Πανεπιστήμιο Πειραιώς
Τμήμα Πληροφορικής
Έτος: 2022 - 2023

Μάθημα:
«ΒΙΟΠΛΗΡΟΦΟΡΙΚΗ»
Εργασία: Υπολογιστική Εργασία Μαθήματος
Εξάμηνο: 6ο

<u>Ομάδα εργασίας:</u>

Αιμιλιανός Κουρπάς-Δανάς Π20100,

Αναστάσιος Μελαχροινούδης Π20124

Ημερομηνία παράδοσης : 30.06.2023

Περιεχόμενα

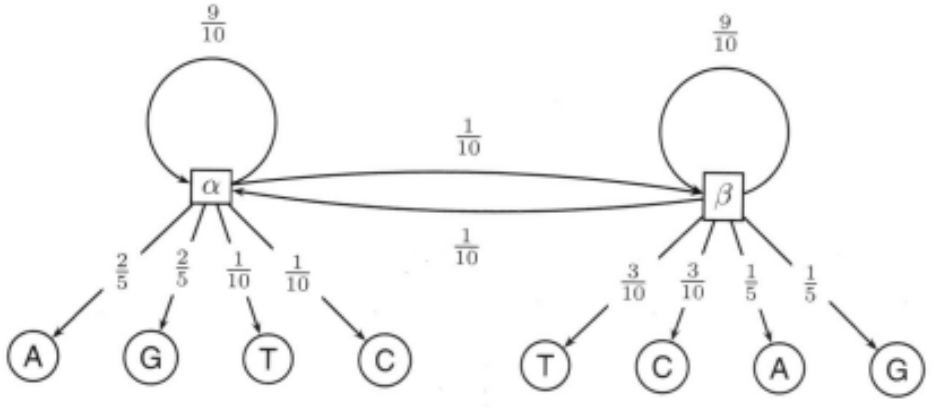
Σημείωση

Τα προγράμματα γράφτηκαν στο προγραμματιστικό περιβάλλον VScode.
Στον κώδικα υπάρχουν περιεκτικά σχόλια ουσίας για την καλύτερη επεξήγησή του.

ΘΕΜΑ 1ο:

1.1 Εκφώνηση:

Στο παρακάτω σχήμα (11.7) φαίνεται ένα HMM με δύο καταστάσεις α και β. Όταν το HMM βρίσκεται στην κατάσταση α, έχει μεγαλύτερη πιθανότητα να εκπέμψει πουρίνες (A και G) . Όταν βρίσκεται στην κατάσταση β έχει μεγαλύτερη πιθανότητα να εκπέμψει πυριμιδίνες (C και T). Αποκωδικοποιήστε την πιο πιθανή ακολουθία των καταστάσεων (α/β) για την αλληλουχία **GGCT.** Χρησιμοποιήστε λογαριθμικές βαθμολογίες αντί για κανονικές βαθμολογίες πιθανοτήτων .



Σχήμα 11.7 Το HMM που περιγράφεται στο Πρόβλημα 11.4.

1.2 Λύση:

1.2.1 Αλγοριθμική Περιγραφή:

Γνωρίζουμε ήδη τα εξής από την εκφώνηση:

- Υπάρχουν 2 καταστάσεις: α, β
- Οι 2 καταστάσεις περιέχουν τις παρατηρήσεις A, G, T, C

Από το σχήμα μπορούμε να εξάγουμε τα εξής δεδομένα:

- Πίνακας πιθανότητας αρχικής κατάστασης Π: [0.5 , 0.5]
  Με λογαριθμική βαθμολογία: [-1 , -1]
- Πίνακας πιθανότητας μετάβασης από κατάσταση σε κατάσταση
  A: [0.9 , 0.1 ; 0.1, 0.9] (αναλυτικά: [p(α/α), p(β/α) ; p(α/β) p(β/β)]
  Με λογαριθμική βαθμολογία:
  [[-0.15200309 ,-3.32192809], [-3.32192809, -0.15200309]])
- Πίνακας εκπομπής συμβόλων πιθανών καταστάσεων
  B:[0.4 ,0.2],[0.4, 0.2],[0.1, 0.3], [0.1, 0.3]

  [p(A/α), p(G/α), p(T/α), p(C/α) ; p(A/β), p(G/β), p(T/β), p(C/β)]

  Με λογαριθμική βαθμολογία:
  [-1.32192809 -2.32192809], [-1.32192809 -2.32192809],

  [-3.32192809 -1.73696559], [-3.32192809 -1.73696559]]

Για την επίλυση του προβλήματος θα χρησιμοποιήσουμε τον αλγόριθμο του Viterbi ο οποίος στηρίζεται στις αρχές του δυναμικού προγραμματισμού. Στον Viterbi, αναζητούμε το πιθανότερο μονοπάτι που οδηγεί σε μια συγκεκριμένη κατάσταση. Αν και υπάρχουν πολλά πιθανά μονοπάτια, θέλουμε να επιλέξουμε αυτό με τη μεγαλύτερη πιθανότητα.

Για να γίνουν οι υπολογισμοί μας πιο αποδοτικοί, χρησιμοποιούμε τον λογάριθμο των πιθανοτήτων αντί να πολλαπλασιάζουμε τις πιθανότητες. Αυτό επιτυγχάνεται υπολογίζοντας το άθροισμα των λογαρίθμων πιθανοτήτων αντί για το γινόμενό τους. Με τη χρήση των

λογαρίθμων, μπορούμε να αποφύγουμε προβλήματα που μπορεί να προκύψουν από την πολλαπλασιαστική πράξη μικρών πιθανοτήτων.

Έπειτα από τον αλγόριθμο Viterbi έχουμε τις εξής τιμές:

α	β
-2.32192809	-3.32192809
-3.79585928	-5.79585928
-7.26979047	-7.68482797
-10.74372166	-9.57379666

Τέλος χρησιμοποιούμε την μέθοδο οπισθοδρόμησης μέχρι και το πρώτο σύμβολο της ακολουθίας για να βρούμε το καλύτερο μονοπάτι το οποίο θα είναι και η πιο πιθανή ακολουθία καταστάσεων.

1.2.2 Υλοποίηση Προγράμματος

Γλώσσα Υλοποίησης: Python

Βιβλιοθήκη: numpy

Μεταβλητές:

- **emissions**: Λίστα που περιέχει τους διαφορετικούς χαρακτηρισμούς εκπομπής.
- **states**: Λίστα που περιέχει τις καταστάσεις.
- **observation**: τύπου string που ειναι η αλληλουχια.

- **transition prob**: Πίνακας numpy που αναπαριστά τις πιθανότητες μετάβασης από μια κατάσταση σε μια άλλη.
- **emission prob:** Πίνακας numpy που αναπαριστά τις πιθανότητες εκπομπής των παρατηρήσεων από τις καταστάσεις
- **initial prob:** Πίνακας numpy που αναπαριστά τις αρχικές πιθανότητες για κάθε κατάσταση.
- **scores**: Πίνακας numpy που περιέχει τις βαθμολογίες για κάθε κατάσταση και παρατήρηση.
- **backpointers**: Πίνακας numpy που περιέχει τους δείκτες για την κατασκευή της καλύτερης διαδρομής.
- **best path**: Μια λίστα που περιέχει την καλύτερη διαδρομή

Συνάρτηση:

➢ **Viterbi**

Η συνάρτηση Viterbi υλοποιεί τον αλγόριθμο Viterbi για τον υπολογισμό της βέλτιστης διαδρομής στο HMM. Αρχικά , αρχικοποιείται ένας πίνακας για τις βαθμολογίες και ένας πίνακας για τους δείκτες προς τα πίσω. Στη συνέχεια, υπολογίζονται οι βαθμολογίες για κάθε παρατήρηση, βασιζόμενες στις προηγούμενες βαθμολογίες και τις πιθανότητες μετάβασης. Εντοπίζονται οι μέγιστες βαθμολογίες και αποθηκεύονται οι αντίστοιχοι δείκτες προς τα πίσω. Τέλος, ανακατασκευάζεται η καλύτερη διαδρομή από τις αποθηκευμένες πληροφορίες. Η συνάρτηση επιστρέφει την καλύτερη διαδρομή και τις βαθμολογίες.

Ορίσματα

- observation
- emissions
- states.
- A: Ο πίνακας μετάβασης.
- B: Ο πίνακας εκπομπής.
- P: Ο αρχικός πίνακας πιθανοτήτων.

1.3 Το τεχνικό κομμάτι

Οδηγίες για εκτέλεση του προγράμματος:

- Χρειάζεται να έχουμε εγκατεστημένη την python
- Χρειάζεται να έχουμε εγκατεστημένη την numpy οπου γινεται με την εντολη στο terminal pip install numpy
- Βρισκουμε το terminal στο path που βρίσκεται το 11_4.py , και γράφετε python3 11_4.py

1.4 Παράδειγμα

```
Scores:

Step 1: a: -2.322, b: -3.322
Step 2: a: -3.796, b: -5.796
Step 3: a: -7.270, b: -7.685
Step 4: a: -10.744, b: -9.574

Best_path:

Step 1: b
Step 2: b
Step 3: b
Step 4: b

The best path for  GGCT is: b b b b
```

Την βιβλιογραφία για το πρώτο θέμα θα το βρείτε στο τέλος του τρίτου θέματος.

ΘΕΜΑ 2ο:

2.1 Εκφώνηση:

Δύο παίκτες παίζουν το παρακάτω παιχνίδι με δύο <<χρωμοσώματα>> που έχουν μήκος n και m νουκλεοτιδίων αντίστοιχα. Σε κάθε γύρο του παιχνιδιού, ένας παίκτης μπορεί να καταστρέψει ένα από τα χρωμοσώματα και να διαχωρίσει το άλλο σε δύο μη κενά τμήματα. Για παράδειγμα , ο πρώτος παίχτης μπορεί να καταστρέψει ένα χρωμόσωμα μήκους n και να διαχωρίσει ένα άλλο χρωμόσωμα σε δύο χρωμοσώματα με μήκη m/3 και m- (m/3). Ο παίχτης που διαγράφει το τελευταίο νουκλεοτίδιο κερδίζει. Ποιος θα κερδίσει; Περιγράψτε τη νικηφόρα στρατηγική για όλες τις τιμές των n και m.

2.2 Λύση:

2.2.1 Αλγοριθμική Περιγραφή

Αρχικά, διαβάζουμε τις 2 ακολουθίες αλληλουχιών τις οποίες επιλέγει ο χρήστης, οι οποίες βρίσκονται αποθηκευμένες σε fna αρχεία. Ορίζουμε τις λίστες canWin και winning με σκοπό στην συνέχεια του προγράμματος να τοποθετήσουμε την νικηφόρα ακολουθία στην winning καθώς και το πως θα πρέπει να διαιρεθούν για να κερδίσουν.
Μέσα σε for loop συγκρίνουμε τις 2 ακολουθίες. Αν επιστραφεί False, σημαίνει ότι ο επόμενος παίκτης θα χάσει οτιδήποτε και να επιλέξει, άρα εμείς θα κερδίσουμε. Αν συμβεί αυτό, αποθηκεύουμε στην λίστα winning τη θέση που βρισκόμαστε. Επιπλέον, ενημερώνουμε στην λίστα canWin ότι στο σημείο αυτό ο παίκτης μπορεί να κερδίσει.
Ελέγχουμε τα στοιχεία στις θέσεις μήκους των δύο αλληλουχιών μας, αν είναι True,θα κερδίσει ο παίχτης Α, αλλιώς ο Β.
Τέλος, τυπώνουμε τα winning splits.

2.2.2 Υλοποίηση Προγράμματος

Γλώσσα Υλοποίησης: Python

Μεταβλητές:

- **seq**: Αναπαριστά τη λίστα νουκλεοτιδίων
- **seq1**: Αναπαριστά την πρώτη ακολουθία.
- **seq2**: Αναπαριστά την δεύτερη ακολουθία.
- **canWin**: Αναπαριστά μια λίστα λογικών τιμών
- **winner**: Αναπαριστά τον νικητή του παιχνιδιού

Συναρτήσεις:

➢ Load Seq

Χρησιμοποιείται για να φορτώσει μια ακολουθία από ένα αρχείο κειμένου. Όταν καλείται, εμφανίζει ένα μήνυμα που υποδεικνύει το όνομα του αρχείου που φορτώνεται. Στη συνέχεια, η συνάρτηση ανοίγει το αρχείο με το όνομα που δόθηκε στην παράμετρο `txtName`, χρησιμοποιώντας την ανάγνωση (read mode).

➢ Validate Input

Χρησιμοποιείται για τον έλεγχο της εισόδου που δίνει ο χρήστης.Η συνάρτηση επαναλαμβάνεται συνεχώς,μέχρι να πληρεί του όρους. Όταν ο χρήστης εισάγει κάτι, αποθηκεύεται στη μεταβλητή `user_input`. Στη συνέχεια, ελέγχεται αν η είσοδος του χρήστη βρίσκεται στις αποδεκτές τιμές και αν διαφέρει από την ακολουθία `seq`. Αν ικανοποιούνται αμφότερες οι συνθήκες, τότε η συνάρτηση επιστρέφει την είσοδο του χρήστη.

➢ Game

Προσομοιώνει το παιχνίδι με τις δύο ακολουθίες. Η πρώτη ενέργεια είναι η φόρτωση των αρχείων χρησιμοποιώντας τη συνάρτηση load Seq,`.Έπειτα με μια εμφωλευμένη επανάληψη βρίσκει τον winner και τα splits

2.3 Το τεχνικό κομμάτι

Οδηγίες για εκτέλεση του προγράμματος:

- Χρειάζεται να έχουμε εγκατεστημένη την python
- Βρισκουμε το terminal στο path που βρίσκεται το 6_12.py , και γράφετε python3 6_12.py

2.4 Παράδειγμα

θα δείξουμε τις πρωτες γραμμές

```
Choose the first Sequence (1-brain, 2-liver, 3-muscle): 1
Choose the second Sequence (1-brain, 2-liver, 3-muscle): 3
Loading the file brain.fna
Loading the file muscle.fna
The winner is A
The winning splits:
[2, 2]
[2, 3]
[3, 3]
[2, 7]
[2, 8]
[3, 8]
[2, 12]
[2, 13]
[3, 13]
[2, 17]
[2, 18]
[3, 18]
[2, 22]
[2, 23]
[3, 23]
[2, 27]
[2, 28]
[3, 28]
[2, 32]
[2, 33]
[3, 33]
[2, 37]
[2, 38]
[3, 38]
[2, 42]
[2, 43]
[3, 43]
```

Την βιβλιογραφία για το δεύτερο θέμα θα το βρείτε στο τέλος του τρίτου θέματος.

ΘΕΜΑ 3ο:

3.1 Εκφώνηση:

Δύο παίκτες παίζουν το εξής παιχνίδι με δύο αλληλουχίες που έχουν μήκος n και m νουκλεοτίδια αντίστοιχα. Σε κάθε γύρο του παιχνιδιού, ένας παίκτης μπορεί να αφαιρέσει δύο νουκλεοτίδια από τη μία αλληλουχία (είτε την πρώτη είτε τη δεύτερη) και ένα νουκλεοτίδιο από την άλλη. Ο παίκτης που δεν μπορεί να κάνει κίνηση κερδίζει. Ποιος θα κερδίσει; Περιγράψτε τη νικηφόρα στρατηγική για όλες τις τιμές των n και m.

3.2 Λύση:

3.2.1 Αλγοριθμική Περιγραφή

Έχουμε δύο σύνολα ενζύμων που ονομάζονται "brain" και "liver" με διαφορετικό μήκος: το "brain" έχει 39253 νουκλεοτίδια και το "liver" έχει 49971 νουκλεοτίδια. Αυτά τα σύνολα ενζύμων είναι αποθηκευμένα στα αρχεία "brain.fna" και "liver.fna". Το παιχνίδι παίζεται από δύο παίκτες χρησιμοποιώντας αυτές τις ακολουθίες ενζύμων. Οι παίκτες παίρνουν σειρά να παίξουν. Σε κάθε γύρο, ο παίκτης επιλέγει μια ακολουθία και αφαιρεί δύο στοιχεία από αυτήν, καθώς και ένα στοιχείο από την άλλη ακολουθία. Αυτή η διαδικασία συνεχίζεται μέχρι ένας παίκτης να μην μπορεί να πραγματοποιήσει άλλη κίνηση. Ο παίκτης που δεν μπορεί να πραγματοποιήσει κίνηση κηρύσσεται νικητής του παιχνιδιού.

Έτσι, ας θεωρήσουμε ότι έχουμε την ακολουθία m και την ακολουθία n. Ο πρώτος παίκτης βρίσκεται σε ευνοϊκή θέση όταν παίρνει τις ακολουθίες ως εξής:

- $m = 3x$ και $n \geq m$, για $x \in N$.
- $m = n = 3k + 1$, για $x \in N$.

Παραδείγματα:

- [m, n] = Για [2 , 0] ή [2, 5] ή [2, 4], ο πρώτος παίκτης κερδίζει, αφού είναι τελικές καταστάσεις. Επίσης, για [5, 5] ή [5, 6] ή [5, 8] ο πρώτος παίκτης μεγιστοποιεί τις πιθανότητες να κερδίσει.
- [m, n] = Για [1, 1], ο πρώτος παίκτης κερδίζει. Επίσης, για [6, 6] ή [9, 9] ή [12, 12], ο πρώτος παίκτης εξίσου μεγιστοποιεί τις πιθανότητες να κερδίσει.

3.2.2 Υλοποίηση Προγράμματος

Γλώσσα Υλοποίησης: Python

Βιβλιοθήκη: Biopython
Η βιβλιοθήκη Biopython χρησιμοποιείται για να διαβάσει αλληλουχίες από τα αρχεία εισόδου "liver.fna" και "brain.fna". Η συνάρτηση `SeqIO.parse` από το Biopython χρησιμοποιείται ειδικά για αυτόν τον σκοπό.

Μεταβλητές:

- **liver sequences**: Μια λίστα που περιέχει τις ακολουθίες που προέρχονται από το αρχείο "liver.fna".
- **brain sequences**: Μια λίστα που περιέχει τις ακολουθίες που προέρχονται από το αρχείο "brain.fna".
- **liver_seq**: Η ακολουθία του ήπατος (liver) που προέρχεται από το αρχείο "liver.fna".
- **counter**: Ένας μετρητής που καταγράφει τον αριθμό των γύρων που έχουν παιχθεί.
- **current_player**: Η ένδειξη του τρέχοντος παίκτη που παίζει σε κάθε γύρο.
- **winner**: Η ένδειξη του παίκτη που κερδίζει το παιχνίδι.

Συνάρτηση:

➢ Make Move

Προσομοιώνει έναν γύρο παιχνιδιού όπου δύο παίκτες παίζουν με τη σειρά τους. Το παιχνίδι αφορά δύο ακολουθίες, που αναπαρίστανται από τα μήκη των ακολουθιών του ήπατος και του εγκεφάλου (`liver_len` και `brain_len`).

3.3 Το τεχνικό κομμάτι

Οδηγίες για εκτέλεση του προγράμματος:

- Χρειάζεται να έχουμε εγκατεστημένη την python
- Χρειάζεται να έχουμε εγκατεστημένη την Biopython οπου γινεται με την εντολη στο terminal `pip install biopython`
- Βρισκουμε το terminal στο path που βρίσκεται το 6_14.py , και γράφετε `python3 6_14.py`

3.4 Παράδειγμα

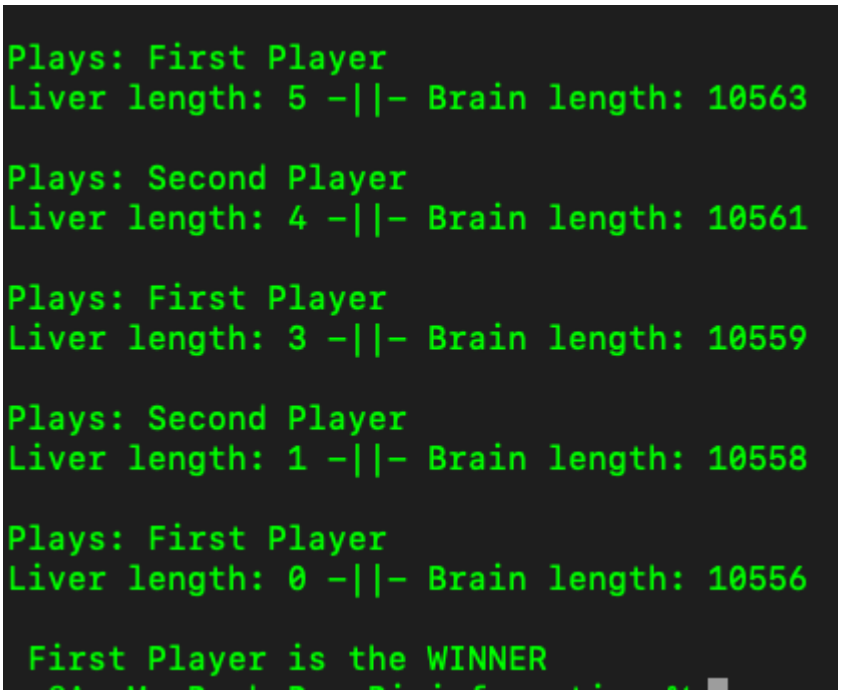
Αρχή

```
Plays: First Player
Liver length: 39265 -||- Brain length: 49927

Plays: Second Player
Liver length: 39263 -||- Brain length: 49926

Plays: First Player
Liver length: 39261 -||- Brain length: 49925
```

Τέλος



Όπως βλέπουμε ο πρώτος παίκτης δεν μπορεί να κάνει κάποια κίνηση, άρα είναι ο νικητής

Βιβλιογραφία Θεμάτων:

Θέμα 1ο:

- Numpy Documentation : https://numpy.org/doc
- Viterbi Algorithm : https://en.wikipedia.org/wiki/Viterbi_algorithm
- Σημειώσεις Βιοπληροφορικής, Πανεπιστήμιο Πειραιώς Πικράκης Άγγελος
- Εισαγωγή στους αλγορίθμους Βιοπληροφορικής
  Neil. C Jones, Pavel A. Pevzner

Θέμα 2ο:

- Σημειώσεις Βιοπληροφορικής, Πανεπιστήμιο Πειραιώς Πικράκης Άγγελος
- Εισαγωγή στους αλγορίθμους Βιοπληροφορικής
  Neil. C Jones, Pavel A. Pevzner

Θέμα 3ο:

- Biopython : https://biopython.org/

- Σημειώσεις Βιοπληροφορικής, Πανεπιστήμιο Πειραιώς Πικράκης Άγγελος
- Εισαγωγή στους αλγορίθμους Βιοπληροφορικής
  Neil. C Jones, Pavel A. Pevzner