ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ
ΤΜΗΜΑ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ

ΕΠΕΞΕΡΓΑΣΙΑ ΦΩΝΗΣ ΚΑΙ ΦΥΣΙΚΗΣ ΓΛΩΣΣΑΣ

ΕΡΑΣΤΗΡΙΑΚΗ ΑΝΑΦΟΡΑ 1ης ΑΣΚΗΣΗΣ

Μετατροπέας για Greeklish

ΡΟΛΑΝΔΟΣ-ΑΛΕΞΑΝΔΡΟΣ ΠΟΤΑΜΙΑΣ
Α.Μ. 03114437

ΑΛΕΞΑΝΔΡΟΣ ΝΕΟΦΥΤΟΥ
Α.Μ. 03113155

Προπαρασκευή

Στο πρώτο στάδιο της εργασίας ανατρέξαμε στα στάδια της προπαρασκευής ώστε να βελτιώσουμε τα αρχεία που παράγονται από των κώδικα μας. Συγκεκριμένα βελτιώσαμε τον κώδικα του ***Part1.py*** *στον οποίο εκτελούνται τα* ***βήματα 1 και 2*** *και παράγονται τα λεξικά που θα χρησιμοποιήσουμε για την δημιουργία του αυτόματου G.fst. Ταυτόχρονα δημιουργήσαμε ένα ενιαίο αρχείο με input/output symbols ώστε να το χρησιμοποιήσουμε σε όλα τα fst που θα δημιουργήσουμε. Επίσης τροποποιήσαμε τον κώδικα* ***accept.py*** *ώστε να δημιουργηθούν σωστά οι αποδοχείς A1 και A2, που αποδέχονται τις λέξεις για το ελληνικό και το αγγλικό λεξικό αντίστοιχα.*

Βήμα 8

Αφού εκτελέσουμε κατά σειρά τους κώδικες ***Part1.py*** *και* ***accept.py****, ή με την εντολή:*
`$ make prelab` *στο terminal , μπορούμε να εκτελέσουμε το αρχείο FST_creat.py μέσω του terminal. Με αυτό το αρχείο εκτελούμε τα βήματα 8, 9, 10. Σε πρώτη φάση δημιουργούμε την ένωση των* ***A1.fsa*** *,* ***A2.fsa*** *(****$A_1 \cup A_2$****) με την εντολή του openFST* ***fstunion****. Στην συνέχεια δημιουργούμε με τον ίδιο τρόπο την ένωση του G.fst (βήμα 3) με το I.fst (βήμα 4). Τα δυο επιμέρους fst που έχουμε δημιουργήσει τα συνθέτουμε με την εντολή* ***fstcompose****. Για την καλύτερη λειτουργία της σύνθεσης εκτελούμε στα δύο fst που πρόκειται να συνθέσουμε την εντολή* ***arcsort*** *για να ταξινομήσουμε τα αυτόματα κατά την είσοδο και κατά την έξοδο του αντίστοιχα*. Με αυτό τον τρόπο έχουμε δημιουργήσει το **T.fst.**

Βήμα 9

Για να δημιουργήσουμε έναν αποδοχέα για κάθε λέξη του αρχείου ανατρέχουμε το αρχείο 'test_greng.txt' και για κάθε λέξη που περιέχει φτιάχνουμε ένα αρχείο 'Gren_ac.txt'. Το αρχείο αυτό περιέχει τις μεταβάσεις από γράμμα σε γράμμα μέχρι το τέλος της λέξης στην μορφή που το δέχεται στην είσοδο του το fstcompile για να δημιουργήσει τον αποδοχέα ***W.fsa*** *. Μπορούμε να εκτελέσουμε το βήμα 9 μαζί με τα βήματα 10 και 11 με την εντολή:*

`$ make Vima9`.

Βήμα 10

Συνεχίζοντας από το βήμα 9 συνθέτουμε τον αποδοχέα ***W.fsa*** *με την μηχανή* ***T.fst*** *που δημιουργήσαμε στο βήμα 8 με την εντολή* ***fstcompose****. Στην συνέχεια βρίσκουμε το πιο σύντομο μονοπάτι για την συγκεκριμένη λέξη με την εντολή* ***fstshortestpath*** *και το αποθηκεύουμε σε ένα txt αρχείο ώστε να το διαχειριστούμε στην συνέχεια****.***

Βήμα 11

Γνωρίζοντας πλέον την πιο πιθανή ελληνική λέξη για την δεδομένη greeklish βρίσκουμε το ποσοστό επιτυχίας του αλγορίθμου μας μετρώντας τέσσερεις πιθανές περιπτώσεις.
Α) την περίπτωση επιτυχίας, δηλαδή την περίπτωση που η λέξη είναι σωστά μεταφρασμένη στα ελληνικά, στην οποία ανήκει το 88% Β) την περίπτωση που η greeklish λέξη δεν ταίριαξε με κάποια λέξη του λεξικού Τ ώστε να βρεθεί σύντομο μονοπάτι, στην οποία ανήκει το 6% Γ) την περίπτωση που η λέξη του greeklish κειμένου ήταν Αγγλική και αυτή μεταφράστηκε στα ελληνικά, στην οποία ανήκει το 0% και τέλος η περίπτωση Δ) στην οποία η λέξη μεταφράστηκε λάθος στην οποία ανήκει το 5% . Το μεταφρασμένο κείμενο αποθηκεύεται στο αρχείο Translated_text.txt το οποίο παρέχεται στο φάκελο που υποβάλαμε. Μπορούμε

να παρατηρήσουμε πως τα περισσότερα από τα λάθη που παρατηρούνται στο μεταφρασμένο κείμενο είναι μεταφράσεις του «I σε I» ενώ η σωστή μετάφραση θα ήταν «I σε H», κάτι το οποίο αναμέναμε αφού το I έχει πολύ μεγαλύτερη πιθανότητα μετάφρασης σε I αντί για H.

Βήμα 12*

Σαν βελτίωση του μοντέλου υλοποιούμε έναν ορθογράφο για τις Ελληνικές λέξεις.
ii)Μέσω του script Vima12a.py συγκρίνουμε τα δύο train κείμενα (train_wr.txt και train_co.txt) και εντοπίζουμε διάφορους τύπους λαθών (Εισαγωγής, Διαγραφής, Αντικατάστασης και Αναδιάταξης).
iii)Υπολογίζουμε το πλήθος εμφάνισης τους και βρίσκουμε την πιθανότητα εμφάνισης τους διαιρώντας με το συνολικό πλήθος λαθών.
iv) Δημιουργούμε έναν αποδοχέα I ο οποίος αντιστοιχίζει κάθε γράμμα στον εαυτό του με μηδενικό κόστος. Ο αποδοχέας αποθηκεύεται στο αρχείο ***I.txt.***
v)Όμοια με προηγουμένως δημιουργούμε έναν μετατροπέα E ο οποίος έχει τα εξής χαρακτηριστικά:

- *Έχει μια κατάσταση αποδοχής (1) και μία κατάσταση εκκίνησης.*
- *Στην τελική κατάσταση μπορούμε να φτάσουμε με μια εισαγωγή χαρακτήρα, δηλαδή με <eps> - char με κόστος τον αρνητικό λογάριθμο της πιθανότητας εισαγωγής του βήματος iii).*
- *Στην τελική κατάσταση μπορούμε να φτάσουμε με μια διαγραφή χαρακτήρα, δηλαδή με char - <eps> με κόστος τον αρνητικό λογάριθμο της πιθανότητας διαγραφής του βήματος iii).*
- *Στην τελική κατάσταση μπορούμε να φτάσουμε με μια αντικατάσταση χαρακτήρα, δηλαδή με char1 – char2 με κόστος τον αρνητικό λογάριθμο της πιθανότητας αντικατάστασης του βήματος iii).*
- *Στην τελική κατάσταση μπορούμε να φτάσουμε με μια αναδιάταξη δυο χαρακτήρων, δηλαδή με char1char2 - char2char1 η οποία αποτελείται από δύο μεταβάσεις μια από την αρχική κατάσταση στην μεταβατική κατάσταση με κόστος τον αρνητικό λογάριθμο της πιθανότητας αναδιάταξης του βήματος iii) και από την μεταβατική κατάσταση στην τελική κατάσταση με μηδενικό κόστος.*

vi)Δημιουργούμε τον μετατροπέα S1 ο οποίος επιτρέπει μια μόνο από τις παραπάνω καταστάσεις υπολογίζοντας διαδοχικά concatenation:

S1 = Concatenation [Concatenation [Closure(I) , E)] , Closure(I)]

vii)Με όμοιο τρόπο δημιουργούμε έναν μετατροπέα S2 που επιτρέπει δύο εισαγωγές χρησιμοποιώντας το concatenation του S1 με το I ως εξής:

Concatenation [S1 , Concatenation[E, Closure(I)]]

viii) Χρησιμοποιούμε τον μετατροπέα A1 του βήματος 5 της προπαρασκευής για το ελληνικό λεξικό ο οποίο αποτελεί έναν αποδοχέα για όλες τις ελληνικές λέξεις του λεξικού. Με αυτόν τον τρόπο η λάθος μεταφρασμένες λέξεις θα προσπαθήσουμε να διορθωθούν με την χρήση του λεξικού. Ο μετατροπέας αυτός δημιουργείται, όπως και όλα τα αρχεία της προπαρασκευής, με την εντολή:

```
$ make prelab
```

ix) Για να συγκρίνουμε τα αρχεία test_co.txt και test_wr.txt ακολουθούμε παρόμοια διαδικασία με τα βήματα 9-10-11. Αναλυτικότερα για κάθε λέξη δημιουργούμε έναν αποδοχέα της λέξης ο οποίος στην συνέχεια συντίθεται με την μηχανή (S1,L) ή με την

μηχανή (S2,L) ανάλογα με την επιλογή που δίνει ο χρήστης. Στην συνέχεια βρίσκουμε το ελάχιστο μονοπάτι για την λέξη αυτή όπως στο βήμα 10. Στην συνέχεια ελέγχουμε την απόδοση του ορθογράφου ως εξής:
Αν η λέξη παρέμεινε ίδια με την αρχική και η αρχική έχει σωστή ορθογραφία είτε η λέξη διορθώθηκε σωστά τότε θεωρούμε ότι ο ορθογράφος λειτούργησε σωστά.
Αν η λέξη δεν διορθώθηκε σωστά τότε θεωρούμε ότι ο ορθογράφος λειτούργησε λάθος.
Τα ποσοστά επιτυχία της ορθογραφίας για την μηχανή (W,S1,L) είναι 75% ενώ για την μηχανή (W,S2,L) 71%.

x) Τέλος αλλάζοντας το όνομα των αρχείων ανάγνωσης στο script ***vima12b.py*** *χρησιμοποιώντας σαν input κείμενο το κείμενο που δημιουργήθηκε από τον μεταφραστή στο βήμα 11 και σαν κείμενο ελέγχου το* ***'test_gr.txt'*** *βελτιώνουμε την απόδοση της μετάφρασης του αρχικού κειμένου* ***'test_gren.txt'*** *σε 85%.*

**Η εκτέλεση του βήματος 12 μπορεί να γίνει απευθείας με την εντολή:*

$ make Vima12

ή σε δύο βήματα (τα βήματα i) μέχρι ix) και ix) μέχρι x) αντίστοιχα):

$ make Vima12a
$ make Vima12b

Σημειώσεις:

1)Προτείνεται η παρακάτω σειρά εκτέλεσης των εντολών ώστε να μην δημιουργηθούν προβλήματα με έλλειψης αρχείων:

```
$ make prelab
$ make Vima9
$ make Vima12a
$ make Vima12b
```

2)Τα αρχεία ***infile****,* ***lex.txt*** *και* ***latin*** *προέρχονται από την προπαρασκευή και είναι απαραίτητα για την κατασκευή του G.fst και I.fst αντίστοιχα.*
Το αρχείο ***accept.py*** *είναι ο κώδικας για την δημιουργία των λεξικών της προπαρασκευής και είναι απαραίτητο για την εκτέλεση των βημάτων του εργαστηρίου.*
Το αρχείο ***chars.txt*** *περιέχει το λεξικό για την δημιουργία του L.fst (I.fst στο script μας) του βήματος 12iv).*