Name:
Matrikelnummer:
Hauptfach:

Bioinformatik I
WS 2006

Klausur 27.02.2007

Aufgabe 1:

1.1 Definieren Sie die folgenden Begriffe:

a) Sequenzalignment (2)

b) lokales Alignment (1)

c) globales Alignment (1)

1.2 Gegeben sind die zwei folgenden Sequenzen:

```
Sequenz1: ACGT
Sequenz2: AGTC
```

a) Wie heisst der Algorithmus zur Suche des globalen optimalen Alignments ? (1)

b) Geben Sie den Pseudocode zur Berechnung des Scores des optimalen Alignments an. Verwenden Sie folgende Score Werte: Match=2, Mismatch=-1, Gappenalty=-3. (10)

c) Führen Sie den entsprechenden Algorithmus per Hand aus. Wie gross ist der Alignmentscore für das optimale Alignment? Geben Sie das Alignment an! (8)

1.3 Welchen Algorithmus/Methode verwendet man am Besten zum Alignieren von drei Sequenzen mit:

a) 10 Nukleotiden
b) 1000 Nukleotiden.

Begründen Sie Ihre Entscheidung. (4)

Aufgabe 2:

Gegeben ist folgendes Sequenzalignment (5 Sequenzen, 6 Positionen):

```
seq1    A T C A T G
seq2    A G A G G C
seq3    A T A A T G
seq4    A G A G A C
seq5    A G A A A G
```

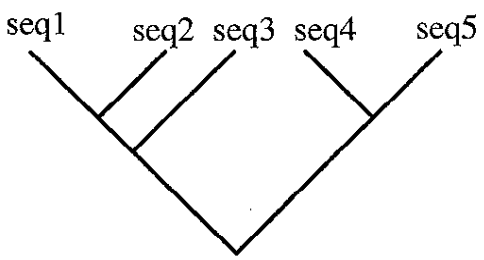
2.1 Geben Sie das Profil des Alignments an. (2)

2.2 Erstellen Sie aus dem Alignment a) den Keyword und b) den Suffix Tree (6).

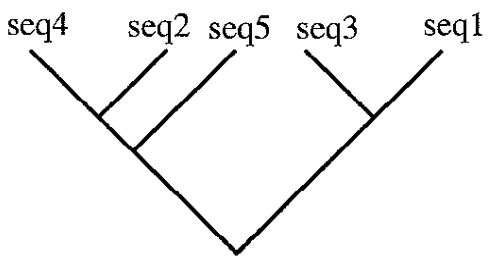
2.3 Begründen Sie, welcher der folgenden Bäume der Stammbaum der Sequenzen ist. Verwenden Sie den Algorithmus nach Fitch! Schreiben sie die Anzahl der Substitutionen pro Position und Baum in folgende Tabelle: (10)

	Pos1	Pos2	Pos3	Pos4	Pos5	Pos6
Baum1						
Baum2						

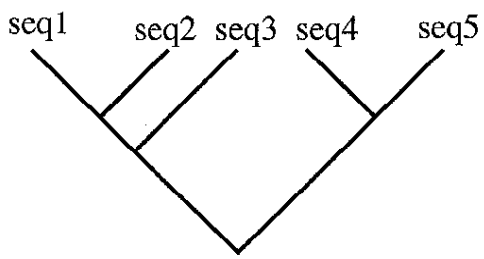
Baum1



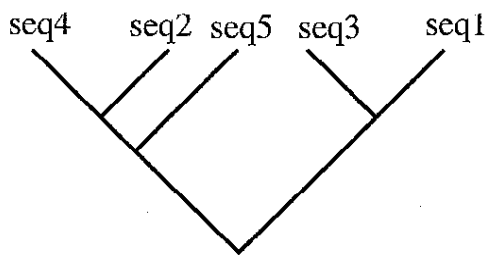
Baum2



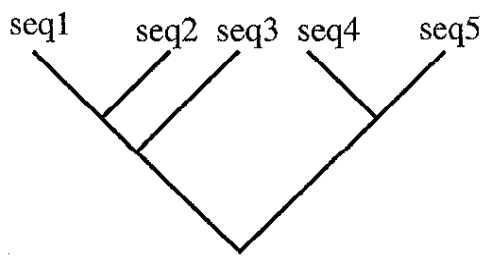
Baum1



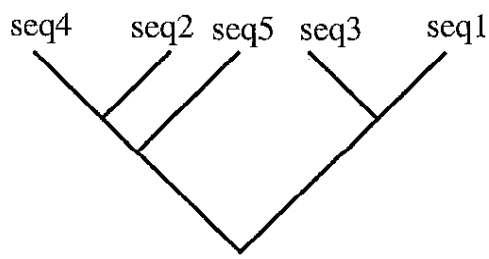
Baum2



Baum1



Baum2



Aufgabe 3:

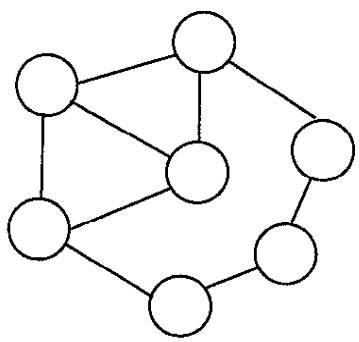
3.1 Was ist:

a) ein Euler Pfad ? (1)

b) ein Hamilton Pfad ? (1)

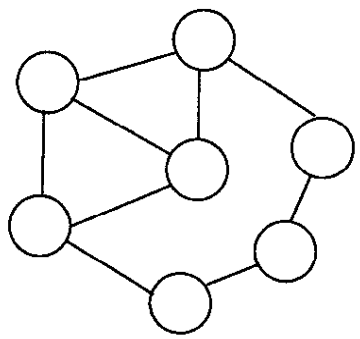
3.2 Gibt es in dem folgenden Graph einen (a) Euler Pfad (b) Hamilton Pfad? Begründen Sie Ihre Entscheidung. (2) Zeichnen Sie die Pfade in die entsprechenden Graphiken ein. (2)

a)



Euler Pfad

b)



Hamilton Pfad

3.3 a) Geben Sie den Überlappungsgraphen (Overlap-Graph) für folgende Sequenzen an. (6)

$S = \{TAG, GGT, GTC, AGG, TCA\}$

b) Finden Sie den kürzesten Superstring. (2)

Aufgabe 4:

Das Genom $\mathscr{G}$ der Steinlaus wird in zwei Laboren sequenziert. Dic Labore sequenzieren Sequenzen verschiedener Längen, wobei Labor A 80% und Labor B 20% des Genoms sequenziert. Die verschiedenen Sequenzen werden danach zu dem Gesamtgenom zusammengesetzt. Labor A hat unsauber gearbeitet, dadurch entstanden vermehrt Sequenzierfehler. Leider weiss man nicht mehr welches Labor welchen Bereich sequenziert hat. Aber man kennt die Nukleotidhäufigkeiten der Sequenzen:

Labor A: $\pi_A^A = 0.1, \quad \pi_C^A = 0.2, \quad \pi_G^A = 0.3, \quad \pi_T^A = 0.4$

Labor B: $\pi_A^B = 0.25, \quad \pi_C^B = 0.25, \quad \pi_G^B = 0.25, \quad \pi_T^B = 0.25$

a) Geben Sie eine graphische Formulierung des Problems, die Bereiche des Genoms den entsprechenden Laboren zuzuordnen. (5)

b) Geben Sie den Namen eines effizienten Algorithmus zur Lösung an. (1)

Aufgabe 6:

Rekonstrieren Sie durch hierarchisches Clustern aus dem folgendem Alignment einen Baum.

```
seq1    T T T A G A A G T T
seq2    C G T T G T A G G C
seq3    T G A T G T A G G C
seq4    T T T A A A A T G C
```

Verwenden Sie als Distanz zwischen den Sequenzen die Hammingdistanz. Tragen Sie die Distanzen in folgende Tabelle ein.

	seq1	seq2	seq3	seq4
seq1				
seq2				
seq3				
seq4				

Verwenden Sie als Distanz zwischen zwei Clustern den optimistischen Abstand, d.h. den kürzesten Abstand zwischen zwei beliebigen Elementen. (8)

Aufgabe 7:

a) Beschreiben Sie kurz, was in den Abschnitten 1-3 des folgenden Perl-Codes passiert.(4)

b) Welcher Algorithmus wird damit implementiert? (4)

```
############################   ABSCHNITT 1    ############################

#!/usr/bin/perl
use strict;

my $states = 3;
my @seq    = ('A','C','G','G','T' ,'A');
my @v;
my @alpha  = ('A','C','T','G');

my @t;
for (my $a=1;$a<=$states;$a++) {
    for (my $b=1;$b<=$states;$b++) {
       $t[$a][$b] = 1/$states;
    }
}

my @e;
for (my $a=1;$a<=$states;$a++) {
    foreach my $c (@alpha) {
       $e[$a]{$c} = 1/@alpha;
    }
}
############################   ABSCHNITT 2    ############################

$v[0][0] = 1;
for (my $k = 1; $k <= $states; $k++) {
    $v[$k][0] = 1;
}
############################   ABSCHNITT 3    ############################

for (my $i=1; $i <= @seq ; $i++) {
    for (my $j=1; $j <= $states; $j++) {
        my $max = -10000000000;
        for (my $k = 1; $k <= $states; $k++) {
            my $result = $e[$k]{$seq[$i-1]} * $v[$k][$i-1] * $t[$k][$j];
            if ( $result > $max ) {
            $max = $result;
            }
        }
        $v[$j][$i] = $max;
    }
}
```

Aufgabe 8:

a) Was ist ein Greedy Algorithmus? (1)

b) Was ist eine Branch-and-Bound Methode? (2)

c) Was ist dynamische Programmierung? (2)

Aufgabe 9:

Was ist die Performance Garantie? (4)

Aufgabe 10:

Was ist der Unterschied zwischen hierarchischem Clustern und Neighbor Joining? (2)