ΠΑΝΕΠΙΣΤΗΜΙΟ ΠΕΙΡΑΙΩΣ

Τμήμα Πληροφορικής



Εργασία Μαθήματος [***ΒΙΟΠΛΗΡΟΦΟΡΙΚΗ***](http://gunet2.cs.unipi.gr/courses/TMC126/)

|  |  |
| --- | --- |
| Άσκηση *<<αριθμός άσκησης>>* | ***<<>>*** |
| Όνομα φοιτητή – Αρ. Μητρώου  (όλων σε περίπτωση ομαδικής εργασίας) | Κοτταρίδης Αθανάσιος Π-15059 |
|  |
|  |
|  |
| Ημερομηνία παράδοσης |  |

**Εκφώνηση της άσκησης**

- Η εργασία είναι απαλλάκτική και εκπονείται σε ομάδες των 1-3 φοιτητών. Όσοι φοιτητές δεν λάβουν προβιβάσιμο βαθμό, εξετάζονται και βαθμολογούνται γραπτώς των Σεπτέμβριο, ομοίως και όσοι δεν υποβάλουν εργασία.

- Η εργασία παραδίδεται με email στο pikrakis@unipi.gr, έως και την ημέρα εξέτασης του μαθήματος στην κανονική εξεταστική περίοδο (όχι την εμβόλιμη, αν υπάρξει), ώρα 23:59. Μπορείτε να στείλετε dropbox link, ΜΟΝΟ εάν τα επισυναπτόμενα υπερβαίνουν σε όγκο τη δυνατότητα αποστολής με email.

- Υλοποιήστε 5 (πέντε) εκ των παρακάτω ασκήσεων του βιβλίου:

6.12, 6.13, 6.14, 6.15, 6.16 (υποχρεωτικά μία ή δύο το πολύ)

6.22, 6.23, 6.27, 6.37, 6.51,

11.4, 11.6 (υποχρεωτικά μία εκ των δύο)

- Οι ασκήσεις είναι ισοδύναμες. Αποδεκτές γλώσσες υλοποίησης είναι μόνο οι Python, Octave/Matlab, C/C++. Κάθε άσκηση συνοδεύεται από τεκμηρίωση της λύσης. Ορισμένες εκ των ασκήσεων συσχετίζονται με επιπρόσθετα δεδομένα, για τις περιγραφές των οποίων θα λάβετε νέα ανακοίνωση.

- Ακολουθήστε τις οδηγίες που έχω δώσει για επιτυχή εκπόνηση. Θα γίνει αυστηρός έλεγχος για να διαπιστωθούν πιθανές περιπτώσεις αντιγραφής.

ΠΙΝΑΚΑΣ ΠΕΡΙΕΧΟΜΕΝΩΝ

**1.** Εισαγωγή………………………………………………………………………………………..…………………………3

**2.** Άσκηση 6.14(Κεφάλαιο 6, προγραμματιστική άσκηση 14, σελ. 247)…………………………………4

**2.1.** ........................................................................................................................

**2.2.** ................................................................................................................

**2.2.1.** ...................…………………………………………………………........................................

**2.2.2.** ...................………………………………………………………….............................

**2.3.** …………………………………………………………………………………………

**3.** Άσκηση 6.15(Κεφάλαιο 6, προγραμματιστική άσκηση 15, σελ. 247)…………………………………4

**3.1.**.....................................................................................................

**3.2.** .............................................................................................

**3.2.1.** …………………………………………………………........................................

**3.2.2.** ………………………………………………………….............................

**3.3.** …………………………………………………………………………………………

**4.** Άσκηση 6.22(Κεφάλαιο 6, προγραμματιστική άσκηση 22, σελ. 249)…………………………………4

**4.1.** Γενική Επεξήγηση.....................................................................................................

**4.2.** Λειτουργία Αλγορίθμου.............................................................................................

**4.3.** Εκτέλεση Προγράμματος …………………………………………………………………………………………

**5.** Άσκηση 6.37(Κεφάλαιο 6, προγραμματιστική άσκηση 37, σελ. 255)…………………………………4

**5.1.** .....................................................................................................

**5.2.** .............................................................................................

**5.3.** …………………………………………………………………………………………

**6.** Άσκηση 11.6(Κεφάλαιο 11, προγραμματιστική άσκηση 6, σελ. 451)…………………………………4

**6.1.** Γενική Επεξήγηση.....................................................................................................

**6.2.** Ανάλυση μοντέλου (ερώτημα 1ο )..........................................................................

**6.2.1** HMM μοντέλο του προβλήματος……………………………………………………………………………..

**6.2.2** Προγραμματιστική αναπαράσταση του μοντέλου…………………………………….

**6.3** Υπολογισμός ακολουθίας παρατηρήσεων (ερώτημα 2ο)…………………………………..

**6.3.1** Ψευδοκοδικας Viterbi:………………………………………………………………..

**6.4** Υπολογισμός δεύτερης βέλτιστης ακολουθίας καταστάσεων (3ο ερώτημα)………………

**6.4.1** Αναλυτική περιγραφή του αλγορίθμου k-best:……………………………………….

**6.5.** Εκτέλεση Προγράμματος …………………………………………………………………………………………

***4 Άσκηση 6.22***

**4.1.** Γενική Επεξήγηση

Για την επίλυση αυτής της προγραμματιστικής άσκησης εφαρμόζετε ο αλγόριθμος Προσαρμογής Αλληλουχιών ο οποίος είναι μια παραλλαγή του αλγορίθμου καθολικής στοίχισης Needleman-Wunsch. Ο αλγόριθμος προσαρμογής αλληλουχιών συμπληρώνει των πίνακα δυναμικού προγραμματισμού σύμφωνα με την παρακάτω σχέση:

S[i,j] = max{S[i-1,j-1)]+s[xi, yj], S[i-1,j]-1, S[i,j-1]-1 }

Με S[0,j] = 0 και S[i,0] = -i

Στη πραγματικότητα η μόνη διαφορά του Αλγορίθμου Προσαρμογής με των αλγόριθμο καθολικής στοίχισης Needleman-Wunsch είναι ότι ο αλγόριθμος Προσαρμογής αρχικοποιει την πρώτη γραμμή του πίνακα S[m,n] με 0.

Για την υλοποίηση της άσκησης χρησιμοποιήθηκε η βιβλιοθήκη:

* Numpy

**4.2.** Λειτουργία Αλγορίθμου

Ο αλγόριθμος διαβάζει δυο αλληλουχίες (V,W) από δυο txt αρχεία τα “data1\_6.22” και “data2\_6.22” αντίστοιχα. Αυτοί οι δυο φάκελοι βρίσκονται μέσα στων φάκελο εκτέλεσης του προγράμματος. Επίσης η αλληλουχίαV πρέπει να έχει μεγαλύτερο μήκος από την αλληλουχία W. Στη συνέχεια, Χρησιμοποιώντας την μέθοδο Fit\_Sequence() ο αλγόριθμος δημιουργεί 2 NumPy array S[n,m] και B[n,m] οπού το n αναπαριστά το μήκος της αλληλουχίας W αυξημένο κατά 1 και το m αναπαριστά το μήκος της αλληλουχίας V αυξημένο κατά ένα αντίστοιχα .Ο S[n,m] ο οποίος περιέχει της βαθμολογίες ομοιότητας που προκύπτουν για της αλληλουχίες. Ο πίνακας B[n,m] ο οποίος περιέχει τους δείκτες οπισθοδρόμησης προκείμενου να μπορέσουμε να αναπαραστήσουμε της ακολουθίες που οδηγούν στην υποσυμβολοσυρα της V που ταιριάζει καλλίτερα με ολόκληρη την W. Για την αναπαράσταση των δεικτών οπισθοδρόμησης χρησιμοποιούνται οι αριθμοί (1,2,3) οπού:

* + Ο αριθμός 1 αναπαριστά ότι πρόκριτε για αφαίρεση του συμβόλου Wi
  + Ο αριθμός 2 αναπαριστά ότι πρόκριτε για αφαίρεση του συμβόλου Vj
  + Ο αριθμός 3 αναπαριστά ότι στης ακολουθίες καταγράφετε το Wi και το Vj είτε αυτά τρίζουν είτε όχι.

Στη συνέχεια ο αλγόριθμος χρησιμοποιεί την σχέση που περιγραφικέ προηγούμενος προκείμενου να συμπληρώσει των πίνακας S και ανάλογα από το αποτέλεσμα της σχέσης παίρνει τιμή και ο πίνακας B . Ακολουθεί ο ψευδοκοδικας για την μέθοδο Fit\_Sequence().

**Ψευδοκοδικας Fit\_Sequence():**

Fit\_Sequence(V,W){

d ← MismatchScore

n ← length(W)+1

m ← length(V)+1

**for** i=0 **to n**

S(i,0) ← d\*i

**for** j=0 **to** m

S(0,j) ← 0

**for** i=1 **to** **n**

**for** j=1 **to** **m**

{

Match ← S(i-1,j-1) + Similar(Ai, Bj)

Delete ← S(i-1, j) + d

Insert ← S(i, j-1) + d

S(i,j) ← **max**(Match, Insert, Delete)

If V[j - 1] = W[i - 1] and Match = **max**(Match, Insert, Delete){

B(i,j) ← **3**

**}**

Else{

If Delete = **max**(Match, Insert, Delete)

B(i,j) ← **1**

Else if Insert = **max**(Match, Insert, Delete)

B(i,j) ← **2**

Else

B(i,j) ← **3**

}

}

Return S,B

}

Για την εμφάνιση των αποτελεσμάτων του αλγόριθμου στων χρήστη χρησιμοποιείτε ο αλγόριθμος Read\_B() ο οποίος διαβάζει των πίνακας Β και ανάλογα με το αποτέλεσμα του συμπληρώνει 2 συμβολοσειρές την AlignmetV και την AlignmetW. Ο υπολογισμός των βέλτιστων ακολουθιών γίνεται ως εξής : Αρχικά βρίσκουμε από την τελευταία γραμμή του S το μέγιστο score και στη συνέχεια πάμε στην αντίστοιχη θέση του πίνακα Β και υπολογίζουμε της αλληλουχίες με των τρόπο που υ περιγράφετε στων παρακάτω ψευδοκοδικα.

**Ψευδοκοδικας Read\_B():**

AlignmentV ← ""

AlignmentW ← ""

i ← **length**(W)

j ← **argmax(S(i))**

BestScore ← S(i,j)

**while** (B(i,j) != 0)

{

**if** (B(i,j) = 3)

{

AlignmentW ← Wi + AlignmentW

AlignmentV ← Vj + AlignmentV

i ← i - 1

j ← j - 1

}

**else** **if** (B(i,j) = 1)

{

AlignmentW ← Wi + AlignmentW

AlignmentV ← "-" + AlignmentV

i ← i - 1

}

**else**

{

AlignmentW ← "-" + AlignmentV

AlignmentV ← Vj + AlignmentV

j ← j - 1

}

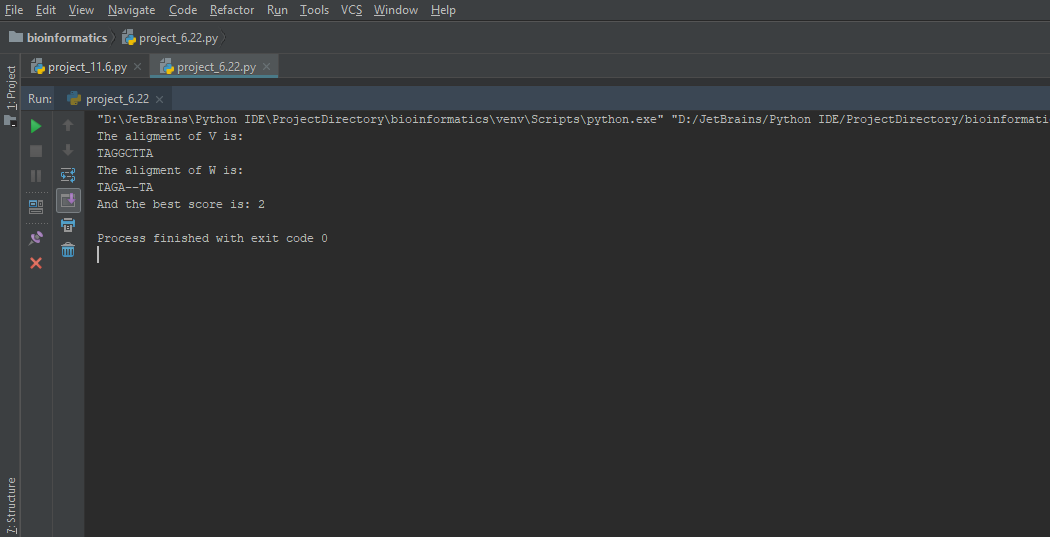
}

**4.3.** Εκτέλεση Προγράμματος.

Παράδειγμα εκτέλεσης για της ακολουθίες του παραδείγματος της άσκησης 6.22 του βιβλίου.

* Ακολουθία V = GTAGGCTTAAGGTTA
* Ακολουθία W = 'TAGATA'

Screenshot Εκτέλεσης:



Στο αποτέλεσμα του προγράμματος προβάλετε η υποσυμβολοσειρά της V καθώς και το πως πρέπει να τροποποιηθεί η w προκείμενου να επιτευχθεί το «καλλίτερο ταίριασμα» επίσης προβάλει και το σκορ που επιτυγχάνετε για την καλλίτερη στοίχιση.

***6 Άσκηση 11.6***

**6.1.** Γενική Επεξήγηση

Στην άσκηση αυτή μας περιγράφετε ένα παιχνίδι τύχης με 2 τρίπλευρα «πειραγμένα» ζαρια των οποίον μας δίνονται όλα τα χαρακτηριστικά και μας ζητάτε να δημιουργήσουμε ένα HMM που να μοντελοποιεί της πιθανές κατάστασης και στη συνέχεια να υπολογίσουμε την ακολουθία καταστάσεων που παράγει το μεγαλύτερο σκορ για μια συγκεκριμένη ακολουθία.

**6.2.** Ανάλυση μοντέλου (ερώτημα 1ο )

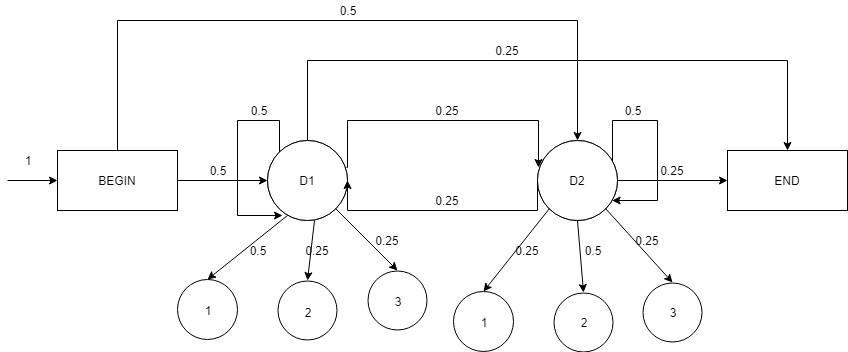
Στο πρώτο ερώτημα της άσκησης μας ζητάτε να αναλύσουμε το μοντέλο που περιγράφετε και να δημιουργήσουμε το HMM του.

* Κατάστασης μοντέλου:
  1. Αρχική κατάσταση BEGIN
  2. Χρήση του ζαριού D1
  3. Χρήση του ζαριού D2
  4. Τερματική κατάσταση END
* Αλφάβητο μοντέλου:

Το αλφάβητο του μοντέλου είναι τα πιθανά αποτελέσματα του ζαριού δηλαδή {1,2,3}

* Πιθανές μεταβάσεις:
  1. Από BEGIN σε Α με πιθανότητα 0.5
  2. Από BEGIN σε B με πιθανότητα 0.5
  3. Από Α σε Α με πιθανότητα 0.5
  4. Από Α σε Β με πιθανότητα 0.25
  5. Από Α σε END με πιθανότητα 0.25
  6. Από B σε Α με πιθανότητα 0.25
  7. Από B σε B με πιθανότητα 0.5
  8. Από B σε END με πιθανότητα 0.25
* Πιθανές παρατηρήσεις:
  1. Με το ζάρι D1 να εκπέμπει 1 με πιθανότητα 0.5
  2. Με το ζάρι D1 να εκπέμπει 2 με πιθανότητα 0.25
  3. Με το ζάρι D1 να εκπέμπει 3 με πιθανότητα 0.25
  4. Με το ζάρι D2 να εκπέμπει 1 με πιθανότητα 0.25
  5. Με το ζάρι D2 να εκπέμπει 2 με πιθανότητα 0.5
  6. Με το ζάρι D2 να εκπέμπει 3 με πιθανότητα 0.25

**6.2.1** HMM μοντέλο του προβλήματος:

******

**6.2.2** Προγραμματιστική αναπαράσταση του μοντέλου:

Η παρακάτω αναπαράσταση είναι σε γλώσσα Python.

obs = ('1', '1', '2', '1', '2', '2')  
  
states = ('D1', 'D2','END')  
  
alpha\_beta = ('1', '2', '3')  
  
start\_prob = {'D1': 0.5, 'D2': 0.5,'END': 0}  
  
trans\_prob = {  
 'D1': {'D1': 0.5, 'D2': 0.25,'END': 0.25},  
 'D2': {'D1': 0.25, 'D2': 0.5,'END': 0.25},  
 'END':{'D1': 0, 'D2': 0,'END': 0}  
 }  
  
emit\_prob = {  
 'D1': {'1': 0.5, '2': 0.25, '3': 0.25},  
 'D2': {'1': 0.25, '2': 0.5, '3': 0.25},  
 'END': {'1': 0, '2': 0, '3': 0},  
 }

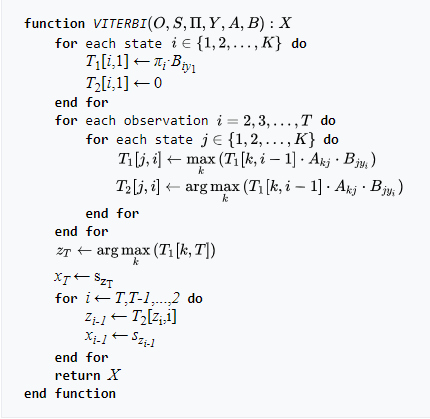
**6.3** Υπολογισμός ακολουθίας παρατηρήσεων (ερώτημα 2ο)

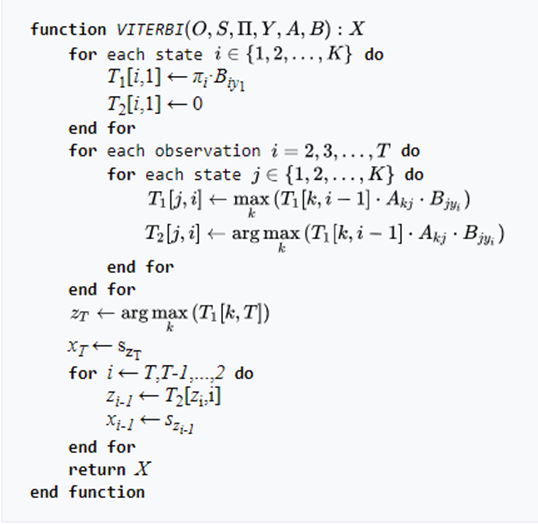
Σε αυτό το ερώτημα της άσκησης μας ζητείτε να υπολογίσουμε την ακολουθία καταστάσεων που παράγει το μεγαλύτερο σκορ για την ακολουθία 112122 χρησιμοποιώντας το ΗΜΜ του προηγούμενου ερωτήματος. Για να γίνει αυτό έχει υλοποιηθεί προγραμματιστικά ο αλγόριθμος Viterbi ο οποίος υλοποιείτε μέσα από τα εξής βήματα.

* Αρχικά ο αλγόριθμος υπολογίζει από ποια κατάσταση θα εκπέμπει το αρχικό σύμβολο ελέγχοντας της πιθανότητες μετάβασης από την αρχική κατάσταση «BEGIN» και της πιθανότητες εκπομπής συμβόλου σε κάθε κατάσταση.
* Στη συνέχεια υπολογίζει και για τα υπόλοιπα σύμβολα από ποια κατάσταση έχουν μεγαλύτερη πιθανότητα να εκπέμπουν υπολογίζοντας το άθροισμα τον πιθανοτήτων μετάβασης και εκπομπής και λαμβάνοντας υπόψιν την τρέχουσα κατάσταση που βρίσκετε και αποθηκεύοντας τα σε κατάλληλες δομές.
* Τέλος διαβάζει «ανάποδα» την ακολουθία καταστάσεων (πηγαίνοντας από την END προς την BEGIN) υπολογίζοντας κάθε φορά σε ποια κατάσταση παρουσιάστηκε η εκπομπή του εκάστοτε συμβόλου με μεγαλύτερη πιθανότητα. Από αυτή την αναδρομή προκύπτει το τελικό BEST PATH που είναι και το αποτέλεσμα του αλγορίθμου.

**6.3.1** Ψευδοκοδικας Viterbi:

Ο ψευδοκοδικας που παρατίθεται παρακάτω έχει τροποποιηθεί ελαφρός για την προγραμματιστική εκτέλεση της άσκησης. Πρώτον το πρόγραμμα που έχουμε υλοποιήσει για την επίλυση της άσκησης χρησιμοποιεί λεξικά «Dictionaries» σε αντίθεση με τον ψευδοκοδικα που χρησιμοποιεί array. Δεύτερον για των υπολογισμός των πιθανοτήτων γίνεται με των τύπο του βίβλου για υπολογισμό λογαριθμικών βαθμολογιών σελ. 439. Παρόλα αυτά παρατίθεται ο γενικός ψευδοκοδικας για την καλλίτερη περιγραφή των βημάτων του αλγορίθμου.





{\displaystyle (O,S,\Pi ,Y,A,B):X}

**6.4** Υπολογισμός δεύτερης βέλτιστης ακολουθίας καταστάσεων (3ο ερώτημα)

Σε αυτό το ερώτημα υλοποιείτε ο αλγόριθμος k-best o οποίος υπολογίζει τα κ καλλίτερα μονοπάτια που αποδίδουν τη βέλτιστη διαδρομή στο μοντέλο του 1ου ερωτήματος για την ακολουθία παρατηρήσεων του 2ου . Σκοπός μας είναι να υπολογίσουμε προγραμματιστικά της 2 βέλτιστες ακολουθίες καταστάσεων που προκύπτουν για την ακολουθία παρατηρήσεων 112122.

**6.4.1** Αναλυτική περιγραφή του αλγορίθμου k-best:

Σκοπός του αλγορίθμου k-best είναι να υπολογίσει όλα τα μονοπάτια που έχουν το βέλτιστο σκορ. Πρακτικά για να υπάρχουν περισσότερα από ένα BEST PATHS πρέπει να ικανοποιείτε τουλάχιστον ένα από τα παρακάτω ενδεχόμενα:

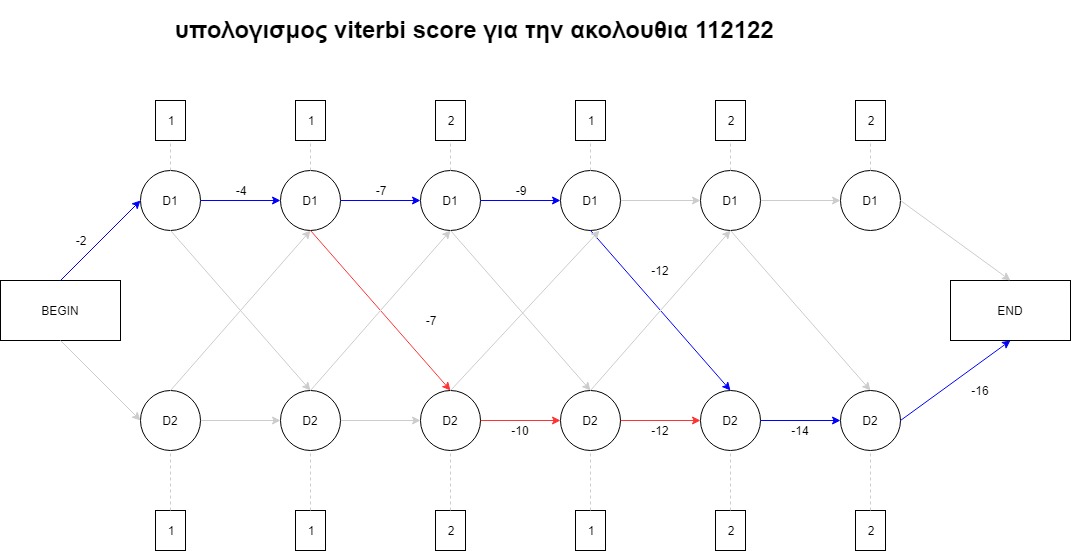
1. Να καταλήγουν στην τελική κατάσταση END, 2 διαφορετικά μονοπάτια με βέλτιστο σκορ
2. Να μπορείς να μετάβεις σε κάποια κατάσταση κάποιου BEST PATH από 2 ή περισσότερους κόμβους με βέλτιστο σκορ ( πράγμα που συμβαίνει και στην άσκηση μας οπού για την παρατήρηση Ο[4] μπορούμε να μετάβουμε στον D2 από 2 paths (το μπλε και το κόκκινο) με σκορ -12 που είναι και το βέλτιστο για αυτή τη κατάσταση όπως φαίνεται και στο σχήμα που παρουσιάζετε παρακάτω).

Κάθε φορά που ικανοποιείτε μια από της 2 συνθήκες σημαίνει ότι υπάρχει και άλλο βέλτιστο μονοπάτι. Σε αυτή τη περίπτωση καταγράφουμε της προηγούμενες κατάστασης είτε είναι μόνο η END είτε είναι περισσότερες και ξεκινάμε από εκείνο το σημείο και καταγράφουμε το κάθε μονόπατη ξεχωριστά. Ακόμα και αν κάποια μονοπάτια στη ποριά ενωθούν (όπως συμβαίνει στην άσκηση μας) συνεχίζουμε να τα καταγράφουμε ξεχωριστά.

Για την εύρεση όλον των βέλτιστων μονοπατιών ο αλγόριθμος k-best λειτουργεί ως έξεις.

* Αρχικά υπολογίζει για κάθε παρατήρηση την πιθανότητα εκπομπής της σε όλες της κατάστασης ξεκινώντας από την BEGIN για την πρώτη παρατήρηση και για της υπόλοιπες θεωρεί ως πιθανότητα προηγουμένης κατάστασης την βέλτιστη πιθανότητα να φτάσει στην προηγουμένη κατάσταση. Για κάθε παρατήρηση αποθηκεύει την πιθανότητα εκπομπής της σε μια κατάσταση και από ποια κατάσταση προέρχεται.
* Αφού υπολογίσει όλες της παρατήρησης ξεκινάει μια αναδρομή ελέγχοντας της συνθήκες που αναφερθήκαν προηγούμενος από την τελική κατάσταση END και για όλες της κατάστασης που έχουν υπολογιστεί. Υπολογίζοντας κάθε φορά σε ποια κατάσταση είναι και από ποιες καταστάσεις θα μπορούσε να βρεθεί εκεί και επιλέγοντας αυτή με την βέλτιστη πιθανότητα.

Στο παρακάτω γράφημα αναπαριστάτε με μπλε χρώμα η βέλτιστη διαδρομή καταστάσεων που προκύπτει από την εκτέλεση του αλγορίθμου Viterbi για την ακολουθία 112122 για την επίλυση του 2ου ερωτήματος και με κόκκινο χρώμα η δεύτερη βέλτιστη ακολουθία που προκύπτει από των από την εκτέλεση του αλγορίθμου των K-best paths για την επίλυση του 3ου ερωτήματος της άσκησης. Οι πιθανότητες των μεταβάσεων είναι λογαριθμισμενες και αναγράφοντες μόνο οι πιθανότητες των βέλτιστων διαδρομών

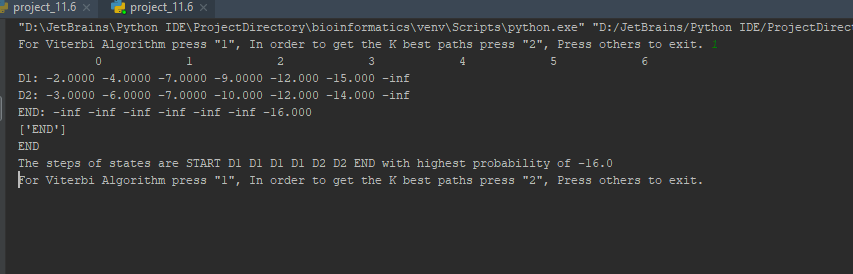


**6.5** Παραδείγματα εκτέλεσης του προγράμματος.

Κατά την εκτέλεση του προγράμματος σας δίνετε η επιλογή να επιλέξετε αν θέλετε να προβάλετε το αποτέλεσμα του Viterbi ή του αλγορίθμου k-best πατώντας 1 ή 2 αντίστοιχα.

* Υπολογισμός BEST PATH με τη χρήση του αλγόριθμου Viterbi.

Στο screenshot της εκτέλεσης του προγράμματος φαίνεται το διάγραμμά trellis προβάλλοντας της πιθανότητες εκπομπή κάθε παρατήρησης σε όλες της κατάστασης και αποκάτω φαίνεται η βέλτιστη ακολουθία καταστάσεων καθώς και η λογαριθμισμενη πιθανότητα εκπομπής της.



* Υπολογισμός τον Κ-βέλτιστων διαδρομών με την χρήση του αλγόριθμου k-best.

Στο screenshot της εκτέλεσης του προγράμματος προβάλλονται όλα τα best paths (στη περίπτωση μας 2) .

