Query by Playing - Formulas e Algoritmos

William Wolff

March 2019

1 Introdução

Para entendermos corretamente o funcionamento do algoritmo desenvolvido e utilizado nesta pesquisa, é necessário primeiramente entender o funcionamento de uma convolução matemática e o que são núcleos de convolução, bem como entender no que isto afeta no funcionamento de algoritmos de alinhamento de sequência e consecutivamente o cálculo de similaridade sendo utilizado para determinar as classificações utilizadas para a avaliação de resultados.

2 Convolução Matemática

A convolução é uma operação matemática importante no processamento de sinais e imagem, pois realiza uma operação matemática entre o sinal original (Convolução 1D - Vetor) , ou imagem original (Convolução 2D - Matriz), de forma que o resultado desta operação gere um novo Sinal ou Imagem melhorado ou alterado.

Se pensarmos no primeiro sinal como a entrada da convolução e o segundo sinal como um "filtro" deste sinal de entrada, produzindo o sinal de saída, podemos chegar a seguinte definição genérica de uma convolução:

$$Convolução(f * g)(i) = \sum_{j=0}^{m} g(j) * f(m - j) \tag{1}$$

Portanto uma convolução ,de acordo com a definição genérica, seria a soma dos valores resultantes da operação matemática sendo realizada na fórmula de convolução, para cada item de entrada sob o núcleo de convolução (também chamado de filtro ou máscara de processamento).

Para melhor exemplificar esta dinâmica , consideremos um sinal 1D , que pode ser representado por um vetor , chamado aqui de f com o tamanho n=7:

10	10	10	10	10	10	10

Considerando agora um nucleo de convolução (Também 1D)chamado aqui de g, com tamanho m=3:

2	2	2

E finalmente um vetor final que receberá o resultado da operação de convolução, com valores zerados:

0	0	0	0	0	0	0

Imaginemos agora que, precisamor determinar a convolução de f em g(3), os vetores poderiam ser dispostos da seguinte maneira:

10	10	10	10	10	10	10
	2	2	2			

$$10 * 2 + 10 * 2 + 10 * 2 = 60 \tag{2}$$

Portanto se realizarmos esta operação para cada uma das posições do vetor do sinal de entrada obteremos o vetor:

50	140	320	680	1400	2840	2860

2.1 O Núcleo de Convolução

Como pudemos ver no vetor 1D resultante , quando estamos percorrendo cada item do vetor e aplicando convolução através do filtro, os valores de cada item do vetor original vão sendo atualizados e afetam o resultado de seus vizinhos.

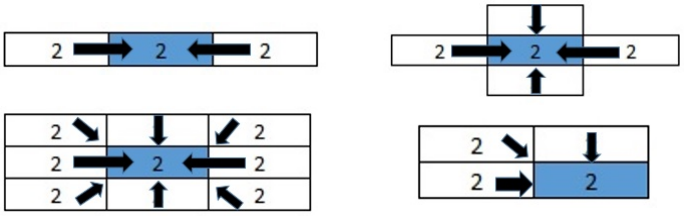
este vetor de "filtro" é chamado de núcleo de convolução , onde o mesmo, de acordo com o seu formato, define a forma (Sequência e Ordem) no qual as operações matemáticas serão realizadas para se chegar ao resultado da convolução.

Por exemplo, na convolução 1D abordada anteriormente o núcleo de convolução possui 3 valores, onde o centro da convolução é a posição no vetor resultante que receberá o resultado da convolução mostrada na figura 1.

Figure 1: Nucleo de Convolução 1D.

Existem diversos tipos de núcleos de convolução que são adaptados de acordo com a estrutura de dados envolvida no algoritmo de convolução, onde o ponto comum entre eles é sempre possuir um ponto central para concentrar o resultado das operações matemáticas, e desta forma controlar o "deslizamento" do núcleo sobre o sinal/imagem de entrada, conforme podemos visualizar nos exemplos da Figura 2.

Figure 2: Tipos de Núcleo: a) 1D , b) Cruz , c) Matriz , d) Canto Matricial



A característica de possuir sempre um ponto central e o seu caminhamento pela estrutura de dados de entrada , caracterizam o núcleo de convolução e justificam o mesmo de ser comumente chamado de Máscara de Processamento, ou apenas Máscara ou Filtro.

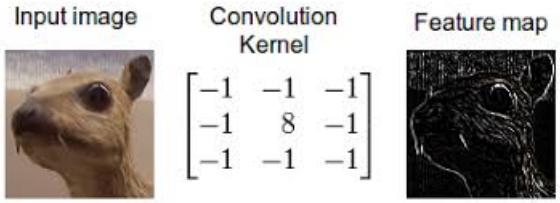
2.2 Convoluções Bidimensionais

As convoluções bidimensionais são uma extensão das convoluções vetoriais e são muito utilizadas pelo processamento de imagens para a realização de filtros para a melhoria da qualidade da imagem.

Usualmente os núcleos de convolução utilizados em algoritmos de processamento de imagens são matriciais para que seja possível alterar um pixel da

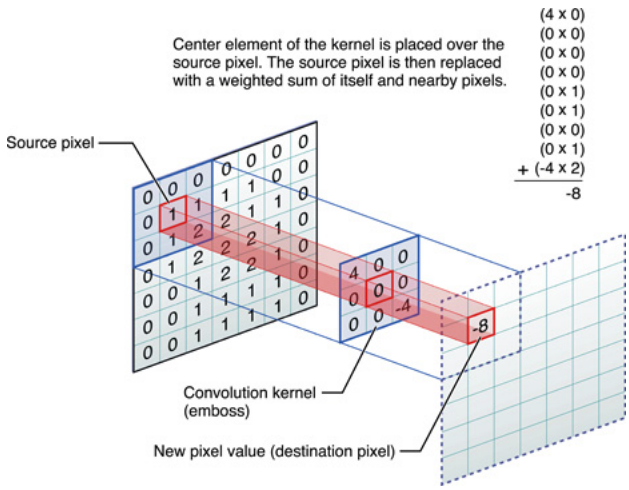
imagem, baseado na convolução dos valores contidos nos pixels vizinhos a ele, conforme mostrado na figura 3.

Figure 3: Filtro Imagem 2D.



A Estrutura de Dados Matricial de uma imagem possibilita a adoção de um nucleo de convolução matricial também, utilizado por inúmeros algoritmos de processamento de imagens e pode ser visualizado na figura 4. Os núcleos de

Figure 4: Matriz Imagem 2D.



convolução matriciais são muito utilizados para a resolução de problemas de visão computacional, como binarização e detecção de bordas para a realização de detecção de objetos específicos na imagem, conforme descrito em(Incluir ref...)

2.3 Os Algoritmos de Alinhamento de Sequência

Os algoritmos de alinhamento de sequência tem por objetivo comparar duas ou mais sequencias para se observar o grau de similaridade que uma sequencia possui em relação a outra.

Os algoritmos de alinhamento de sequencia são muito utilizados na BioInformática para a identificação de cadeias de caracteres correspondentes as proteinas contidas nas sequencias de DNA.

Os algoritmos de alinhamento de sequencia podem ser divididos primordialmente em duas classes:

- Algoritmos de alinhamento Global
- Algoritmos de alinhamento Local

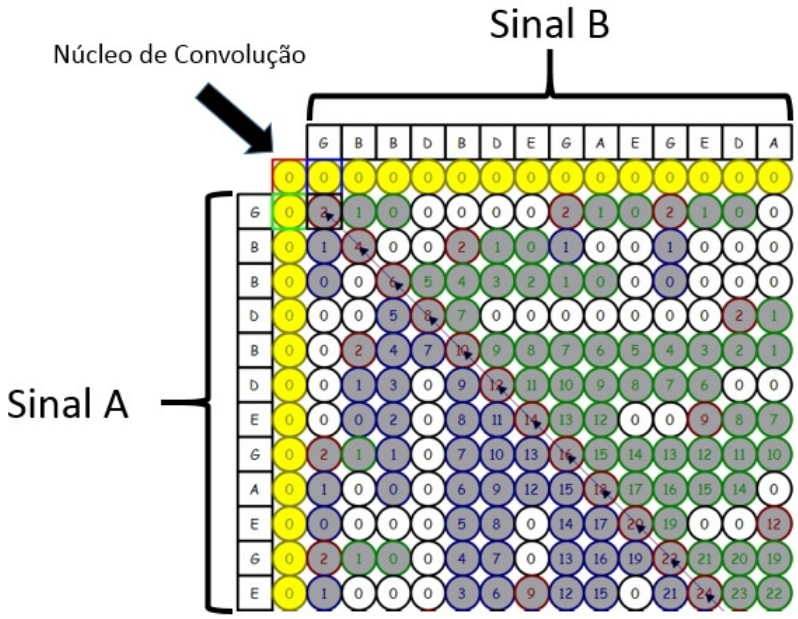
Os algoritmos de alinhamento global procuram alinhar sequencias levando-se em conta toda a sequencia , enquanto os algoritmos de alinhamento local procuram fragmentos de alinhamentos por toda a sequencia.

A grande ligação entre os algoritmos de convolução e os algoritmos de alinhamento Local e Global reside no fato que para poder realizar este alinhamento de sequencias, é utilizado uma matriz 2D , que ao invés de armazenar uma imagem, armazena uma pontuação de similaridade entre cada valor do Sinal A comparado com o Sinal B.

Para a realização do cálculo da pontuação de similaridade, um núcleo de convolução é utilizado, no qual a fórmula varia de implementação para implementação gerando assim diversos tipos de algoritmos de alinhamento.

Portanto para a realização de um alinhamento de sequências, podemos afirmar que está se utilizando um algoritmo de convolução 2D que processa dois sinais vetoriais utilizando um núcleo de convolução para pontuar similaridades entre estes sinais, conforme apresentado visualmente na imagem 5.

Figure 5: Convolução de Alinhamento.



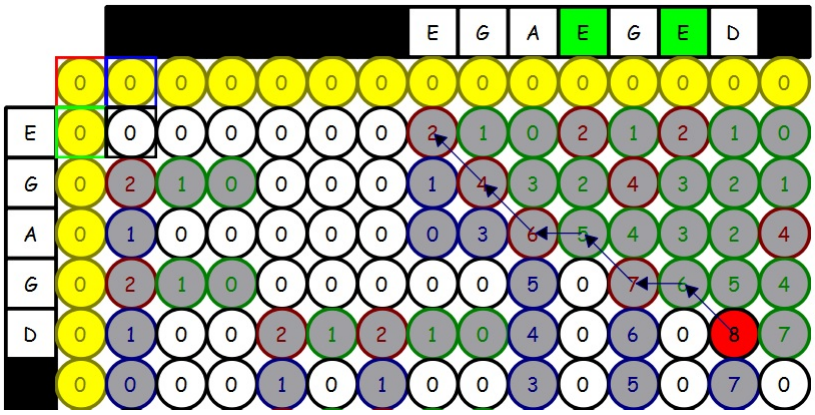
Apesar do núcleo de convolução realizar boa parte da tarefa necessária para um alinhamento de sequência, após o seu processamento possuímos apenas os

valores de pontuação , mas nenhum alinhamento ainda, que deve ser realizado por um processo de caminhamento reverso chamado Backtracing.

2.3.1 O Caminhamento Reverso - Backtracing

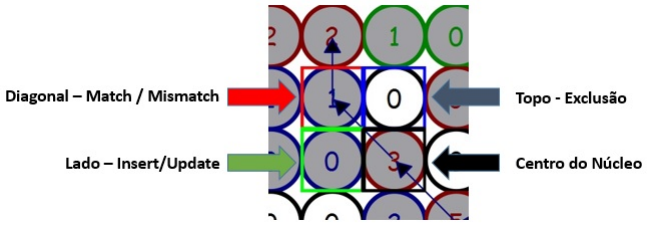
O processo de caminhamento reverso pode ser considerado um pós-processamento do algoritmo de convolução utilizado para a "construção" do caminho de alinhamento. Este processo tem por objetivo identificar o melhor alinhamento entre os sinais levando-se em conta as melhores pontuações de similaridade, conforme podemos visualizar na figura 6.

Figure 6: Caminhamento Reverso de um Alinhamento.



Não há uma regra geral de como este caminho precisa ser traçado, porém os algoritmos de alinhamento mais famosos utilizam um nucleo de convolução do tipo Canto Matricial , onde a direção de atribuição gerada durante o caminhamento convolutivo é armazenado para que no processo de backtracing seja possível "Reconstruir" o caminho gerado pelas melhores pontuações, conforme a figura 7 de um núcleo de convolução para alinhamento local.

Figure 7: Núcleo de Convolução - Alinhamento Local.



2.4 Etapas de Processamento - Alinhamento

De forma análoga ao processamento de uma busca MIR, o processamento de um algoritmo de alinhamento de sequencias , pode ser subdividido em 3 etapas principais para a realização do cálculo de pontuação de similaridade. Estas etapas podem ser identificadas como Inicialização,Processamento e Pós-Processamento.

2.4.1 A Inicialização

A primeira etapa de processamento de um algoritmo de alinhamento de sequencias é a inicialização de um alinhamento.

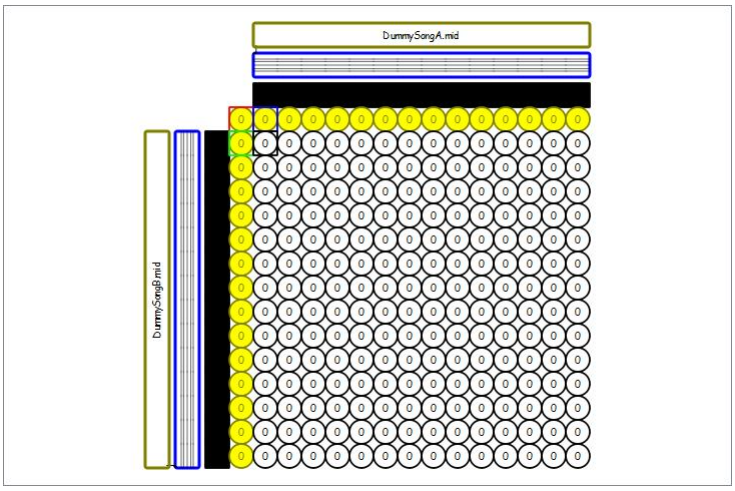
primordialmente a inicialização de um alinhamento visa configurar valores iniciais na matriz de convolução. No caso de um algoritmo de alinhamento, esta matriz é chamada de Substitution Matrix.

A inicialização da Matriz pode ser representada pelas seguintes formulas:

$$\begin{aligned} \text{M(i,0)} &= 0 \ , \ 0 \leq i \leq m \\ \text{M(0,j)} &= 0 \ , \ 0 \leq i \leq n \end{aligned} \tag{3}$$

Esta inicialização é de suma importancia para garantir o bom funcionamento do sistema e garantir que os valores calculados estejam dentro do esperado.

Figure 8: Inicialização de Uma Matriz de Convolução.



Para um alinhamento local adequado, sempre a substitution matrix será inicializada com O Tamanho do sinal(A e B) + 1 , para garantir uma coluna e uma linha com valores zerados e assim evitarmos de realizar o teste de valores inválidos na borda da matriz.

2.4.2 O Processamento

Após a inicialização da substitution matrix, iniciamos o caminhamento convolutivo do algoritmo.

Para cada valor (i,j) contido na matriz,utilizando a Máscara de Processamento, executamos fórmula de convolução definida pelo algoritmo de alinhamento utilizado.

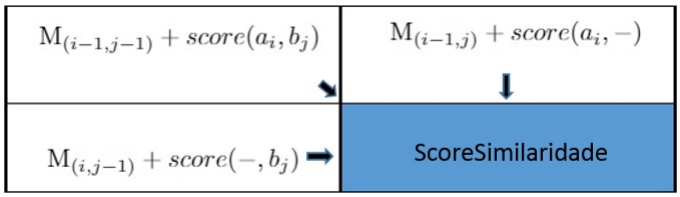
A diferença entre a execução de uma formula de convolução normal e a convolução realizada por um algoritmo de alinhamento, consiste em que para o alinhamento de sequencia cada valor da mascara de processamento(Núcleo de Convolução) é realizado dinâmicamente um calculo, baseado em parametros de processamento pré definidos(Match/Mismatch e Gap).

A formula de calculo da pontuação de similaridade do alinhamento de sequencia pode ser representada da seguinte forma:

$$ScoreSimilaridade(i,j) = max \begin{cases} \mathrm{M}_{(i-1,j-1)} + score(a_i, b_j) \\ \mathrm{M}_{(i-1,j)} + score(a_i, -) \\ \mathrm{M}_{(i,j-1)} + score(-, b_j) \\ 0 \end{cases} \quad (4)$$

A transposição de cada uma destas formulas pode ser visualizada graficamente na imagem 9.

Figure 9: Transposição de Formulas no Núcleo de Convolução.



Uma atenção especial deve ser dada as formulas parciais $score(a_i, b_j)$, $score(a_i, -)$ e $score(-, b_j)$, responsáveis pelo teste de similaridade entre o valor do Sinal A com o Sinal B.

Estas formulas parciais representam os valores de Match , Mismatch e Gap parametrizados no algoritmo de alinhamento e que em conjunto com a direção do valor máximo definem o calculo final da pontuação de similaridade para cada item da matriz.

A função $Max()$ utilizada no algoritmo de alinhamento pode ser representada

através de pseudo código:

Algorithm 1: A Funcao Max

input : A posicao atual i,j
output : Pontuação de Similaridade

Result: O resultado sera a pontuação maxima de similaridade.

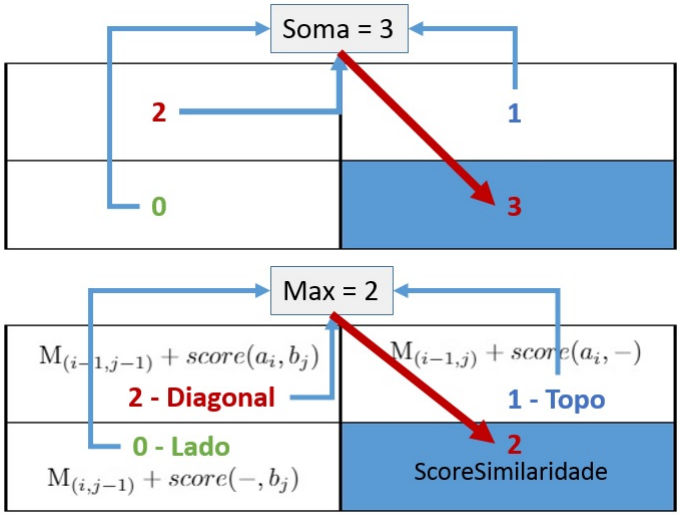
```
1 lado = M[i-1,j ] + gap;
2 topo = M[i ,j-1] + gap;
3 diagonal = M[i-1,j-1] + ScoredeErro : Acerto;
4 retorno = Maximo(topo,lado,diagonal)
```

Outra diferença significativa entre um núcleo de convolução normal e um núcleo de convolução para alinhamento de sequências, é relativo a operação matemática realizada entre os vizinhos para a geração do valor resultante da convolução.

Em um calculo de convolução pela regra geral, o valor resultante da soma entre os seus vizinhos é atribuído ao centro do núcleo de convolução.

Em um calculo de convolução para o alinhamento de uma sequencia local, o valor resultante do maior valor entre os vizinhos, define o valor a ser atribuído ao centro do nucleo de convolução, e define a Direção em que este valor foi originado.Estas diferenças podem ser visualizadas na imagem 10

Figure 10: Comparativo de Núcleo: a) Núcleo Normal , b) Núcleo Alinhamento



É baseado no registro da direção resultante de cada valor calculado pelo núcleo de processamento, que é possível posteriormente realizar o caminhamento reverso do algoritmo.

2.4.3 O Pós-Processamento

Após concluído todo o caminhamento convolutivo da substitution matrix (matriz de convolução) , é realizado o caminhamento reverso dos itens desta matriz para a construção do melhor alinhamento local resultante do processamento do algoritmo.

Este pós-processamento é realizado através da execução sequencial de 3 passos:

1. Obtenção do maior score contido na matriz.
2. A partir da posição do maior item, percorrer a matriz utilizando as direções coletadas durante o processamento, até que se encontre um valor de score zerado(Caminho Bloqueado).
3. Calcular o Score Final de Similaridade do Sinal atribuindo 1 ponto para cada valor do alinhamento resultante de um Match(Diagonal).

Estes 3 passos podem ser representados através de pseudo-código para a realização do backtracing utilizado nesta pesquisa:

Algorithm 2: O Algoritmo de Caminhamento Reverso

Result: Construção do Alinhamento Local.

```
//Passo 01
GetMax(IdxMaxX,IdxMaxY);
//Passo 02
while !TraceEnd do
    //obtem pontuação e direção nos indices X,Y
    IScore = GetVal(IdxMaxX, IdxMaxY, iDirection);
    //decrementa indices de acordo com a direção
    switch (iDirection) do
        case Diagonal do
            IdxMaxX-=1;
            IdxMaxY-=1;
        end
        case Lado do
            IdxMaxY-=1;
        end
        case Topo do
            IdxMaxX-=1;
        end
    end
end
//Passo 03
CalculaScore();
```

A função que retorna o maior valor na matriz de substituição pode ser representada também através de pseudo código:

Algorithm 3: A Função GetMax

Result: Retorna os Indices X,Y com o Maior Valor na Matriz.

```
27 iMaxValue = 0;
28 iMaxValueX = 0;
29 iMaxValueY = 0;
30 for j ← 0 to SizeY do
31 |   for i ← 0 to SizeX do
32 |   |   if Matrix[Idxj][Idxi].iValue ≥iMaxValue then
33 |   |   |   iMaxValue = Matrix[Idxj][Idxi].iValue;
34 |   |   |   iMaxValueX = Idxi;
35 |   |   |   iMaxValueY = Idxj;
36 |   |   end
37 |   end
38 end
39 //Retorno;
40 x = iMaxValueX;
41 y = iMaxValueY;
```

Finalmente a função que calcula o score final, percorre o caminho traçado no pos-processamento e atribui 1 ponto a cada Match encontrado.

Algorithm 4: A Função CalculaScore

Result: Calcula e Retorna a Pontuação de Similaridade Final.

```
42 iTraceScore = 0;
43 for i ← 0 to iPathSize do
44 |   if vTraceBack[i].iDirection == Diagonal then
45 |   |   iTraceScore++;
46 |   end
47 end
```

3 O algoritmo Smith Waterman e Comparativo de Processamento

O algoritmo de alinhamento de sequencia escolhido para a realização da pesquisa é o algoritmo Smith Waterman, para a realização de alinhamento local, ou seja , tanto o núcleo de convolução quanto o caminhamento e inicialização da matriz de substituição foram configurados para a realização deste tipo de alinhamento de sequencia.

O Pseudo-Código do processamento SW segue descrito abaixo:

Algorithm 5: O Algoritmo Smith Waterman

Result: Matriz de Scores Calculada(Top-Down).

```
Inicialização;
D(i,j) = 0;
Calculo de Scores;
for i ← 1 to M do
    for j ← 1 to N do
        D(i,j)= Max(i,j);
    end
end
```

Algorithm 6: A Funcao Max

input : A posicao atual i,j
output : Distancia Maxima

Result: O resultado sera o score maximo de distancia.

```
lado = M[i-1,j ] + gap;
topo = M[i ,j-1] + gap;
diagonal = M[i-1,j-1] + ScoredeErro : Acerto;
retorno = Maximo(topo,lado,diagonal)
```

Todos os outros algoritmos descritos anteriormente foram implementados de forma geral, ou seja, não mudam de acordo com o tipo de alinhamento sendo realizado.

3.1 Parametros de Processamento - SW

Como vimos anteriormente na seção de processamento, as formulas parciais de alinhamento são responsáveis pelos parâmetros the Match, Mismatch e Gap utilizados pelo algoritmo para o calculo da pontuação de similaridade projetada na substitution Matrix.

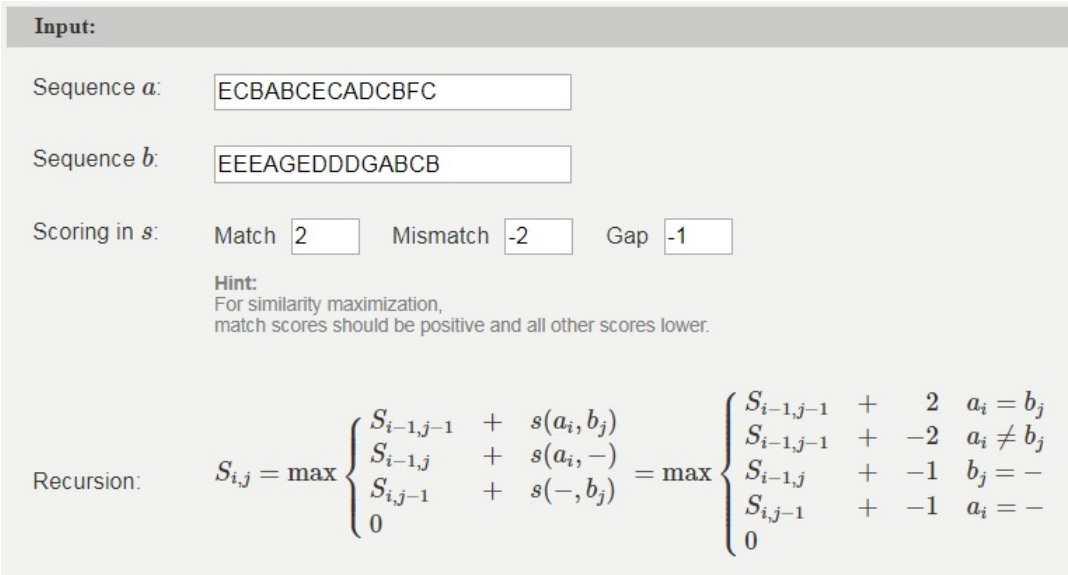
Portanto, para validar o processamento do algoritmo implementado por esta pesquisa e poder prosseguir com os trabalhos, foi escolhido um site web contendo um simulador do algoritmo, configurado os mesmos valores de Match,Mismatch e Gap, e realizado sua comparação.

O simulador Smith Waterman pode ser utilizado no link web: http://rna.informatik.uni-freiburg.de/Teaching/index.jsp?toolName=Smith-Waterman

Os parametros configurados em ambos os algoritmos segue descrito na figura 11, e podem ser utilizados para a realização de outros testes de validação.

Estes testes são importantes para que possamos validar o funcionamento principal do algoritmo , antes de realizarmos alterações , mudanças de valores de erro/acerto e etc.

Figure 11: Parametros de Simulação SW.



3.2 Comparativo de Processamento

Após a configuração correta de ambos os algoritmos, foi realizado o alinhamento de duas cadeias sequenciais de caracteres em cada um dos algoritmos e coletado seus resultados(Figura 12). O objetivo principal deste teste é validar os valores de pontuação de similaridade de ambas matrizes e verificar quais são as diferenças obtidas, já que ambas implementações referem-se ao mesmo algoritmo implementado.

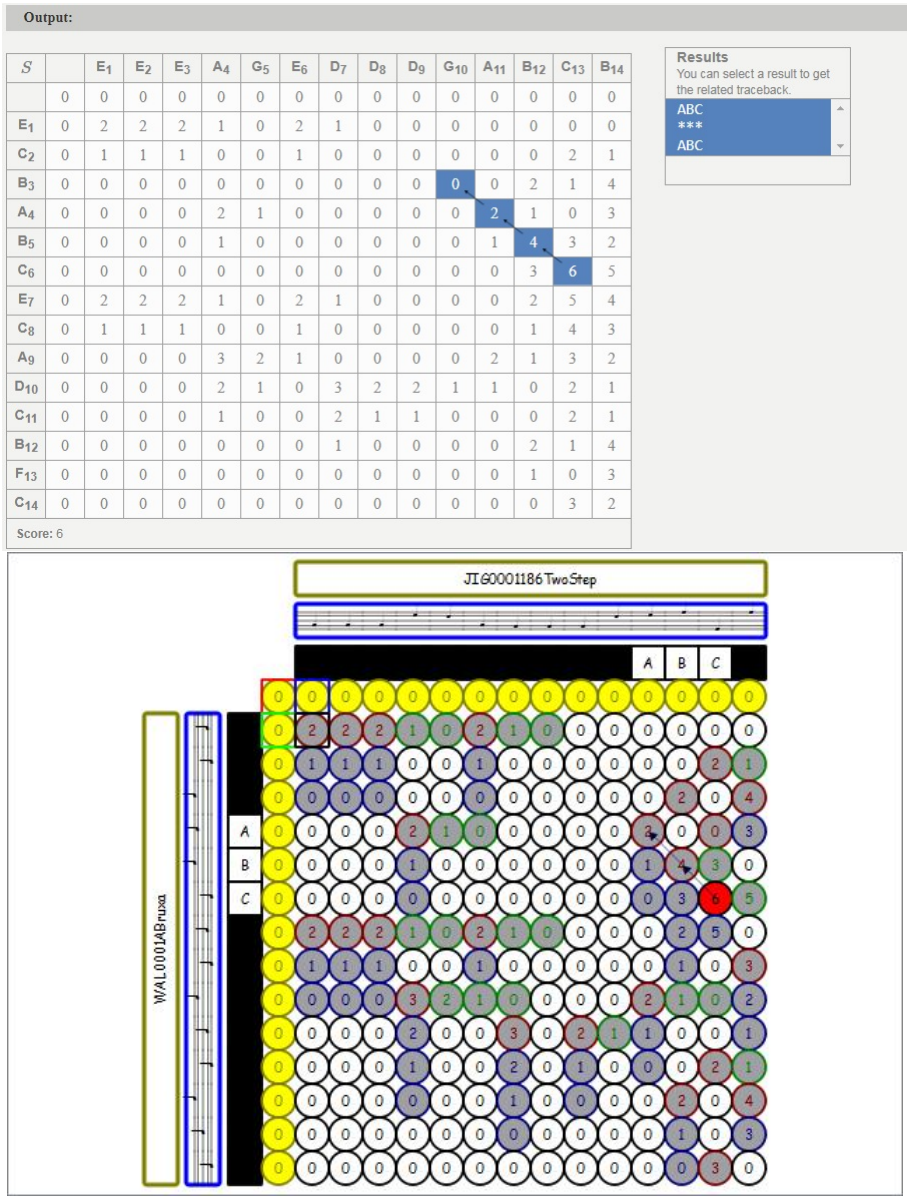
Como resultado, foi obtido os mesmos resultados de pontuação e mesmo alinhamento local, porém em alguns valores da matriz de substituição (que não afetaram os resultados), foram identificados como divergentes e são foco de verificação.

3.3 Tipos de Parametrização Smith Waterman

Aliado as etapas de processamento descritas anteriormente relativas ao algoritmo Smith Waterman, este algoritmo através dos parâmetros de Match , Mismatch e Gap, possibilitam refinar os resultados de alinhamento através da alteração destes parâmetros durante o processamento do algoritmo.

Existem 3 tipos de parametrização de funcionamento para este algoritmo, identificados aqui como Parametrização Fixa(Original Smith Waterman) , Parametrização Linear e Parametrização Affine Gap(Gotho).

Figure 12: Comparativo de Processamento: a) SW , b) QBP



3.3.1 Parametrização Fixa - Smith Waterman Original

A parametrização Fixa é a parametrização original Smith Waterman, ou seja, os valores utilizados nas formulas de calculo de pontuação de similaridade Match , Mismatch e Gap são determinados antes da execução do algoritmo e

não mudam de acordo com o andamento convolutivo do algoritmo.

Todos os testes de validação do processamento de alinhamento local utilizando outras implementações foi realizada utilizando parametros fixos para garantir situações iguais de processamento.

3.3.2 Parametrização Linear

A parametrização Linear utiliza-se de uma formula, ou tabelas de processamento para alterar os parametros de execução do algoritmo de acordo com os testes de similaridade que são realizados a cada iteração do caminhamento convolutivo.

3.3.3 Parametrização Affine Gap

Este tipo de parametrização procura identificar a extensão dos fragmentos vazios (Gaps) gerados durante o processamento do alinhamento local para calcular a penalidade a ser atribuida nos gaps de processamento do algoritmo.

Este é um caso especial de alinhamento, pois passa a utilizar duas outras matrizes de processamento (de igual tamanho) para determinar o inicio e fim do fragmento.

Portanto o falor final das penalidades sendo aplicadas no alinhamento são proporcionais ao tamanho do fragmento gerado entre um alinhamento e outro do algoritmo gerando uma curva exponencial de processamento, ao invés de uma linha linear.

Este tipo de parametrização é muito utilizado na biotecnologia, pois a configuração correta das penalidades aumenta a incidência de fragmentos melhorando a detecção de mutações genéticas no DNA.