "t"
        t ← 0
        inicializar P( t )
        avaliar P ( t )
        ENQUANTO o critério de parada não esteja satisfeito
FAÇA
        t ← t+1
        selecionar P( t ) a partir de P ( t –1 )
        aplicar permuta sobre P( t )
        aplicar mutação sobre P ( t )
        avaliar P( t )
        FIM ENQUANTO
FIM
```

Figura 10: Pseudo-código de um AG.

3.7. Exemplo do funcionamento de um AG

A seguir, uma demonstração do funcionamento de um AG, utilizada por Goldberg [9]. Considere um problema simples de maximização de uma função:

$f(x)=x^2$, onde x pode variar de 0 a 31.

Inicialmente, é necessária uma representação de cromossomo que seja satisfatória para representar o problema. Nesse caso, poder-se-ia utilizar a representação de um número decimal, na base binária. Dessa forma, os cromossomos poderiam variar desde o vetor de números

binários 00000 (0) até 11111 (31).

O valor de aptidão desse cromossomo, seria o valor de x aplicado a função f(x). Por exemplo, o cromossomo representado pelo vetor binário 01100, que representa o número decimal 12, teria 144 como seu valor de aptidão .

Com esta representação de cromossomo, uma população poderia ser formada com, por exemplo, 4 indivíduos gerados aleatoriamente. Suponha que a tabela 1 esteja mostrando os indivíduos que foram gerados, com respectivos valores de aptidão:

Id	População inicial	Valor de x	f(X)
A	0 1 1 0 1	13	169
B	1 1 0 0 0	24	576
C	0 1 0 0 0	8	64
D	1 0 0 1 1	19	361

Tabela 1: Uma possível população inicial.

O próximo passo seria selecionar os pais para se cruzarem. Por exemplo, utilizando seleção por torneio de 2 (ver Seção 3.5).

Suponha que os indívuos A e B foram selecionados como o primeiro casal de pais e os indivíduos C e D o segundo casal. O operador de crossover escolhe aleatoriamente um sítio em cada dupla e realiza a troca de trechos (tabela 2).

A partir do indivíduo com aptidão 27 na primeira geração de novos indivíduos, poderia rapidamente se chegar a um valor de x desejado (que nesse caso seria 31). Para isso, bastaria este indivíduo sofrer mutação.

Pais selecionados e ponto de crossover escolhido		Filhos gerados após crossover	Valor de x	f(x)
0 1 1 0 | 1 1 1 0 0 | 0	→	0 1 1 0 0 1 1 0 0 1	12 25	144 625
1 1 | 0 0 0 1 0 | 0 1 1	→	1 1 0 1 1 1 0 0 0 0	27 16	729 256

Tabela 2: Demonstração do crossover ocorrendo entre os pais.

1 1 0 1 1 → 1 1 1 1 1

3.8. AGs para alinhamento de seqüências

Durante os últimos anos, têm-e observado uma grande expansão na utilização de Algoritmos Evolutivos, especialmente Algoritmos Genéticos (AGs), para problemas de otimização em Biologia Molecular como, por exemplo, alinhamentos [5],[6] e multialinhamentos [7],[8] de seqüências biológicas.

Neste trabalho, utilizou-se a técnica de *AGs* para o desenvolvimento de uma ferramenta computacional que realizasse o alinhamento de pares de seqüências biológicas de nucleotídeos e aminoácidos. A metodologia empregada, será descrita a seguir.

4. METODOLOGIA

Esta seção descreve a metodologia que emprega *AGs* para alinhamentos de pares de seqüências. Utilizou-se como modelo básico o *AG* Simples de Goldberg [9]. A partir dele, foram realizadas alterações para que o AG fosse ajustado ao problema de alinhamentos de pares de seqüências, baseando-se no trabalho de Wayama [6].

4.1. Linguagem de programação e ambiente utilizados

Para o desenvolvimento do projeto proposto, foi utilizado um computador com processador AMD Duron 1.2GHz, com 256MB de memória. Implementou-se a metodologia de alinhamento de seqüências utilizando-se a linguagem de programação *Object Pascal*, com o ambiente de desenvolvimento *Kylix*, sob a plataforma *Linux.*

Para a realização dos testes utilizou-se tanto o computador de desenvolvimento da ferramenta, como o *cluster* do Laboratório de Inteligência Computacional (*LABIC*), do Instituto de Ciências Matemáticas e de Computação (*ICMC*), da Universidade de São Paulo.

4.2. Codificação do Cromossomo

Cada cromossomo representa um alinhamento entre duas seqüências. São utilizados vetores de valores binários na representação de cada cromossomo. O valor *1* está relacionado a um resíduo da seqüência,

enquanto que o valor *0* determina um espaço (*gap*) da seqüência. O número de espaços a ser incluído no alinhamento será o tamanho das duas seqüências originais de resíduos somadas. Desta forma, o comprimento do cromossomo depende do comprimento das seqüências de entrada.

Sejam duas seqüências *S1=ACTGGTACTA* (comprimento 10) e *S2=ATCGCTG* (comprimento 7). O cromossomo é formado por um vetor binário composto por duas metades, cada qual com comprimento igual a soma dos comprimentos das seqüências de entrada. Cada seqüência é armazenada em uma das metades do cromossomo. Desta forma na primeira metade há 17 posições, sendo que destas apenas 10 contêm resíduos. As demais são espaços. Na segunda metade, analogamente, são colocados 7 resíduos e 10 espaços. Um possível cromossomo pode ser visto na figura 11. O alinhamento correspondente pode ser visto na figura 12:

```
1 0 1 0 1 0 0 1 1 0 1 1 1 1 0 0 1
1 0 0 0 1 1 1 0 1 0 0 1 1 0 0 0 0
```

Figura 11: Exemplo de cromossomo.

```
1 0 1 0 1 0 0 1 1 0 1 1 1 1 0 0 1        A - C - T - - G G - T A C T - - A
1 0 0 0 1 1 1 0 1 0 0 1 1 0 0 0 0   ⇒    A - - - T C G - C - - T G - - - -
```

Figura 12: Exemplo de um cromossomo gerado pela metodologia e sua respectiva representação em termos biológicos.

Na primeira metade do cromossomo, o primeiro 1 representa o primeiro resíduo no alinhamento. O segundo 1 representa o segundo

resíduo e assim por diante. Os valores *0* são substituídos por espaços (símbolo '-'). Com esse tipo de codificação, é possível representar todos os alinhamentos que poderiam ocorrer entre essas duas seqüências.

4.3. Função de Aptidão

O valor de aptidão de cada cromossomo é avaliado segundo dois critérios: a partir de matrizes de substituição PAM ou BLOSUM, escolhidas arbitrariamente, e da quantidade de pareamentos corretos (*hits*) entre as seqüências.

O critério de pareamento correto foi levado em consideração na função devido ao fato de que alinhamentos diferentes podem ter a mesma pontuação por matrizes de substituição. Com a adição desse critério, o valor de aptidão dos cromossomos torna-se melhor definido, facilitando a busca das melhores soluções, tanto pelo número de pareamentos corretos quanto pelo valor obtido nas matrizes.

Sejam s_1' e s_2' duas seqüências a serem alinhadas. A representação do cromossomo exige que os espaços sejam incluídos no alinhamento, conforme já foi explicado anteriormente. A representação destas seqüencias com espaços seria s_1 e s_2, onde ambas possuem o mesmo comprimento de valor T.

Para que a função de aptidão seja apresentada, é necessário definir algumas funções auxiliares:

- $\mathrm{Hit}(i)=\begin{cases}0, s_{1i}\neq s_{2i}\\ 1, s_{1i}=s_{2i}\end{cases}$,

onde s_{1i} = resíduo da i-ésima posição da seqüência s_1;

- $f_h=\sum_{i=0}^{T}\mathrm{Hit}(i)$
- $\mathrm{MatSubst}(r_{1,}r_2)$ = valor na matriz de substituição entre os resíduos r_1 e r_2
- $f_m=\sum_{i=0}^{T}\mathrm{MatSubst}(s_{1i},s_{2i})$

O valor de aptidão de um alinhamento *j* é definido como:

$$F_j=(F_{mj},F_{hj})$$

A relação de ordem é definida da seguinte forma. Sejam F_1 e F_2 os valores de fitness de 2 diferentes alinhamentos, dessa forma:

$F_1=(F_{m1,}F_{h1})$ e $F_2=(F_{m2,}F_{h2})$

Assim $F_1>F_2$ se $\begin{cases}F_{m1}>F_{m2,}\\ \text{ou}\\ F_{m1}=F_{m2}\,e\,F_{h1}>F_{h2}\end{cases}$

Duas codificações foram utilizadas para a função de fitness, relacionados aos valores atribuídos aos espaços encontrados nas seqüências.

Na primeira delas, atribuiu-se um valor igual para espaços iniciais e de extensão (*gap simples*). Na segunda, penalizou-se de forma diferente os espaços, onde os espaços iniciais receberiam penalizações maiores que os espaços de extensão (*gap avançado*).

4.4. Operadores Genéticos

A seleção dos pais para reprodução é realizada pelo método de torneio [9]. Entre três cromossomos escolhidos aleatoriamente, o que tiver maior aptidão é selecionado para progenitor.

O operador de recombinação combina dois indivíduos, gerando novos descendentes. A implementação deste operador foi baseada no trabalho de Wayama [6]. Dois indivíduos são escolhidos e ocorre uma mistura de ambos (meio a meio) gerando uma nova prole, conforme mostrado na figura 13.

1 0 1 0 1 0 0 1 1 0 1 1 1 1 0 0 1 : 1 0 0 0 1 1 1 0 1 0 0 1 1 0 0 0 0

1 1 1 1 1 0 0 0 0 1 1 1 1 1 0 0 0 : 1 1 1 1 0 0 0 1 1 1 0 0 0 0 0 0 0

⇓ Recombinação ⇓

1 0 1 0 1 0 0 1 1 0 1 1 1 1 0 0 1 : 1 1 1 1 0 0 0 1 1 1 0 0 0 0 0 0 0

1 1 1 1 1 0 0 0 0 1 1 1 1 1 0 0 0 : 1 0 0 0 1 1 1 0 1 0 0 1 1 0 0 0 0

Figura 13: Funcionamento do operador de Crossover.

Para o operador de mutação, uma posição no cromossomo é escolhida ao acaso. A mutação age sobre esta posição trocando *0* por *1,* e vice-versa.

Como a quantidade de *0*s e *1*s deve ser preservada, uma segunda posição é escolhida e modificada. A segunda posição escolhida é a próxima posição do cromossomo cujo valor do gene é oposto ao valor

selecionado anteriormente (resíduo-espaço ou espaço-resíduo). A figura 14 ilustra essa operação. Sobre um trecho de cromossomo ocorrem duas mutações. Na primeira mutação, da esquerda para a direita, um espaço (segunda posição) é selecionado e transformado em resíduo. Como a quantidade de resíduos deve mantida em 10, um resíduo deve ser transformado em espaço. Assim, o próximo resíduo a direita desta posição também é alterado. O mesmo ocorre na segunda mutação, mas alterando resíduo para espaço.

Figura 14: Funcionamento do operador de Mutação

4.5. Critério de Parada

Dois tipos de critérios de parada foram implementados para uma comparação. Parada determinada por um máximo de gerações pré-definido e parada após um número de gerações sem mudança no melhor indivíduo.

4.6. Casos de teste

Foram preparados vários casos de teste, que se diferenciavam pelo tamanho e similaridade das seqüências, todos adquiridos da base de dados SWISS-PROT, como demonstrado na tabela 3 e 4.

As seqüências do tipo *short* são seqüências curtas que possuem por volta de 100 aminoácidos. As seqüências do tipo *medium* são seqüências de aminoácidos consideradas de tamanho intermediário para o trabalho e possuem por volta de 300 a 400 aminoácidos. Já as seqüências do tipo *long* são aquelas que possuem por volta de 500 a 900 aminoácidos.

As similaridades das seqüências dos casos de teste foram adquiridas por meio da base de dados *BAliBASE* (*Benchmark Alignment dataBASE*) [23]. Dessa forma, os casos de teste possuíam desde seqüências com no máximo 25% de similaridade entre si até seqüências com mais de 35% de similaridade.

Com esses dados, foi possível avaliar em quais casos a ferramenta desenvolvida teria um desempenho melhor, tanto em relação ao tamanho de seqüências, quanto em relação ao grau de similaridade.

Casos de Teste	
Similaridade	**Swiss-Prot Accession**
Short < 25%	P20857
	P10599
Short 20% a 40%	P00097
	P24469
Short > 35%	P00195
	P00198
Medium < 25%	P02906
	P37329
Medium 20% a 40%	Q27743
	P14295
Medium > 35%	P00784
	P14080
Long < 25%	P00924
	P08310
Long 20% a 40%	P38673
	P45888
Long > 35%	P00489
	P00490

Tabela 3: Casos de teste utilizados nos experimentos.

Casos de Teste – short < 25%	
Identificação do teste	**Swiss-Prot Accession**
Thi1	P20857
	P10599
Thi2	P00277
	P20857
Thi3	P20942
	P20857
Ubi1	P56408
	P62988
Ubi2	P62834
	P62988

Tabela 4: Segundo conjunto de casos de teste utilizado. Nesse caso com seqüências com menos que 25% de similaridade.

5. EXPERIMENTOS E RESULTADOS

Neste capítulo são apresentados os experimentos realizados e os resultados obtidos com a metodologia citada anteriormente.

5.1. Tamanho inicial da população

Vários tamanhos iniciais de população foram testados. Reparou-se que uma população de 1000 indivíduos gerava em um tempo satisfatório resultados adequados, mas apenas para certos casos de seqüências.

Se o número de indivíduos da população fosse aumentado, o tempo que o algoritmo levava para ser executado tornava a simulação impraticável. Por outro lado, se o tamanho da população fosse diminuído, a população não conseguia convergir para resultados apropriados, devido ao espaço de busca pequeno.

O principal problema em relação ao tamanho populacional foi que quanto maiores eram as seqüências a serem alinhadas, maior ficava o cromossomo que representaria o alinhamento e, conseqüentemente, o tamanho do espaço de busca. Dessa forma, uma população inicial pequena, para casos onde as seqüências eram longas, não geravam resultados satisfatórios, como será demonstrado adiante.Os cromossomos da população inicial foram gerados aleatoriamente.

5.2. Taxas de crossover, mutação e substituição

Determinar o valor ótimo das taxa de crossover, mutação e susbstituição nos *AGs* é um problema complexo. Nesse trabalho utilizou-se um conjunto de valores desses parâmetros de forma arbitrária. O critério utilizado nessa seleção foi avaliar alguns parâmetros desse conjunto e verificar o impacto das mudanças deles no desempenho do *AG*. O conjunto de valores de parâmetros que apresentou o melhor resultado dentre todos os casos verificados, foi o escolhido para esse trabalho.

Notou-se nesses experimentos que tanto as taxas de crossover e mutação não poderiam ser muito altas. Para determinar estas taxas, fixou-se um dos parâmetros e alterou-se o outro, verificando os efeitos de cada um. As taxas de crossover em que os resultados foram melhores estavam entre 0.2 e 0.1. Quanto maior a taxa maior era a queda de desempenho do algoritmo (tabela 5).

Taxa de Crossover	Hits
0.5	25
0.3	28
0.2	29
0.1	31

Tabela 5: Diferentes parâmetros testados para o operador de crossover.

No caso da taxa de mutação, os resultados demonstraram que o algoritmo tinha seu desempenho melhorado quando o valor de mutação ficava em torno de 0.01. Da mesma forma que o crossover, quando esses valores eram alterados para números maiores, a eficiência do algoritmo

era menor (Tabela 6).

Taxa de Mutação	Hits
0.50	11
0.20	12
0.10	18
0.05	25
0.01	27

Tabela 6: Diferentes parâmetros testados para o operador de mutação.

Uma possível explicação para o ocorrido em ambos os casos pode ser elaborada. Uma mutação alta poderia fazer com que alinhamentos melhores fossem alterados de forma a piorá-los, com a movimentação constante entre espaços e resíduos. Com uma taxa de mutação mais baixa, a variabilidade genética dos indivíduos da população é menor, juntamente com a exploração do espaço de busca. Esta baixa taxa de mutação seria mais indicada em casos de refinamento dos alinhamentos.

Por outro lado, no caso do crossover, quando um indivíduo acima da média é encontrado, é possível que este localize-se em uma área promissora do espaço de busca. Uma taxa de crossover alta pode resultar em uma diminuição do número de indivíduos que explorarão o espaço de busca naquela região em especial (por meio de clonagens e mutação). Dessa forma, o desempenho do *AG* ficaria aquém do seu potencial.

Quanto à taxa de substituição, vários valores foram testados mas esta parece não influir de forma significativa no desempenho do *AG*.

5.3. Critério de parada

O critério de parada por número máximo de gerações foi descartado devido aos resultados pouco satisfatórios logo com os primeiros testes.

Os resultados não satisfatórios com este critério provavelmente se devem ao fato da aleatoriedade dos indivíduos da população inicial. Quando era definido um número de gerações limitado para a população, esta não conseguia explorar adequadamente o espaço de busca, mesmo com os operadores de mutação e crossover. Assim, o *AG* não conseguia convergir para um resultado adequado.

No segundo critério o *AG* pára após 500 gerações sem alteração do melhor indivíduo da população.

5.4. Experimentos com os casos de teste

Nos primeiros testes, utilizou-se um AG com codificação de *gap simples* e sem o mecanismo *elitismo.* Em seguida, para efeitos comparativos, utilizou-se a codificação de *gap avançado sem elitismo* (Tabela 7 e Figura 15). O experimento foi executado 3 vezes e a média foi tirada em cada um dos casos.

A codificação com gap avançado resultava em melhoria considerável no algoritmo quando comparada a codificação simples. Tal melhoria provavelmente é devido ao fato que os resíduos estavam mais agrupados no alinhamento e quando o espaço era inserido entre resíduos, ocorria uma melhoria no valor de aptidão do alinhamento.

Casos de Teste		***Gap Simples***	***Gap Avançado***
Similaridade	**Swiss-Prot Accession**		
Short < 25%	P20857	11,00	27,67
	P10599		
Short 20% a 40%	P00097	9,00	19,67
	P24469		
Short > 35%	P00195	21,67	24,33
	P00198		
Medium < 25%	P02906	13,00	17,67
	P37329		
Medium 20% a 40%	Q27743	25,33	47,33
	P14295		
Medium > 35%	P00784	40,67	43,33
	P14080		
Long < 25%	P00924	24,67	32,33
	P08310		
Long 20% a 40%	P38673	19,33	38,67
	P45888		
Long > 35%	P00489	28,00	82,67
	P00490		

Tabela 7: Resultados da comparação entre gap simples e avançado.

Em codificações com gap simples, esse tipo de situação não é levada em consideração, o que pode gerar alinhamentos com um número muito grande de espaços iniciais na seqüência. Esse tipo de alinhamento normalmente não é muito satisfatório pois os resíduos com pareamento correto podem demorar muitas gerações até conseguirem se agrupar.

Após esse experimento com codificações distintas para o manejo de espaços, novos operadores foram levados em consideração para verificar se os resultados poderiam ser melhorados.

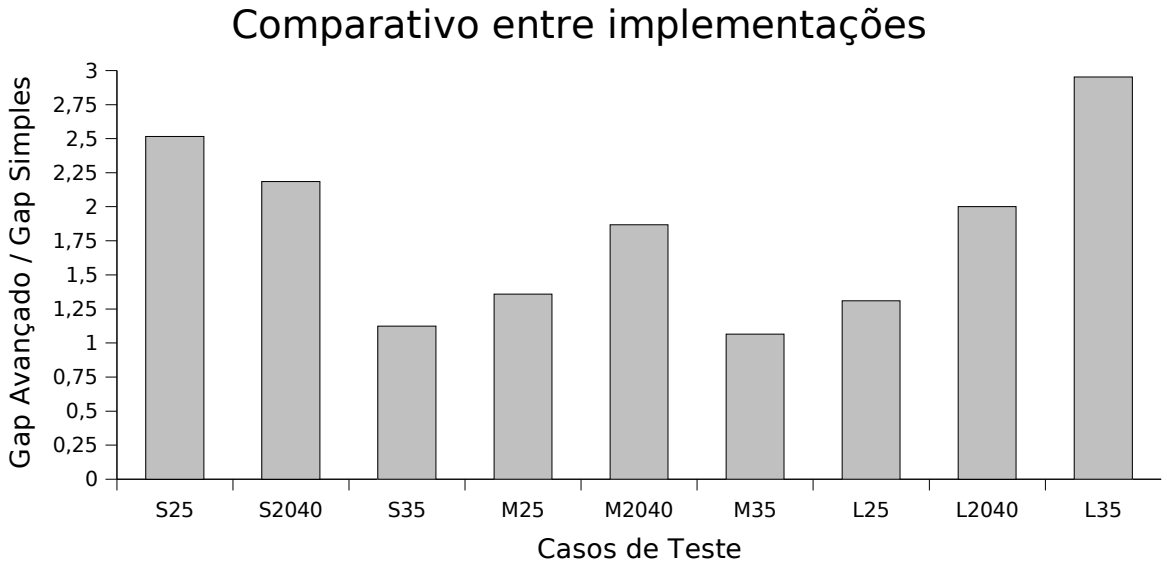


Figura 15: Comparação de desempenho entre gap avançado e simples.

Assim, implementou-se o operador *Elitismo*. Este operador faz com que o melhor indivíduo de uma população não fosse descartado na próxima geração, caso este não tivesse sido escolhido na fase de seleção.

Houve uma melhoria significativa entre a implementação de gap avançado com e sem elitismo (*AG2* e *AG1* respectivamente na Tabela 8 e Figura 16), justificando o emprego desse operador.

Casos de Teste		***AG1***	***AG2***
Similaridade	**Swiss-Prot Accession**		
Short < 25%	P20857	27,67	32
	P10599		
Short 20% a 40%	P00097	19,67	32
	P24469		
Short > 35%	P00195	24,33	40
	P00198		
Medium < 25%	P02906	17,67	61
	P37329		
Medium 20% a 40%	Q27743	47,33	81
	P14295		
Medium > 35%	P00784	43,33	157
	P14080		
Long < 25%	P00924	32,33	79
	P08310		
Long 20% a 40%	P38673	38,67	134
	P45888		
Long > 35%	P00489	82,67	159
	P00490		

Tabela 8: Comparativo entre codificações.

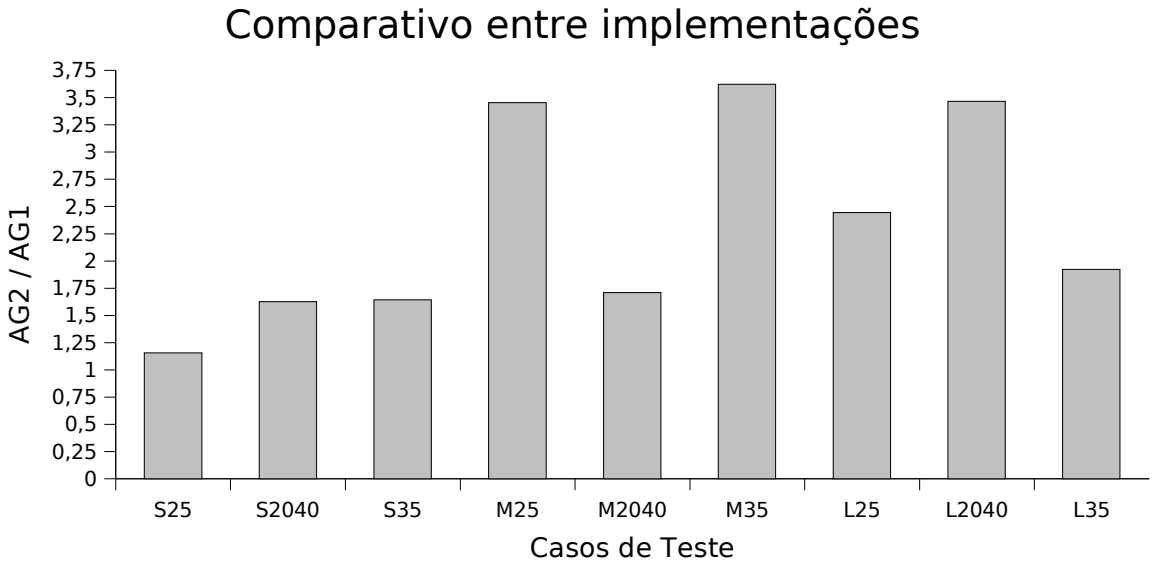


Figura 16: Comparação de desempenho entre a implementação com e sem elitismo.

Os resultados obtidos pelo AG proposto foram comparados com os resultados do programa *bl2seq* (Tabela 9).

Apesar dos resultados de grande parte dos testes do AG serem inferiores ao *bl2seq*, notou-se um grupo de seqüências onde o número de hits obtidos pelo AG foi maior. Eram os de seqüências curtas e de baixa similaridade (até 25%), como demonstrado na Figura 17.

Casos de Teste		***AG hits***	***Blast hits***
Similaridade	**Swiss-Prot Accession**		
Short < 25%	P20857	32	29
	P10599		
Short 20% a 40%	P00097	32	36
	P24469		
Short > 35%	P00195	40	79
	P00198		
Medium < 25%	P02906	61	205
	P37329		
Medium 20% a 40%	Q27743	81	245
	P14295		
Medium > 35%	P00784	157	565
	P14080		
Long < 25%	P00924	79	374
	P08310		
Long 20% a 40%	P38673	134	441
	P45888		
Long > 35%	P00489	159	1952
	P00490		

Tabela 9: Casos de teste utilizados nos experimentos com respectivos resultados.

Um novo conjunto de testes foi selecionado do SWISS-PROT (Tabela 10), todas com características semelhantes àquelas em que o AGs foram superiores no teste anterior.

Novamente, os resultados do AG mostraram um número de pareamentos corretos próximo ou superior daquele obtido pelo *bl2seq*, como demonstrado na Figura 18.

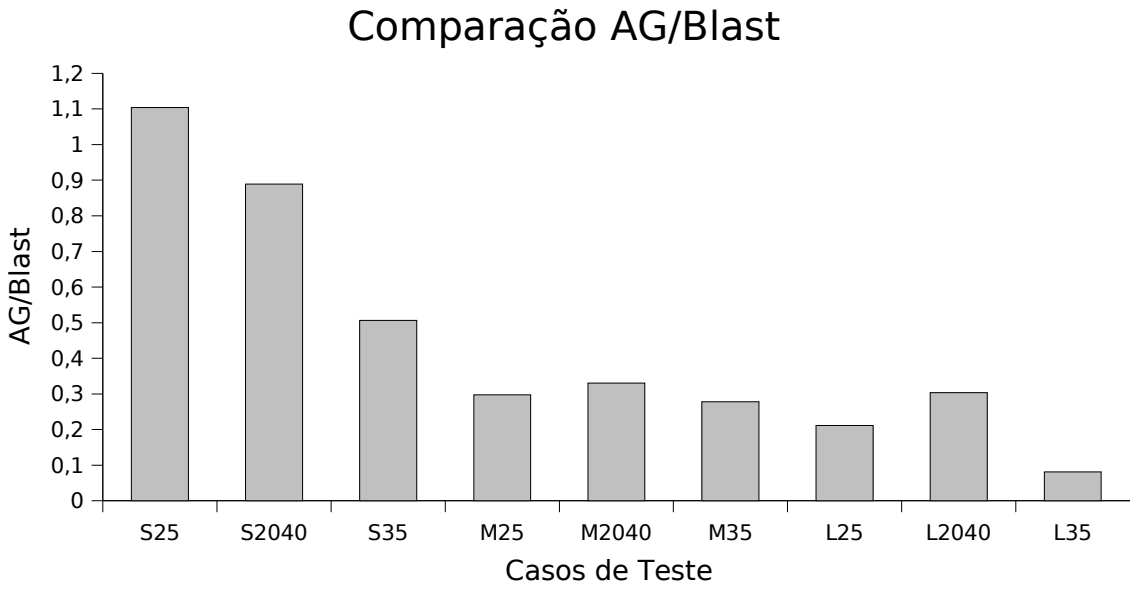


Figura 17: Comparação entre AG e bl2seq utilizando os casos de teste gerais.

Casos de Teste – short < 25%		***AG hits***	***Blast hits***
Identificação do teste	**Swiss-Prot Accession**		
Thi1	P20857	31,00	29
	P10599		
Thi2	P00277	19,60	21
	P20857		
Thi3	P20942	20,50	20
	P20857		
Ubi1	P56408	23,00	17
	P62988		
Ubi2	P62834	29,00	28
	P62988		

Tabela 10: Novos testes realizados, utilizando seqüências menos que 25% similares entre si.

Apesar dos resultados satisfatórios encontrados pela ferramenta no caso de seqüências curtas com menos de 25% de similaridade, não se pode afirmar que esta seria uma ferramenta muito útil em termos

práticos. Os pesquisadores normalmente buscam ferramentas que alinhem seqüências com alto grau de similaridade e não foi esse o caso desta ferramenta, até onde se foi possível estudar. A não ser para casos especiais de análises filogenéticas entre seqüências de proteínas que já poderiam ter divergido há muito tempo, uma ferramenta que só conseguisse melhorar esse tipo especial de alinhamento, não seria muito recomendada.

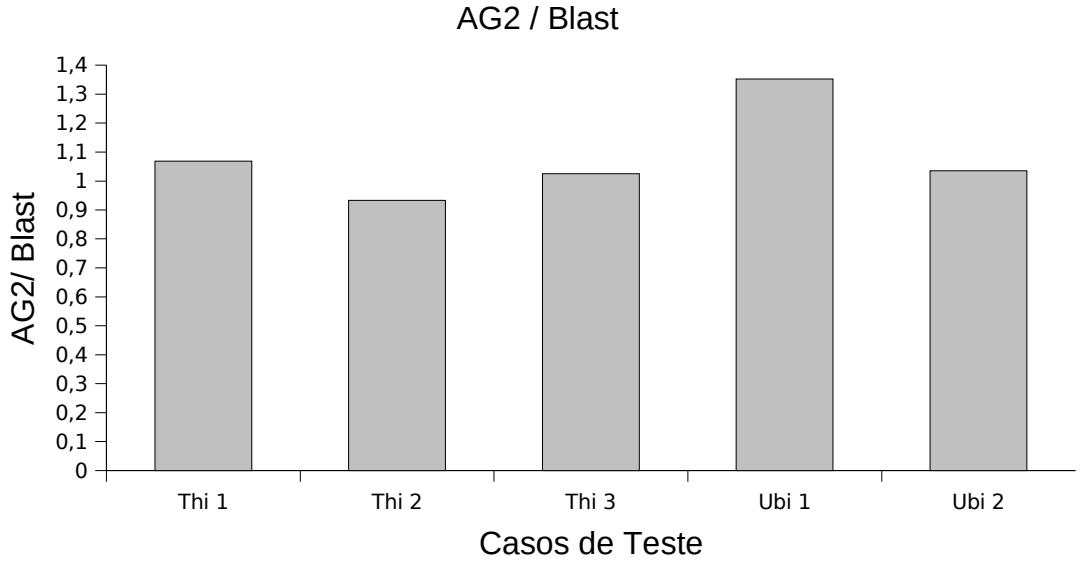


Figura 18: Comparação entre AG e bl2seq utilizando os casos de teste de seqüências curtas com até 25% de similaridade.

Alguns problemas foram possíveis de se identificar para que os outros casos de teste não tivessem resultados igualmente satisfatórios:

- A população inicial deveria aumentar consideravelmente a medida que as próprias seqüências escolhidas fossem mais longas. Caso contrário, não seria possível criar um espaço de busca adequado para que o *AG* explorasse. Quanto mais longa fosse a seqüência, mais tempo computacional seria gasto (vários dias) para que um

resultado fosse encontrado, o que ficou inviável;

- A própria implementação, que poderia ser otimizada, quer seja com o uso de outra linguagem de programação, quer seja pela própria reestruturação do código;
- A metodologia poderia ser modificada em alguns aspectos com o objetivo de agilizar e melhorar os resultados nos outros casos de teste. Como por exemplo modificar o modo como o cromossomo é construído, fazendo com que a inserção de espaços entre seqüências fosse realizada conforme a necessidade.

Apesar disso, foi possível se notar que os *AGs* poderiam ser utilizados talvez não como uma ferramenta principal de alinhamento e sim como uma ferramenta para otimização ou melhorias de resultados, em ferramentas híbridas com outras técnicas, como, por exemplo, refinamento de resultados obtidos pelo *Blast*.

6. CONCLUSÕES E PERSPECTIVAS FUTURAS

Como citado anteriormente, a Bioinformática diz respeito à utilização de técnicas e ferramentas da computação para a resolução de problemas da biologia. Uma das ferramentas mais utilizadas na área de Bioinformática seria relacionada com o alinhamento de seqüências, já que esta serve de suporte para várias outras ferramentas como, por exemplo, de análises filogenéticas ou modelamento tridimensional de proteínas.

Ao longo desta pesquisa de mestrado os objetivos foram, incialmente, estudar as principais técnicas de alinhamentos de seqüências utilizadas em problemas de Bioinformática. O estudo voltou-se principalmente ao pacote de ferramentas *Blast*, já que é a comumente utilizada pelos pesquisadores. Mas outras ferramentas foram estudadas, como *Dot Plot* e *FASTA*, afim de levar a um maior aprofundamento no assunto e conhecimento de diferentes técnicas.

Baseado no conhecimento adquirido em alinhamentos de seqüências, foi proposto um método para alinhamento de seqüências baseado em *AGs*. Esse método foi implementado na linguagem *Object Pascal*, utilizando o ambiente de desenvolvimento *Kylix*, sob a plataforma *Linux*.

Após implementada, a ferramenta foi aplicada em seqüências de proteínas com diversos graus de similaridades e tamanho, para se observar a desempenho dessa nova ferramenta. Para efeito comparativo, utilizou-se a ferramenta *bl2seq* do pacote de ferramentas *Blast*.

Os resultados sugerem que os *AGs* podem obter alinhamentos

melhores que os alinhamentos obtidos pelo *Blast* em seqüências curtas, com até 25% de similaridade. Em outros tipos de seqüências, esses valores foram inferiores, o que pode ser conseqüência do tamanho inicial da população, insuficiente para explorar adequadamente o espaço de busca.

Testou-se outros tamanhos iniciais de população para tais seqüências, contudo devido ao tempo requerido para que uma solução adequada fosse encontrada, não foi possível verificar até o momento se o problema dos resultados em outros tipos de seqüência foi o tamanho inicial da população ou se a metodologia empregada não seria adequada para seqüências maiores e com outros graus de similaridade.

Pode-se salientar duas vantagens na metodologia empregada. A primeira é que o *AG* mostra o alinhamento total entre as duas seqüências, tanto as regiões similares, como as pouco similares. A ferramenta *bl2seq* não possui tal característica, por ser uma ferramenta de alinhamento local. Dessa forma, com *AGs*, informações sobre o alinhamento estariam prontamente disponíveis.

Além disso, o *AG* é uma metodologia inerentemente paralelizável, o que permite a distribuição da execução de um experimento entre várias máquinas, diminuindo o tempo de processamento.

Em trabalhos futuros, a meta é otimizar a implementação do *AG* visando uma diminuição no tempo de execução. Isso permitiria a exploração de regiões maiores do espaço de busca, o que poderia melhorar a qualidade dos resultados. Outra conseqüência seria a

verificação do desempenho dos *AGs* em outros casos de teste onde, no presente trabalho, os resultados não foram satisfatórios e biologicamente relevantes.

Um estudo adicional poderia ser realizado para desenvolver uma implementação híbrida, onde o *AG* poderia otimizar um resultado já produzido pelo *Blast*. Nesse caso, o resultado obtido pelo *Blast* seria um dos indivíduos da população inicial do *AG*. Os outros indivíduos poderiam ser gerados aleatoriamente ou seguindo algum padrão relacionado ao resultado do *Blast*, afim de direcionar a busca.

Finalmente, como resultados do Projeto de Mestrado desenvolvido, podem ser mencionados:

- Um conhecimento aprofundado de técnicas de alinhamento de seqüências;
- Apresentação da técnica de *AGs* como uma possível metodologia para busca de melhores resultados ou refinamentos daqueles já obtidos por outras ferramentas;
- Promover uma interação entre pesquisadores das áreas de Computação e Biológicas;
- E, finalmente, uma contribuição para as pesquisas em Bioinformática, com o emprego de uma nova metodologia que posteriormente possa ser utilizada como mais uma opção para análises de seqüências em Projetos Genomas.

REFERÊNCIAS

[1] Voet, D., Voet, J. G., Pratt, C. W., 2002. Fundamentos de Bioquímica. Artmed Editora.

[2] Baldi, P., Brunak, S., 1998. Bioinformatics: The Machine Learning Approach.. MIT Press.

[3] Setúbal, J., Meidanis, J., 1997. Introduction to Computational Biology. PWS Publishing Company.

[4] Altschul, S.F., Gish, W., Miller, W., Myers, E. W., Lipman, D. J., 1990. Basic Local alignment search tool. J. Mol. Biol. 215, 403-410.

[5] Cancino, W., Liang, Z., Carvalho, A. C. P. L. F., 2003. Aplicação de Algoritmos Multi-Objetivo para o alinhamento de proteínas. Proceedings of the IV Encontro Nacional de.

[6] Wayama, M., Takahashi, K., Shimizu, T., 1995. An Approach to Amino Acid Sequence Alignment Using a Genetic Algorithm. Genome Informatics, 6, 122-123.

[7] Hanada, K., Yokoyama, T., Shimizu, T., 2000. Multiple Sequence Alignment by Genetic Algorithm. Genome Informatics 11: 317-318.

[8] Yokoyama, T., Watanabe, T., Taneda, A., Shimizu, T., 2001. A web server for multiple sequence alignment using Genetic Algorithm. Genome Informatics 12 : 382-383.

[9] Goldberg, D. E., 1989. Genetic Algorithms in Search, Optimization, and Machine Learning. Addison-Wesley Publishing Company, Inc..

[10] Telles, G.P., Braga, M. D. V., Dias, Z., Lin, T. L., Quitzau, J. A. A., da,

2001. Bioinformatics of the Sugarcane EST Project. Genetics and Molecular Biology 24 (1-4): 9-15.

[11] Gibas, C., Jambeck, P., 2001. Developing Bioinformatics Computer Skills. O'Reilly & Associates, Inc..

[12] Vettore, A., da Silva, F. R., Kemper, E., Arruda, P., 2001. The libraries that made SUCEST. Genetics and MolecularBiology 24 (1-4): 1-7.

[13] Baxevanis, A. D., Oullette, B. F. F., 2001. Bioinformatics A Pratical Guide to the Analysis of Genes and Proteins. Wiley-Interscience.

[14] Mount, D. W., 2001. Bioinformatics: Sequence and Genome Analysis. Cold Spring Harbor Laboratory Press.

[15] Maizel, J. V., Lenk, R. P., 1981. Enhanced graphic matrix analysis of nucleic acid and protein sequences. Acad Sci USA 78:7665.

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

[17] Smith, T. F., Waterman, M. S, 1981. Identification of common molecular subsequences. J. Mol. Biol. 147, 195-197.

[18] Weir, B. S., 1996. Genetic Data Analysis II. Sinauer Associates, Inc..

[19] Ridley, M., 1996. Evolution. Blackwell Science, Inc..

[20] Stearns, S. C., Hoekstra, R. F., 2003. Evolução - Uma Introdução. Atheneu.

[21] Mitchell, M., 1999. An introduction to Genetic Algorithms. MIT Press.

[22] Fogel, G. B, Corne, D. W., 2003. Evolutionary Computation in

Bioinformatics. Morgan Kaufmann Publishers.

[23] Bahr, A., Thompson, J. D., Thierry, J. C., Poch, O., 2001. BAliBASE enhancements for repeats, transmembrane, sequences and circular... Nucleic Acids Research, vol 29.