Biyoinformatiğe Giriş Proje 3:

Derya Öztürk

Local alignment ve global alignment, ikisi de DNA, RNA veya protein dizilerini karşılaştırmak için kullanılan yöntemlerdir. Farkları ise, local alignment, yalnızca iki dizinin benzer bölümlerini eşleştirirken, global alignment, iki diziyi tüm uzunlukları boyunca karşılaştırır ve farklılık olan her bölgeyi de dahil eder.

Local Aligment İçin:

```python
def local_alignment(s1, s2, match, mismatch, gap):
    m = len(s1) # s1'in uzunluğunu m değişkenine atama
    n = len(s2) # s2'nin uzunluğunu n değişkenine atama

    # Skor matrisini 0'larla doldurma
    score_matrix = [[0 for j in range(n + 1)] for i in range(m + 1)]

    # Takip matrisini 0'larla doldurma
    traceback_matrix = [[0 for j in range(n + 1)] for i in range(m + 1)]

    # Maksimum skor ve pozisyonunu takip etmek için değişkenleri başlatma
    max_score = 0
    max_score = 0
    max_i = 0
    max_j = 0

    # Skor ve takip matrislerini doldurma
    for i in range(1, m + 1):
        for j in range(1, n + 1):
            # Calculate score for match/mismatch and gap
            match_mismatch_score = score_matrix[i-1][j-1] + (match if s1[i-1] == s2[j-1] else mismatch)
            gap_i_score = score_matrix[i-1][j] + gap
            gap_j_score = score_matrix[i][j-1] + gap

            # Mevcut pozisyon için skoru eşleşme/uyumsuzluk ve boşluk skorlarının maksimumu olarak ayarla
            score_matrix[i][j] = max(0, match_mismatch_score, gap_i_score, gap_j_score)

            # Maksimum skor yönünü göstermek için takip matrisini güncelle
            if score_matrix[i][j] == 0:
                traceback_matrix[i][j] = 0
```

```
        # Maksimum skor yönünü göstermek için takip matrisini güncelle
        if score_matrix[i][j] == 0:
            traceback_matrix[i][j] = 0
        elif score_matrix[i][j] == match_mismatch_score:
            traceback_matrix[i][j] = 1
        elif score_matrix[i][j] == gap_i_score:
            traceback_matrix[i][j] = 2
        else:
            traceback_matrix[i][j] = 3

        # Maksimum skor ve pozisyon değişkenlerini güncelle
        if score_matrix[i][j] >= max_score:
            max_score = score_matrix[i][j]
            max_i = i
            max_j = j

# Uygun hizalanmış diziler için değişkenleri başlatma
aligned_s1 = ""
aligned_s2 = ""

 # Maksimum skorun olduğu pozisyondan 0 skorlu pozisyona kadar geriye doğru takip etme
i = max_i
j = max_j
while i > 0 and j > 0 and score_matrix[i][j] > 0:
    if traceback_matrix[i][j] == 1:
        aligned_s1 = s1[i-1] + aligned_s1
        aligned_s2 = s2[j-1] + aligned_s2
        i -= 1
        j -= 1
    elif traceback_matrix[i][j] == 2:
```

```
    # Maksimum skorun olduğu pozisyondan 0 skorlu pozisyona kadar geriye doğru takip etme
    i = max_i
    j = max_j
    while i > 0 and j > 0 and score_matrix[i][j] > 0:
        if traceback_matrix[i][j] == 1:
            aligned_s1 = s1[i-1] + aligned_s1
            aligned_s2 = s2[j-1] + aligned_s2
            i -= 1
            j -= 1
        elif traceback_matrix[i][j] == 2:
            aligned_s1 = s1[i-1] + aligned_s1
            aligned_s2 = "-" + aligned_s2
            i -= 1
        else:
            aligned_s1 = "-" + aligned_s1
            aligned_s2 = s2[j-1] + aligned_s2
            j -= 1

    return (max_score, aligned_s1, aligned_s2)


score, aligned_seq1, aligned_seq2 = local_alignment("KVLEFGY", "EQLLKALEFKL",4,-2,-1)
print("Score:", score)
print("Alignment:")
print(aligned_seq1)
print(aligned_seq2)
```

```
PS C:\Users\MONSTER> python -u "c:\Users\MONSTER\Desktop\Python.py\proje3.py"
Score: 14
Alignment:
KVLEF
KALEF
```

Global Aligment İçin:

```python
def global_alignment(seq1, seq2, match_score=1, mismatch_penalty=-1, gap_penalty=-1):
    # ilk adım: matrisi hazırlayıp ve başlangıç değerlerini belirledim
    n, m = len(seq1), len(seq2)
    dp_matrix = [[0] * (m+1) for _ in range(n+1)]
    for i in range(1, n+1):
        dp_matrix[i][0] = i * gap_penalty
    for j in range(1, m+1):
        dp_matrix[0][j] = j * gap_penalty

    # ikinci adım: matrisi doldurdum
    for i in range(1, n+1):
        for j in range(1, m+1):
            match = dp_matrix[i-1][j-1] + (match_score if seq1[i-1] == seq2[j-1] else mismatch_penalty)
            delete = dp_matrix[i-1][j] + gap_penalty
            insert = dp_matrix[i][j-1] + gap_penalty
            dp_matrix[i][j] = max(match, delete, insert)

    # üçüncü adım: while döngüsü ile en uygun hizalamayı buldum
    aligned_seq1, aligned_seq2 = '', ''
    i, j = n, m
    while i > 0 or j > 0:
        if i > 0 and j > 0 and dp_matrix[i][j] == dp_matrix[i-1][j-1] + (match_score if seq1[i-1] == seq2[j-1] else mismatch_penalty):
            aligned_seq1 += seq1[i-1]
            aligned_seq2 += seq2[j-1]
            i -= 1
            j -= 1
        elif i > 0 and dp_matrix[i][j] == dp_matrix[i-1][j] + gap_penalty:
            aligned_seq1 += seq1[i-1]
            aligned_seq2 += '-'
            i -= 1
```

```python
    i, j = n, m
    while i > 0 or j > 0:
        if i > 0 and j > 0 and dp_matrix[i][j] == dp_matrix[i-1][j-1] + (match_score if seq1[i-1] == seq2[j-1] else mismatch_penalty):
            aligned_seq1 += seq1[i-1]
            aligned_seq2 += seq2[j-1]
            i -= 1
            j -= 1
        elif i > 0 and dp_matrix[i][j] == dp_matrix[i-1][j] + gap_penalty:
            aligned_seq1 += seq1[i-1]
            aligned_seq2 += '-'
            i -= 1
        else:
            aligned_seq1 += '-'
            aligned_seq2 += seq2[j-1]
            j -= 1

    # dördüncü adım: sonucu tersine çevirip ve döndürdüm
    return aligned_seq1[::-1], aligned_seq2[::-1],dp_matrix[i-1][j-1]

#kullanacağım sekans değerlerini ve puanlarını verdim
seq1 = 'GATTACA'
seq2 = 'GCATGCU'
match = 1
mismatch = -1
gap = -1

alignment,sekans2, dp_matrix = global_alignment(seq1, seq2, match, mismatch, gap)
print('Alignment:', alignment)
print('Alignment:',sekans2)
print('Score:', dp_matrix)
```

```
PS C:\Users\MONSTER> python -u "c:\Users\MONSTER\Desktop\Python.py\proje3.py"
Alignment: G-ATTACA
Sekans: GCA-TGCU
Score: 0
PS C:\Users\MONSTER>
```