Distância de Edição: Programação Dinâmica

O algoritmo funciona da seguinte forma:

dadas duas strings [A e B] e os custos de inserção, remoção e substituição de elementos, é calculado o custo para transformar a string A na string B através do cálculo do custo de transformação de substrings de A em substrings de B e armazenado os resultados em uma tabela.

Distância de Edição

O tamanho da matriz é M+1 x N+1, onde M e N são os tamanhos da string A e B respectivamente.

Cada linha da tabela representa uma substring de A e cada coluna representa uma substring de B. Então a linha i representa a string com os i primeiros caracteres de A [string vazia para i = 0] e a coluna j representa a string com os j primeiros caracteres de B. A posição [i][j] da tabela representa o custo de transformar a substring com i caracteres de A na substring com j caracteres de B

Tabela

		P	A	I
C				
A				
S				
A				

Distância de Edição

A matriz é iniciada com 0 na posição [0][0], para as demais posições da linha 0 é adicionado o custo de uma inserção ao elemento da esquerda e para as demais posições da coluna 0 é adicionado o custo de remoção ao elemento de cima.

Inicialização

		P	A	I
	0	1	2	3
C	1			
A	2			
S	3			
A	4			

Distância de Edição

Para as demais posições [i][j], o algoritmo verifica se é melhor:

-Transformar a substring i na j-1 e inserir um elemento;

Exemplo: substring "cas" → "pa": transforma "cas" em "p" e insere 'a' no fim da substring, resultando em "pa"

-Transformar a substring i-1 na j e remover um elemento;

Exemplo: substring "cas" → "pa": transforma "ca" em "pa", ficando "cas" e depois remove o último caractere ['s']

-Transformar a substring i-1 na j-1 e substituir um elemento (caso o elemento A[i] seja diferente do B[j])

Exemplo: substring "cas" → "pa": transforma "ca" em "p", ficando "ps" e depois substitui o último caractere['s'] por 'a'

Distância de Edição

Exemplo: custo para transformar casa em pai.

Inicialização

		P	A	I
	0	1	2	3
C	1			
A	2			
S	3			
A	4			

Substituição

		P	A	I
	0	1	2	3
C	1	1		
A	2			
S	3			
A	4			

Substituição

		P	A	I
	0	1	2	3
C	1	1	2	
A	2			
S	3			
A	4			

Substituição

		P	A	I
	0	1	2	3
C	1	1	2	3
A	2			
S	3			
A	4			

Substituição

		P	A	I
	0	1	2	3
C	1	1	2	3
A	2	2		
S	3			
A	4			

Match

		P	A	I
	0	1	2	3
C	1	1	2	3
A	2	2	1	
S	3			
A	4			

Distância de Edição

Inserção

		P	A	I
	0	1	2	3
C	1	1	2	3
A	2	2	1	2
S	3			
A	4			

Substituição

		P	A	I
	0	1	2	3
C	1	1	2	3
A	2	2	1	2
S	3	3		
A	4			

Remoção

		P	A	I
	0	1	2	3
C	1	1	2	3
A	2	2	1	2
S	3	3	2	
A	4			

Substituição

		P	A	I
	0	1	2	3
C	1	1	2	3
A	2	2	1	2
S	3	3	2	2
A	4			

...

Substituição

		P	A	I
	0	1	2	3
C	1	1	2	3
A	2	2	1	2
S	3	3	2	2
A	4	4	3	3

Menor Custo = 3

Distância de Edição

Pseudo-Código:

```
Custos: Remoção=R, Inserção=I , Substituição=S e Match=M=0;

m = tamanho[A];
n = tamanho[B];
matriz[0][0] = 0;
Para i = 1 até m
   matriz[i][0] = matriz[i-1][0] + 1  // soma uma I;
Para j = 1 até n
   matriz[0][j] = matriz[0][j-1] + 1  // Soma uma R;
Para i = 1 até m
   Para j = 1 até n
      Se A[i] == B[j]
         custoExtra = 0 //Operação M;
      Senão
         custoExtra = 1 //Operação S;
      matriz[i][j] = Mínimo(matriz[i-1][j] +1, matriz[i][j-1] +1,
                            matriz[i-1][j-1] + custoExtra];
devolva matriz[m][n];
```