

ΠΑΝΕΠΙΣΤΗΜΙΟ ΙΩΑΝΝΙΝΩΝ ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΤΗΛΕΠΙΚΟΙΝΩΝΙΩΝ

ΑΛΓΟΡΙΘΜΟΙ ΚΑΙ ΠΟΛΥΠΛΟΚΟΤΗΤΑ

Το πρόβλημα της Μέγιστης κοινής υποακολουθίας (LCS = Longest Common Subsequence).

Περσεφόνη Πατσέα

A.M.: 16106 -1731

7ο Εξάμηνο

26 Ιανουαρίου 2022

ΠΕΡΙΛΗΨΗ

Στη συγκεκριμένη άσκηση ζητούνταν η επίλυση του προβλήματος της μέγιστης κοινής υποακολουθίας [Bla20], [Sta98]. Η επίλυση του προβλήματος, μεταξύ άλλων, έχει εφαρμογή στη βιοπληροφορική, και ειδικότερα στην ανάλυση ομοιοτήτων σε ακολουθίες DNA. Οι ακολουθίες DNA αναπαρίστανται με συμβολοσειρές που σχηματίζονται από 4 χαρακτήρες (A,G,C,T) που αναπαριστούν τα νουκλεοτίδια, αδερίνη, γουανίνη, κυτοσίνη και θυμίνη.

0.1 Εισαγωγή

Έστω ότι δίνονται δύο συμβολοσειρές X και Y με μήκη m και n αντίστοιχα. Στο πρόβλημα της μέγιστης κοινής υποακολουθίας ζητείται να βρεθεί η μέγιστη σε μήκος υποακολουθία που εντοπίζεται και στις δύο συμβολοσειρές X και Y . Η υποακολουθία αποτελείται από χαρακτήρες που μπορούν να εντοπιστούν και στις δύο συμβολοσειρές με την ίδια σειρά από αριστερά προς τα δεξιά.

Ο αλγόριθμος ωμής δύναμης (Brute Force) δημιουργεί όλες τις υποακολουθίες του X (2^m σε πλήθος υποακολουθίες) και ελέγχει ποια είναι η μεγαλύτερη που υπάρχει και στο Y .

0.2 Αποτελέσματα

Για την συγγραφή των πειραμάτων χρησιμοποιήθηκε η Python 3.10.0 και το Visual Studio Code. Τα χαρακτηριστικά του υπολογιστή είναι i5 7600 (3.50 GHz), 16.0 GB RAM.

Οδηγίες Εκτέλεσης του Κώδικα:

Για να εκτελέσουμε τον κώδικα, κάνουμε τα εξής βήματα:

- i) Ανοίγουμε το cmd (Windows + R)
- ii) Πηγαίνουμε στο φάκελο που είναι αποθηκευμένος ο κώδικας
- iii) Πατάμε την εντολή `python file_name.py`, προκειμένου να γίνει η εκτέλεση του κώδικα

Για το *main.py*:

```
(python -u " c:\Users\user\Desktop\Εργασία 3\Κώδικες\main.py")
```

Για το *UnitTest.py*:

```
(python -u " c:\Users\user\Desktop\Εργασία 3\Κώδικες\UnitTest.py")
```

Αλγόριθμος Longest Common Subsequence:

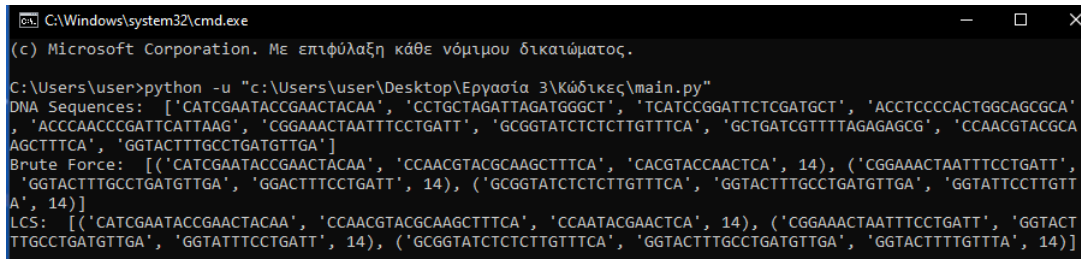
Δήλωση Προβλήματος LCS: Με δεδομένες δύο ακολουθίες, βρίσκεται το μήκος της μεγαλύτερης υποακολουθίας που υπάρχει και στις δύο. Μία υποακολουθία είναι μία ακολουθία που εμφανίζεται με την ίδια σχετική σειρά, αλλά όχι απαραίτητα συνεχόμενη.

Για την εύρεση της πολυπλοκότητας ωμής δύναμης, πρέπει πρώτα να είναι γνωστός ο αριθμός των πιθανών διαφορετικών υποακολουθιών μιας συμβολοσειράς με μήκος n . Δηλαδή θα πρέπει πρώτα να βρεθεί ο αριθμός των υποακολουθιών με μήκη που κυμαίνονται από: $1, 2, \dots, n-1$.

Από τη θεωρία μετάθεσης και του συνδυασμού, ισχύει ότι ο αριθμός των συνδυασμών με 1 στοιχείο είναι nC_1 . Ο αριθμός των συνδυασμών με 2 στοιχεία είναι nC_2 , και ούτω καθεξής. Επομένως, για n στοιχεία ισχύει: ${}^nC_0 + {}^nC_1 + {}^nC_2 + \dots + {}^nC_n = 2^n$. Άρα μία συμβολοσειρά μήκους n έχει $2^n - 1$ διαφορετικές πιθανές υποακολουθίες αφού δε θεωρείται

υποακολουθία με μήκος 0. Αυτό σημαίνει ότι η χρονική πολυπλοκότητα της προσέγγισης της ωμής δύναμης θα είναι $O(n * 2^n)$.

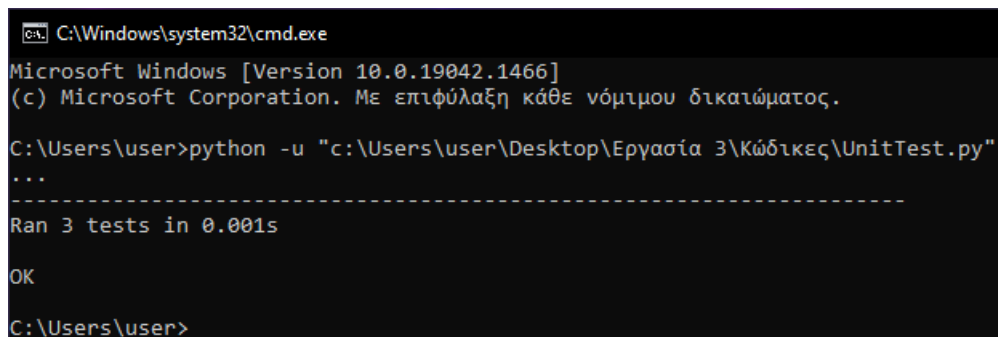
Αποσπάσματα οθόνης από εκτέλεση του κώδικα:



```
C:\Windows\system32\cmd.exe
(c) Microsoft Corporation. Με επιφύλαξη κάθε νόμιμου δικαιώματος.

C:\Users\user>python -u "c:\Users\user\Desktop\Εργασία 3\Κώδικες\main.py"
DNA Sequences: ['CATCGAATACCGAACTACAA', 'CCTGCTAGATTAGATGGGCT', 'TCATCCGGATTCTCGATGCT', 'ACCTCCCCACTGGCAGCGCA',
, 'ACCCAAACCCGATTCAATTAAG', 'CGGAACTAATTCTCTGATT', 'GCGGTATCTCTCTTGTTC', 'GCTGATCGTTTAGAGAGCG', 'CCAACGTACGCA
AGCTTTCA', 'GGTACTTTGCCTGATGTTGA']
Brute Force: [('CATCGAATACCGAACTACAA', 'CCAACGTACGCAAGCTTTCA', 'CACGTACCAACTCA', 14), ('CGGAACTAATTCTCTGATT',
, 'GGTACTTTGCCTGATGTTGA', 'GGACTTTCTGATT', 14), ('GCGGTATCTCTCTTGTTC', 'GGTACTTTGCCTGATGTTGA', 'GGTATTCTTGT
A', 14)]
LCS: [('CATCGAATACCGAACTACAA', 'CCAACGTACGCAAGCTTTCA', 'CCAATACGAACTCA', 14), ('CGGAACTAATTCTCTGATT', 'GGTACT
TTGCCTGATGTTGA', 'GGTATTCTGATT', 14), ('GCGGTATCTCTCTTGTTC', 'GGTACTTTGCCTGATGTTGA', 'GGTACTTTGTTC', 14)]
```

Εικόνα 1: Έξοδος Κώδικα "main.py"



```
C:\Windows\system32\cmd.exe
Microsoft Windows [Version 10.0.19042.1466]
(c) Microsoft Corporation. Με επιφύλαξη κάθε νόμιμου δικαιώματος.

C:\Users\user>python -u "c:\Users\user\Desktop\Εργασία 3\Κώδικες\UnitTest.py"
...
-----
Ran 3 tests in 0.001s

OK

C:\Users\user>
```

Εικόνα 2: Έξοδος Κώδικα "UnitTest.py"

*** Οι κώδικες πραγματοποιήθηκαν για 10 υποθετικές υποακολουθίες, με 20 χαρακτήρες η κάθε μία ***

0.3 Συμπεράσματα

Τα χαρακτηριστικά των προβλημάτων στα οποία εφαρμόζεται ο δυναμικός προγραμματισμός είναι η ιδιότητα των βέλτιστων επιμέρους δομών. Η αναδρομική εξίσωση περιγραφής της βέλτιστης λύσης, συνήθως, βασίζεται στην ιδιότητα των βέλτιστων επιμέρους δομών. Έτσι, η ύπαρξη της ιδιότητας των βέλτιστων επιμέρους δομών για ένα πρόβλημα αποτελεί ικανοποιητική ένδειξη ότι ο δυναμικός προγραμματισμός εφαρμόζεται σε αυτό.

Ένα άλλο χαρακτηριστικό των προβλημάτων στα οποία εφαρμόζεται ο δυναμικός προγραμματισμός, είναι ο σχετικά μικρός αριθμός των διαφορετικών επιμέρους προβλημάτων, τα οποία εμφανίζονται κατά την εφαρμογή της μεθόδου. Με τον όρο «σχετικά μικρός», συνήθως, εννοείται ένας αριθμός επιμέρους προβλημάτων που φράσσεται άνω από κάποιο πολυώνυμο του μεγέθους της εισόδου, δια κάποια σταθερά, δηλαδή είναι $O(n^d)$, $d > 0$.