ΠΑΝΕΠΙΣΤΗΜΙΟ ΠΕΙΡΑΙΩΣ

Τμήμα Πληροφορικής

A blue and red logo

AI-generated content may be incorrect.

Εργασία Μαθήματος «Βιοπληροφορική»

|  |  |
| --- | --- |
| Τίτλος | ***Υπολογιστική Εξαγωγή Μέτρων Κεντρικότητας και Πρόγνωση Ακμών*** |
| Όνομα φοιτητή – Αρ. Μητρώου | Λαζαρίδης Χρήστος-Λάζαρος - Π22083 |
| Καλογερόπουλος Αθανάσιος – Π22223 |
| Ημερομηνία παράδοσης |  |

ΠΙΝΑΚΑΣ ΠΕΡΙΕΧΟΜΕΝΩΝ

[1. Θέμα 1 3](#_Toc196685420)

[a. Υλοποίηση 3](#_Toc196685421)

[b. Εκτέλεση 4](#_Toc196685422)

[2. Θέμα 2 5](#_Toc196685423)

[a. Σύσταση του αλγόριθμου Πολλαπλής Στοίχησης 5](#_Toc196685424)

[b. Σύγκριση της μεθόδου με τη μέθοδο clustal (bonus) 8](#_Toc196685425)

[c. Εκτέλεση 8](#_Toc196685426)

[3. Θέμα 3 9](#_Toc196685427)

[a. Ισχυρές Καταστάσεις 9](#_Toc196685428)

[b. Ζητήματα Markovιανής Μοντελοποίησης 10](#_Toc196685429)

[c. Fine Tuning (εκπαίδευση) 11](#_Toc196685430)

[4. Θέμα 4 12](#_Toc196685431)

[5. Σχόλια υλοποίησης 12](#_Toc196685432)

[6. Βιβλιογραφία 12](#_Toc196685433)

ΠΙΝΑΚΑΣ ΠΙΝΑΚΩΝ

[Table 1: Παράδειγμα πίνακα συχνοτήτων 6](#_Toc196685434)

[Table 2: Αρχικό lookup table 7](#_Toc196685435)

[Table 3: Αποτέλεσμα πολλαπλής στοίχησης για το dataset A 9](#_Toc196685436)

# Θέμα 1

## Υλοποίηση

Στο θέμα 1 μας ζητείτε να δημιοουργήσουμε 3 dataset με συγκεκριμένο τρόπο.

Μας δώθηκαν 4 συγκεκριμένα patterns του αλφάβητου

* Pattern 1 = *“ΑΤΤΑGA“*
* Pattern 2 = *“ΑCGCΑTΤT“*
* Pattern 3 =*”ΑCGCΑTΤT“*
* Pattern 4 =*”ATTTCAGT“*

Από τα οποία πρέπει να δημιουργήσουμε 100 συμβολοσειρές , τις οποίες τις δημιουργούμε μεταλλάσοντας και επαυξάνοντας τα παραπάνω patterns με τον εξής τρόπο

* Κάθε sequence αρχίζει με 1 έως 3 τυχαία σύμβολα του αλφάβητου
* Για κάθε καινούργιο sequence που θα δημιουργηθεί επιλέγουμε τα 4 patterns με τη σειρά από το πρώτο στο τέταρτο, μια φορά το καθένα και αλλάζουμε από κανένα μέχρι δύο σύμβολα σε τυχαίες θέσεις, είτε με ένα τυχαίο επιλεγμένα σύμβολο του αλφάβητου, είτε με το κενό, διαγράφοντας το σύμβολο, και τελικά προσθέτουμε το αποτέλεσμα της μετάλλαξης στη μέχρι τώρα συμβολοσειρά, μέχρι να έχουμε προσθέσει το αποτέλεσμα της τυχαίας μετάλλαξης και των 4 patterns.
* Τέλος κάθε sequence θα τελειώνει με 1 έως 2 τυχαία σύμβολα του αλφάβητου.

Η διαδικασία επαναλαμβάνεται 100 φορές και τα 100 sequences που δημιουργούνται θα εισαχθούν με τυχαίο τρόπο σε 3 datasets

* Dataset A: 10 sequences
* Dataset B: 70 sequences
* Dataset C: 20 sequences

Για τις ανάγκες του θέματος κάνουμε χρήση των συναρτήσεων

* **mutate\_pattern**: Εφαρμόζει έως *max\_mutations* τυχαίες αλλαγές (substitution ή deletion) στο δοθέν pattern.
  + Arguments: *pattern* (σε μορφή string) και *max\_mutations,* δηλαδή ένας ακέραιος με τον αριθμό των μεταλλάξεων (default 2)
  + Output: Το μεταλλαγμένο pattern σε μορφή string
* **create\_random\_sequence:** δημιουργεί τυχαία ακολουθία από *min\_len* μέχρι *max\_len*
  + Arguments: 2 ακέραιοι *min\_len* και *max\_len* που δηλώνουν το ελάχιστο και το μέγιστο επιτρεπτό μήκος της συμβολοσειράς
  + Output: Η τυχαία παραγώμενη συμβολοσειρά σε μορφή string

Ακόμη χρησιμοποιούνται και 2 μέθοδοι **save\_fasta** που σώζουν τα παραγώμενο datasets σε αρχεία τύπου fasta. Για το θέμα δημιουργήθηκε το script **thema\_1.py**

## Εκτέλεση

Η εκτέλεση του προγράμματος μας δίνει αυτά τα αποτελέσματα με αυτά τα στατιστικά:

* Dataset A:
  + Αριθμός Ακολουθιών: 10
  + Μέσο μήκος ακολουθίας: 33.0
  + Πρώτη ακολουθία: *“ACGATTAGGACCCTTTACGACTAAAATTTCACTCC” (μήκος = 35)*
  + Τελευταία ακολουθία *“GAATTAGAACGCATTTAGGACTCAAATTTCAGTC” (μήκος = 34)*
* Dataset Β:
  + Αριθμός Ακολουθιών: 70
  + Μέσο μήκος ακολουθίας: 32.79
  + Πρώτη ακολουθία: *“TAATTAGAACCATCTAAGACTCAAATTTCGTG” (μήκος = 32)*
  + Τελευταία ακολουθία *“TAAGGTAGAACGCATATAGGTCTCACATTTCATCG” (μήκος = 35)*
* Dataset C:
  + Αριθμός Ακολουθιών: 20
  + Μέσο μήκος ακολουθίας: 32.10
  + Πρώτη ακολουθία: *“CCTATTAGACGCATTTAGGACGCAAATTTCATGC” (μήκος = 34)*
  + Τελευταία ακολουθία *“AAATGAGACGCTTTTAGGACTCAATTTCAGC” (μήκος = 31)*

# Θέμα 2

## Σύσταση του αλγόριθμου Πολλαπλής Στοίχησης

Και τα 2 ΑΜ της ομάδας καταλήγουν σε περριτό αριθμό, επομένως για τις ανάγκες του θέματος α =2.

Θα πρέπει να δημιουργηθεί αλγόριθμος πολλαπλής στοίχισης βασισμένος σε έναν αλγόριθμο global alignment (καθολικής στοίχησης) ο οποίος σύμφωνα με την εκφώνηση θα οριστεί ως εξής

* Η οριζόντια και η κάθετη μετάβαση στο πλέγμα δυναμικού προγραμματισμού θα τιμωρείτε με gap penalty -2
* Η τοπική ομοιότητα θα προσθέτει score +1 (αρχικά, θα μιλήσουμε για το lookup table αργότερα)
* Η τοπική ανομοιότητα θα επιβάλει penalty -1
* Δυνατοί πρόγονοι του κόμβου (i,j) θα είναι οι κόμβοι (i-1, j-1), (i, j-1) και (i-1, j) δηλαδή, ο από κάτω του, ο αριστερά του και ο κάτω και αριστέρα διαγώνιος
* Προφανώς ο αλγόριθμος αποτελεί αλγόριθμο μεγιστοποίησης με βάση την ομοιότητα 2 συμβολοσειρών, οπότε επιλέγουμε πάντα τη μετάβαση που μεγιστοποιεί το σκορ
* Η στοίχηση των 2 προφίλ γίνεται μέσο backtracking με τον εξής τρόπο:
  + Διαγραφή για κάθετη κίνηση (προσθήκη “\_” στη δεύτερη ακολουθία)
  + Εισαγωγή για οριζόντια κίνηση (προσθήκη “\_” στη πρώτη ακολουθία)
  + Όταν εφαρμώσαμε διαγώνια κίνηση τότε μιλάμε για αντιστοίχηση (δεν μας ενδιαφέρει αν τα σύμβολα είναι ίδια)

Για τη διαδικασία της πολλαπλής στοίχισης θα υιοθετηθεί η διαδικασία hierarchical clustering η οποία παραδόθηκε στην 7η διάλεξη του μαθήματος και:

* Για κάθε εποχή (μέχρι να μείνουν με 2 sequences)
  + Δημιουργούμε ένα πίνακα ανομοιοτήτων
  + Επιλέγουμε το το λιγότερο ανόμοιο ζευγάρι
  + Εφαρμόζουμε τον αλγόριθμο καθολικής στοίχισης στο ζευγάρι
  + Φτιάχνουμε την ακολουθία συνένεσης (όχι στη τελευταία εποχή)
  + Αντικαθιστούμε το ζευγάρι με την ακολουθία συνένεσης (όχι στη τελευταία εποχή)
* Κάνουμε unfold τα αποτελέσματα

Για τη δημιουργία του πίνακα ανομοιοτήτων θα χρησιμοποιήσουμε τον αλγόριθμο k-mer ο οποίος διδάχθηκε στην 6η διάλεξη του μαθήματος και ορίζεται ως εξής

* Ορίζουμε κάποιο k
* Με τεχνική συρρώμενου παραθύρου βρίσκουμε όλες τις πιθανές k-αδες και μετράμε τη συχνότητα τους, φτιάχνοντας ένα πίνακα συχνοτήτων ο οποίος θα μπορούσε να μοιάζει ως εξής:

Παράδειγμα: 2 συμβολοσειρές

* + Seq1: “ATGCTAGC”
  + Seq2: “ATGCGC”

Ο πίνακας συχνοτήτων θα ήταν

|  |  |  |
| --- | --- | --- |
|  | Seq1 | Seq2 |
| ATG | 1 | 1 |
| TGC | 1 | 1 |
| GCT | 1 | 0 |
| CTA | 1 | 0 |
| TAG | 1 | 0 |
| AGC | 1 | 0 |
| GCG | 0 | 1 |
| CGC | 0 | 1 |

Table 1: Παράδειγμα πίνακα συχνοτήτων

Από εκεί και πέρα μπορούμε να διανυσματοποιήσουμε το παραπάνω πίνακα για κάθε sequence και να υπολογίσουμε της απόστασης συνημιτόνου

Ορίζουμε την απόσταση συνημιτόνου (cosine distance) ως εξής

όπου:

το εσωτερικό γινόμενο των a και b

το ευκλείδιο μήκος ενός διανύσματος

Τελικά θα καταλήξω με ένα άνω τριγωνικό πίνακα, του οποίου το κελί (i , j) αποτελεί το

* Για την ακολουθία συνένεσης θα χρησιμοποιήσουμε την εξής διαδικασία στις συμβολοσειρές μετά το alignment (το αποτέλεσμα δηλαδή του backtracking στο πλέγμα δυναμικού προγραμματισμού)
  + Αν τότε
  + Αν τότε (και αντίστοιχα)
  + Αν τότε ειςάγω ένα καινούργιο σύμβολο (πχ X) το οποίο ορίζεται ως:

Τα scores αυτά ανανεώνουν το lookup table,έτσω ώστε το X θα προσθέσει score = 0.5 αν βρεθεί με τα ή στον αλγόριθμο στοίχισης. Προφανώς στο lookup table οι βαθμολογίες αρχικά είναι 1 για όμοια σύμβολα και 0 για ανόμοια σύμβολα, δηλαδή

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | A | C | G | T |
| A | 1 | 0 | 0 | 0 |
| C | 0 | 1 | 0 | 0 |
| G | 0 | 0 | 1 | 0 |
| T | 0 | 0 | 0 | 1 |

Table 2: Αρχικό lookup table

Τα scores για κάθε σύμβολο προφανώς και μπορεί να αλλάξουν, αν ένα σύμβολο χρειάζεται να αντιπροσωπεύει περισσότερα σύμβολα, με τα scores να αλλάζουν με βάση το frequency

* Τέλος απαιτείτε να κάνουμε unfold το αποτέλεσμα της πολλαπλής στοίχισης. Ουσιαστικά αυτό είναι το αποτέλεσμα όλων των στοιχισμένων συμβολοσειρών, δηλαδή κάθε στοιχισμένη ακολουθία από την οποία φτιάχνουμε κάποια ακολουθία συνένεσης, θεωρείτε στοιχισμένη, οπότε αν πχ στη πρώτη εποχή στοιχίσω 2 ακολουθίες  *“ACCGTTACCATAC” και “”AGTTTACATC”* και λάβω τη στοίχιση *“ACCGTT\_ACCATAC” και “A\_\_GTTTAC\_AT\_C”* , προφανώς και θα φτιάξουμε την ακολουθία συνένεσης *“ACCGTTTACCATAC”* και οι 2 στοιχισμένες ακολουθίες που θα αφαιρεθούν από τη διαδικασία θα προστεθούν στο αποτέλεσμα της πολλαπλής στοίχισης και θα θεωρηθούν πλέον πλήρως στοιχισμένες. Όποια από τις αρχικές ακολουθίες βγαίνει από τη διαδικασία θεωρείτε στοιχισμένη. Το unfold είναι στη περίπτωση μας η εκτύπωση της στοίχησης
* **Clustering:** Κατά τη δημιουργία του αλγόριθμου σε python βρήκαμε ένα θεμελιώδες πρόβλημα της παραπάνω στρατηγικής το οποίο οφείλεται στο γεγονός ότι σε κάποια επανάληψη μπορεί να επιλεχτούν προς στοίχηση 2 ακολουθίες συνένεσης. Αν σε αυτές εισαχθεί κάποιο κενό, τότε προστίθεται πληροφορία σε ακολουθία συνένεσης που δεν προήλθε από το dataset και αυτό οδηγεί σε ένα αποσυγχρονισμό των στηλών των ακολουθιών που συνέβαλαν στη δημιουργία της αντίστοιχης ακολουθίας συνένεσης. Τελικά κάποιες ακολουθίες είχαν μικρότερο μήκος από κάποιες άλλες. Αυτό λύνετε με 2 τρόπους: Ο ένας θα ήταν να χρησιμοποιούμε πάντα έστω μια ακολουθία από τις αρχικές, και ο δεύτερος (και αυτός που επιλέχθηκε) θα ήταν να διατηρούμε πληροφορία, ομαδοποιόντας μαζί (clustering) όλες τις ακολουθίες που συνέβαλαν στη δημιουργία μιας ακολουθίας συνένεσης. Αν σε αυτή εισαχθεί κάποιο κενό το εισάγουμε μέσο μιας διαδικασίας propagation σε όλες τις ακολουθίες του αντίστοιχου cluster

## Σύγκριση της μεθόδου με τη μέθοδο clustal (bonus)

Στο τέλος του θέματος κρίναμε σκόπιμο να συγκρίνουμε τη τελική μέθοδο μας με κάποια γνωστή μέθοδο Multi Sequence Alignment η οποία χρησιμοποιείτε ευρέως στο πεδίο της Βιοπληροφορικής. Συγκεκριμένα επιλέξαμε τη μέθοδο clustalW της GenomeNet, με την οποία κάναμε align το dataset A. Τα αποτελέσματα τα κατεβάσαμε σε ένα αρχείο *clustalw.aln* και με βάση αυτό συγκρίναμε τα αποτελέσματα της δικιάς μας custom στοίχισης. Για τη διαδικασία του benchmarking κάναμε:

* **Άθροισμα Ζευγαριών (SP) (32.5%)**

Για κάθε Στοίχησηκακολουθιών k και L στηλών η μετρική ορίζεται ως:

Όπου

Το παραπάνω κανονικοποείτε ως:

* **Σκορ Στύλης (CS) (16.7%)**

Ο αριθμός των στηλών στον οποίο όλες οι ακολουθίες έχουν τον ίδιο χαρακτήρα

Τα παραπάνω υλοποιούνται για τις 2 αντιστοιχήσεις, ακολουθία προς ακολουθία και στήλη προς στήλη για τις ακολουθίες με ίδιο id

* **Ποσοστό ταυτότητας στήλης στις τελικές ακολουθίες συνένεσης (86.11%)**

Με αυτό μετράμε κατά πόσον η τελική ακολουθία συνένεσης των 2 στοιχήσεων ήταν ίδια

Όπως παρατηρούμε η μέθοδος μας, αν και βασισμένη στη λογική, παράγει αρκετά ληψά αποτελέσματα σε σχέση με μια industry standard μέθοδο πολλαπλής στοίχησης, αποτέλεσμα που το περιμέναμε. Για το θέμα 3 θα χρησιμοποιήσουμε τα αποτελέσματα της δικής μας μεθόδου στοίχισης. Οι μετρικές αυτές παρόλα αυτά μπορούν να μας φανούν χρήσιμες στο τέταρτο ερώτημα

## Εκτέλεση

|  |
| --- |
| ACGATTAGGACCCTTTACGACTAAAATTTCACTCC |
| \_\_ACAAGTAAACGATTAGG\_CTCAAATTTCAG\_\_C |
| \_CGATTAGAACGATCTAGGACTCAAATTTCAGTAT |
| ACTATAGCACCCATTTAGGATTCAAATTT\_GGTCG |
| GCCATAGAACGCA\_TTAGGACTCAAATTTCAGTGG |
| AAATTAG\_\_CGC\_ATTAGGACTCCAATTTCAGTTC |
| T\_TAATTAGAAGCTTTAGGACTCAAATTT\_GAGTG |
| \_\_AAAAGAAGG\_ATTTAGGACTCAAGTTTCAGTAG |
| \_\_GATTAGAACGCTTTAGGACT\_AAATTCCAGTAG |
| GAATTAGAACGCATTTAGGACTCAAATTTCAGT\_C |

Table 3: Αποτέλεσμα πολλαπλής στοίχησης για το dataset A

# Θέμα 3

Στο θέμα 3 μας ζητείται η αρχικοποίηση και η εκπαίδευση ενός Hidden Markov Model (HMM) με τα στοιχισμένα προφίλ του θέματος 2 και να το εκπαιδεύσουμε με τις συμβολοσειρές του dataset B. Η εργασία ζητάει μια δική μας υλοποίηση ενός HMM profile οπότε και αυτό θα πράξουμε.

## Ισχυρές Καταστάσεις

Πρώτη μας δουλειά είναι, με αυτόματο τρόπο να κατηγοριοποιήσουμε τις στήλες σε δυνατές και αδύναμες. Αυτό μπορεί να γίνει με έτοιμα μέσα πάνω στα νουκλεοτίδια, παρόλα αυτά οι ακολουθίες με τις οποίες ξεκινήσαμε είναι εντελώς τυχαίες. Μια ιδέα θα ήταν να υπολογίσω το ποσοστό ταυτότητας στοίχησης, και οι καταστάσεις των οποίων το σκορ θα ξεπερνάει ένα κατόφλι θα θεωρούνται ισχυρές. Μπορούμε αυθαίρετα να πούμε ότι το 75% των τιμών είναι ισχυρές. Αυτό είναι μια αθώα (naïve) ιδέα. Στη βιολοπληροφορική έχουν παρατηρηθεί προσπάθειες εύρεσης τέτοιων καταστάσεων με χρήση της θεωρίας πληροφορίας του Shannon. Η μέθοδος αυτή θα υλοποιούταν με τον εξής τρόπο

* Για κάθε στήλη c:

Όπου pi είναι η συχνότητα του αντίστοιχου συμβόλου στη στήλη (θεωρόυμε αυθαίρετα, A = 1, C = 2, G = 3, T = 4, gap = 5)

* Η μέγιστη εντροπία για αλφάβητο 4 χαρακτήρων θα ήταν log2(5), οπότε για να βρουμε τη λεγόμενη περιεκτικότητα σε πληροφορία (IC) θα πρέπει να υπολογίσουμε

Αν IC(c) > 1 τότε θα λέμε ότι η κατάσταση είναι ισχυρή, αλλίως θα λέμε ότι είναι ανίσχυρη

Βρίσκουμε ότι με αυτά τα κριτήρια 26 από τις 35 στήλες θεωρούνται ισχυρές, ενώ η μέση τιμή της μετρικής IC στο σύνολο των στοιχισμένων ακολουθιών είναι 1.44. Η απόδοση μας καλύπτει οπότε θα θεωρήσουμε το παραπάνω ως τον αλγόριθμο εύρεσης ισχυρών καταστάσεων.

## Ζητήματα Markovιανής Μοντελοποίησης

Το HMM το οποίο θα αρχικοποιήσουμε λοιπόν θα έχει **26 + 2** (start , end) καταστάσεις αντιστοίχησης (**match states**), οι οποίες πιθανόν να συνοδεύονται από τις αντίστοιχες καταστάσεις διαγραφής (**delete states**) και **35 – 26 = 9** καταστάσεις εισαγωγής (**insert states**). Η βασικοί κανόνες είναι πως:

* Κάθε **match state** εκπέμπει τα σύμβολα της αντίστοιχης ισχυρής στήλης (όχι το gap) με την αντίστοιχη συχνοτική πιθανότητα (στη πράξη αρχικοποιούμε όλα τα σύμβολα, αυτά τα οποία δεν εμφανίζονται στη στήλη με κάποια οσοδήποτε μικρή πιθανότητα)
* Ακόμη μοντελοποιούμε τις ανίσχυρες στήλες με **insert state**, διατηρώντας πάλι τον ανάλογο πίνακα πιθανοτήτων. Αξίζει να σημειωθεί πως μπορούμε να μοντελοποιήσουμε πολλαπλά συνεχόμενα insert state με αυτοαναφορά.
* Από κάθε **match state** μπορούμε να μεταβούμε στην επόμενη **match state** με κάποια πιθανότητα, ίση με τον αριθμό των συμβόλων διάφορων του κενού στην επόμενη ισχυρή κατάσταση. Ακόμη μπορούμε να μεταβούμε στην μεθεπόμενη κατάσταση (πρακτικά σε οποιαδήποτε κατάσταση πέραν της επόμενης) με κάποιο **delete state**, η οποία πιθανότητα μετάβασης ρυθμίζετε με βάση τον αριθμό των κενών στη στήλη που αντιπροσωπεύει το επόμενο **match state**. Επίσης για κάθε **match state** θα μοντελοποιήσουμε και ένα αντίστοιχο **delete state** με οσοδήποτε μικρή πιθανότητα

αν δε υπάρχει **insert state**

Για τους σκοπούς της μοντελοποίησης θα δημιουργηθούν με τη χρήση numpy:

* Α = πίνακας πιθανοτήτων για τη μετάβαση καταστάσεων (εδώ θα χρειαστούμε header, ή κάποιο mapping ως μεταδεδομένα για να αναγνωρίσουμε τη κάθε κατάσταση)
* B = πίνακας πιθανοτήτων συμβόλων για όλες τις καταστάσεις (πάλι θα χρειαστεί ένα header ή κάποιο άλλο είδος μεταπληροφορίας όπως και στον Α)
* Π = ένα vector με τις αρχικές πιθανότητες των συμβόλων (το οποίο σύμφωνα με τη θεωρία δεν αλλάζει κατά τη διάρκεια της εκπαίδευσης) (χρησιμοποιέιτε κυρίως για την αρχικοποίηση των μηδενικών συμβόλων στο πίνακα Β, δηλαδή αρχικοποιούμε το Β με Dirichlet Prior)

## Fine Tuning (εκπαίδευση)

Ο αλγόριθμος εκπαίδευσης ενός μοντέλου ΗΜΜ μπορεί να να περιγραφεί ως:

* **Για κάθε εποχή:**
  + Για κάθε ακολουθία στο training set (στη περίπτωση μας το dataset B) κάνουμε Viterbi alignment μεταξύ της ακολουθίας και του μοντέλου HMM. Επιστρέφονται μια ακολουθία καταστάσεων (best\_path) και ένα σκορ
  + Αν το λογαριθμικό σκορ όλης της εποχής αποτελεί βελτιστοποίηση, τότε
    - Φτιάχνουμε τον A\*, για όλα τα sequences στο training set και θέτουμε A = A\*
    - Αντίστοιχα για Β
    - **Note:**Στη πραγματικότητα, θα πρέπει να υπολογίσουμε τα Α’ και Β’ και να υπολογίσουμε το σκορ της διαδικασίας Viterbi πριν και μετά, αν το σκορ με τα Α’ και Β’ είναι καλύτερο τότε θα κρατήσουμε τις ανανεωμένες παραμέτρους. Τα Α’ και Β’ ανανεώνονται κατά τη διαδικασία Viterbi, κάθε φορά που επιλέγεται μια μετάβαση ή εκπέμπεται ένα σύμβολο κατά την αναδρομή
* **Viterbi Alignment:**
  + **Αρχικοποίηση:**  (καταστάσεις του μοντέλου Markov)

**πj** = η αρχική πιθανότητα της κατάστασης j

**ej(x1)** = πιθανότητα εκπομπής x1

* + **Αναδρομή:** Για t από 2 έως len(input sequence)
    - Κρατάμε το δείκτη που μεγιστοποίει τη παραπάνω σχέση
  + **Backtracking:** Για t από len(input sequence) έως 2
    - Το σύνολο είναι το σύνολο καταστάσεων
* **Scoring:**
  + Για κάθε ακολουθία που στοιχίζω με το μοντέλο υπολογίζω τη μέγιστη από κοινού πιθανότητα παρατηρήσεων-καταστάσεων, δηλαδή πόσο πιθανή είναι η βέλτιστη διαδρομή
  + Αν το άθροισμα των σκορ της εποχής είναι καλύτερο από το προηγούμενο τότε και μόνο τότε θα ενεργοποιηθεί η διαδικασία fine tuning για όλες τις ακολουθίες της εποχής

# Θέμα 4

# Σχόλια υλοποίησης

# Βιβλιογραφία

* [Multiple Sequence Alignment - CLUSTALW](https://www.genome.jp/tools-bin/clustalw)
* [Bio.AlignIO package — Biopython 1.85 documentation](https://biopython.org/docs/latest/api/Bio.AlignIO.html)
* [Bio.Align package — Biopython 1.85 documentation](https://biopython.org/docs/latest/api/Bio.Align.html#Bio.Align.MultipleSeqAlignment)
* [Bio.motifs package — Biopython 1.85 documentation](https://biopython.org/docs/latest/api/Bio.motifs.html)
* [Microsoft PowerPoint - 05\_msa\_clustalw.pptx](https://www.informatik.hu-berlin.de/de/forschung/gebiete/wbi/teaching/archive/ss11/vl_bioinfo/05_msa_clustalw.pdf?)
* [Microsoft PowerPoint - week4-monday.ppt](https://www.cs.cornell.edu/courses/cs426/2003fa/week4-monday.pdf)
* [BAliBASE (Benchmark Alignment dataBASE): enhancements for repeats, transmembrane sequences and circular permutations - PMC](https://pmc.ncbi.nlm.nih.gov/articles/PMC29792/)
* [EMBOSS: needle manual](https://www.bioinformatics.nl/cgi-bin/emboss/help/needle?utm_source=chatgpt.com)
* [2308.12103](https://arxiv.org/pdf/2308.12103)
* [H2r: Identification of evolutionary important residues by means of an entropy based analysis of multiple sequence alignments | BMC Bioinformatics | Full Text](https://bmcbioinformatics.biomedcentral.com/articles/10.1186/1471-2105-9-151)
* [Incorporating background frequency improves entropy-based residue conservation measures - PMC](https://pmc.ncbi.nlm.nih.gov/articles/PMC1562451/)
* [machine learning - Baum Welch training of HMM - Cross Validated](https://stats.stackexchange.com/questions/275535/baum-welch-training-of-hmm?)
* [Viterbi algorithm - Wikipedia](https://en.wikipedia.org/wiki/Viterbi_algorithm)