Programowanie dynamiczne w bioinformatyce

Tomasz Żok

1 Algorytm Needlemana-Wunscha

1. W problemie znalezienia dopasowania sekwencji (alignment), jako dane wejściowe mamy sekwencje: s oraz t
2. Rozwiązanie stanowią sekwencje: s' oraz t', takie że:
   - Długość s' = długość t'
   - Oprócz alfabetu użytego w s i t, sekwencje wyjściowe mogą zawierać znak pusty (gap) oznaczony myślnikiem –
   - Na każdej pozycji, znaki w s' i t':
     - są równe (match)
     - są różne (mismatch)
     - jeden z nich jest myślnikiem (indel, insertion/deletion)
3. Każda pozycja jest punktowana (domyślnie match = 1, mismatch = -1, indel = -1)
4. Rozwiązanie optymalne to takie, którego suma punktacji dla wszystkich pozycji jest maksymalna
5. Algorytm Needlemana-Wunscha znajduje je przy użyciu programowania dynamicznego, które buduje pełne rozwiązanie korzystając z rozwiązań cząstkowych

1.1 Działanie algorytmu na przykładzie

1. Znajdźmy dopasowanie sekwencji aminokwasów: $s = PRETTY$ oraz $t = PRTTEIN$

2. Utwórz macierz ze znakami z s w kolumnach oraz t w wierszach, dodając przy tym pustą kolumnę i pusty wiersz na początku:

		P	R	E	T	T	Y
P							
R							
T							
T							
E							
I							
N							

3. Wypełnij dodatkową kolumnę i wiersz malejącymi wartościami:

		P	R	E	T	T	Y
	0	-1	-2	-3	-4	-5	-6
P	-1						
R	-2						
T	-3						
T	-4						
E	-5						
I	-6						
N	-7						

4. Wypełniamy macierz od góry do dołu, od lewej do prawej

5. W puste pole wpisujemy maksimum z:

   - Komórki po lewej minus 1 (indel)
   - Komórki na górze minus 1 (indel)
   - Komórki w lewym-górnym rogu minus 1, jeżeli jest mismatch
   - Komórki w lewym-górnym rogu plus 1, jeżeli jest match

6. Wypełniamy pierwszy wiersz, na przecięciu P-P pojawia się 1, bo mamy match:

		P	R	E	T	T	Y
	0	-1	-2	-3	-4	-5	-6
P	-1	1	0	-1	-2	-3	-4
R	-2						
T	-3						
T	-4						
E	-5						
I	-6						
N	-7						

7. Kontynuujemy z dalszymi wierszami by otrzymać pełną macierz:

		P	R	E	T	T	Y
	0	-1	-2	-3	-4	-5	-6
P	-1	1	0	-1	-2	-3	-4
R	-2	0	2	1	0	-1	-2
T	-3	-1	1	1	2	1	0
T	-4	-2	0	0	2	3	2
E	-5	-3	-1	1	1	2	2
I	-6	-4	-2	0	0	1	1
N	-7	-5	-3	-1	-1	0	0

8. Należy teraz odczytać wynik, zaczynając od prawego dolnego rogu macierzy

9. Trzeba prześledzić wstecz kroki prowadzące do wartości w macierzy

10. Czasami można mieć do czynienia z niejednoznacznością, wówczas dopuszczamy dowolność

11. Przykładowa ścieżka odtwarzania wyniku z macierzy:

		P	R	E	T	T	Y
	0	-1	-2	-3	-4	-5	-6
P	-1	**1**	0	-1	-2	-3	-4
R	-2	0	**2**	**1**	0	-1	-2
T	-3	-1	1	1	**2**	1	0
T	-4	-2	0	0	2	**3**	2
E	-5	-3	-1	1	1	2	**2**
I	-6	-4	-2	0	0	1	**1**
N	-7	-5	-3	-1	-1	0	**0**

12. Odczytany wynik z powyższej ścieżki:

    (a) indel pionowo, $s' = -$, $t' = N$

(b) indel pionowo, $s' = --$, $t' = IN$

(c) mismatch, $s' = Y --$, $t' = EIN$

(d) match, $s' = TY --$, $t' = TEIN$

(e) match, $s' = TTY --$, $t' = TTEIN$

(f) indel poziomo, $s' = ETTY --$, $t' = -TTEIN$

(g) match, $s' = RETTY --$, $t' = R - TTEIN$

(h) match, $s' = PRETTY --$, $t' = PR - TTEIN$

13. Rezultat:
    `PRETTY--`
    `PR-TTEIN`

1.2 Zadanie

Napisz program, który otrzyma dwie sekwencje wejściowe i wypisze ich dopasowanie.

```
$ echo -e 'PRETTY\nPRTTEIN' | ./program
PRETTY--
PR_TTEIN
```

2 Algorytm Nussinov

Algorytm Nussinov został zaproponowany w 1980 roku do znalezienia struktury drugorzędowej RNA zawierającej maksymalną liczbę par.

2.1 Działanie algorytmu na przykładzie

1. Znajdźmy strukturę drugorzędową RNA o sekwencji: GAUUACA
2. Tworzymy macierz kwadratową zawierającą poszczególne litery zarówno w kolumnach, jak i w wierszach:

	G	A	U	U	A	C	A
G							
A							
U							
U							
A							
C							
A							

3. Wypełniamy przekątną oraz jej przesunięcie w dół zerami:

	G	A	U	U	A	C	A
G	0						
A	0	0					
U		0	0				
U			0	0			
A				0	0		
C					0	0	
A						0	0

4. W tym algorytmie, wypełniamy macierz do prawego górnego rogu

5. Dla każdego pola $m[i, j]$ wpisujemy maksimum z:

   - Wartość w lewym-dolnym rogu, jeżeli na przecięciu i-j nie mamy pary G-C ani A-U
   - Wartość w lewym-dolnym rogu plus 1, jeżeli na przecięciu i-j mamy parę G-C lub A-U
   - $m[i, k] + m[k + 1][j]$ dla każdego $i \leq k < j$

6. Wizualnie przedstawia się to tak, że aby wpisać wartość oznaczoną znakiem zapytania, musimy albo wybrać wartość z pola oznaczonego **z** albo sumę pól **t** i **t'**, **u** i **u'**, **v** i **v'**, **w** i **w'**, **x** i **x'** lub **y** i **y'**:

	G	A	U	U	A	C	A
G	t	u	v	w	x	y	?
A	0	0				z	t'
U		0	0				u'
U			0	0			v'
A				0	0		w'
C					0	0	x'
A						0	y'

7. Wypełnijmy zatem pierwszą serię wartości:

	G	A	U	U	A	C	A
G	0	0					
A	0	0	1				
U		0	0	0			
U			0	0	1		
A				0	0	0	
C					0	0	0
A						0	0

8. W kolejnej iteracji proszę zwrócić uwagę na pole $i = 1$, $j = 3$. Na przecięciu mamy G-U, którego nie uznajemy w tym algorytmie za parę. Natomiast szukając maksimum znajdujemy $m[1][1] + m[2][3] = 1$:

	G	A	U	U	A	C	A
G	0	0	1				
A	0	0	1	1			
U		0	0	0	1		
U			0	0	1	1	
A				0	0	0	0
C					0	0	0
A						0	0

9. W następnym kroku, wartość dla $i = 2$, $j = 5$ wynosi 2. Wynika to z maksimum równego $m[2][3] + m[4][5] = 2$:

	G	A	U	U	A	C	A
G	0	0	1	1			
A	0	0	1	1	2		
U		0	0	0	1	1	
U			0	0	1	1	1
A				0	0	0	0
C					0	0	0
A						0	0

10. Ostatecznie otrzymujemy macierz::

	G	A	U	U	A	C	A
G	0	0	1	1	2	3	3
A	0	0	1	1	2	2	2
U		0	0	0	1	1	2
U			0	0	1	1	1
A				0	0	0	0
C					0	0	0
A						0	0

11. Odczytując wynik, musimy odtworzyć kroki prowadzące do wartości w prawym-górnym rogu macierzy:

    - Wartość $m[1][7]$ pochodzi z $m[1][6] + m[7][7]$
    - Wartość $m[1][6]$ pochodzi z $m[2][5] + 1$ (para G-C)
    - Wartość $m[2][5]$ pochodzi z $m[2][3] + m[4][5]$
    - Wartość $m[2][3]$ pochodzi z $m[3][2] + 1$ (para A-U)

- Wartość $m[4][5]$ pochodzi z $m[5][4] + 1$ (para A-U)

12. Wynikowa struktura drugorzędowa to:
    `GAUUACA`
    `(()()).`

2.2 Zadanie

Napisz program, który otrzyma sekwencję RNA i wypisze strukturę drugorzędową w formacie dot-bracket.

```
$ echo GAUUACA | ./program
(()()).
```