Επεξήγηση Συνάρτησης Parser

1. Βασικός Κορμός της συνάρτησης

Η συνάρτηση έχει γραφτεί σε Python και για να λειτουργήσει χρειάζεται στην μεταβλητή File Path στην αρχή του κώδικα να οριστεί το path του txt αρχείου που περιέχει το LP1.

Η λογική πίσω από τη συνάρτηση είναι να βρίσκουμε σταθερά ένα σημείο σε κάθε γραμμή ώστε να οριοθετούνται ανάλογα οι αναζητήσεις των συντελεστών αλλά και του δείκτη κάθε μεταβλητής. Για παράδειγμα, για την αντικειμενική συνάρτηση βρίσκουμε το σύμβολο '=' που θα υπάρχει πάντα μετά το όνομα της (π.χ. max z = x1 + …) και από εκεί να βρίσκουμε τον συντελεστή της πρώτης μεταβλητής αναζητώντας μέχρι την θέση πριν το 'x'. Για τον δείκτη αναζητάμε αμέσως μετά το 'x' και σταματάμε όταν βρίσκουμε το πρώτο σύμβολο '+' ή '-'.

Με την ιδιά λογική συνεχίζουμε και στην υπόλοιπη συνάρτηση αλλά αυτό που πρέπει να σημειωθεί είναι ότι η εύρεση των στοιχείων αυτών για την πρώτη μεταβλητή διαφέρει από τις υπόλοιπες μεταβλητές. Αυτό συμβαίνει γιατί στην πρώτη μεταβλητή οριοθετούμε την αναζήτηση από το '=' αλλά για την δεύτερη μεταβλητή δεν θα υπάρχει αυτό το σύμβολο πριν τον συντελεστή της. Οπότε για κάθε μεταβλητή μετά την πρώτη η αναζήτηση οριοθετείτε από τα σύμβολα '+' , '-' που πάντα θα προηγούνται του συντελεστή τους (π.χ. x1 + 3x2 >= 1).

Κατά αυτό το τρόπο μπορούμε να βρούμε όλα τα δεδομένα που χρειαζόμαστε χωρίς να μας ενδιαφέρει πόσες μεταβλητές περιέχει κάθε περιορισμός ή η αντικειμενική συνάρτηση μέχρις ότου καταλάβει η συνάρτηση πως η γραμμή τελείωσε, όταν συναντήσει τον χαρακτήρα '\n'.

Στην αρχή της συνάρτησης υπολογίζεται η αρχή των δεδομένων βασιζόμενοι στο γεγονός ότι η πρώτη γραμμή που περιέχει το σύμβολο 'x' είναι τα πρώτα μας δεδομένα. Υπολογίζεται επίσης το τέλος των δεδομένων, η γραμμή δηλαδή που περιέχει το 'end', οπού και η συνάρτηση ολοκληρώνει την εύρεση δεδομένων.

Στην παρακάτω εικόνα παρουσιάζεται συνοπτικά ο τρόπος λειτουργίας της συνάρτησης.

Εικόνα 1. Ο βασικός κορμός της συνάρτησης.

```
For i in range(Αρχη των data,Τελος των data):

    while(TRUE):

        if (Ειμαστε στην αντικειμενικη συναρτηση) ΚΑΙ
            (Στην πρωτη επαναληψη)
                --κωδικας

        if(Ειμαστε στην αντικειμενικη συναρτηση) ΚΑΙ
            ΟΧΙ(Στην πρωτη επαναληψη)
                --κωδικας

        if (Ειμαστε στην πρωτη γραμμη των περιορισμων) ΚΑΙ
            (Στην πρωτη επαναληψη)
                --κωδικας

        if (Ειμαστε στην πρωτη γραμμη των περιορισμων) ΚΑΙ
            ΟΧΙ(Στην πρωτη επαναληψη)
                --κωδικας

        if(Ειμαστε στις υπολοιπες γραμμες των περιορισμων) ΚΑΙ
            (Στην πρωτη επαναληψη)
                --κωδικας

        if(Ειμαστε στις υπολοιπες γραμμες των περιορισμων) ΚΑΙ
            ΟΧΙ(Στην πρωτη επαναληψη)
                --κωδικας
```

Όπως φαίνεται και στην εικόνα, η πρώτη γραμμή των περιορισμών διαχειρίζεται διαφορετικά από τις υπόλοιπες γιατί πρέπει να περιέχει κάποιο από τα: {s.t. , st , subject to} από τα οποία οριοθετούνται οι αναζητήσεις. Για τις υπόλοιπες αναζητήσεις οριοθετουνται από το πρώτο στοιχείο της κάθε γραμμής το οποίο θα είναι το πρώτο ψηφίο του συντελεστή της πρώτης μεταβλητής.

2. Βασικές Λειτουργίες της συνάρτησης

Α) Δημιουργία του LP2.txt

Όλα τα δεδομένα του προβλήματος αρχικά συγκεντρώνονται σε λίστες (εκτός του MinMax) καθώς όσο αναφορά την απλή εισαγωγή δεδομένων προσφέρουν μεγαλύτερη ταχύτητα και ευελιξία ως δομές δεδομένων.

Στο τέλος του προγράμματος μετατρέπονται σε Arrays, γίνεται transpose στο Eqin Matrix και το b Matrix και τυπώνονται στο αρχείο LP2.txt.

Β) Αντιμετώπιση κενών εσωτερικά στις γραμμές και αναμεσά στις γραμμές του αρχείου LP1.txt

Αρχικά μέσω του module linecache διαβάζουμε το κείμενο γραμμή προς γραμμή. Για κάθε γραμμή αφαιρούμε τα κενά ή συνεχίζουμε στο διάβασμα της επομένης γραμμής αν συναντήσουμε κενή γραμμή βασιζόμενοι στο εξής κώδικα:

Εικόνα 2. Αντιμετώπιση κενών

```
76 # Diasxizoyme grammh-pros-grammh ta dedomena toy LP1
77 for currentLine in range(startOfData,endOfData):
78
79
80     retrieved_line_with_spaces = linecache.getline(FilePath, currentLine)
81
82     # An anamesa sta dedomena yparxoyn kenes grammes pareleipse tis.(sta dedomena toy grammikoy problimatos kathe grammi exei estw ena x)
83     if retrieved_line_with_spaces.find('x') == -1:
84         continue
85
86     # Se kathe grammi afairoyme ta kena
87     retrieved_line = retrieved_line_with_spaces.replace(" ","")
```

C) Έλεγχοι

Κάθε φορά που εξετάζεται μια γραμμή ανάλογα από το είδος της γραμμής (αντικειμενική συνάρτηση, περιορισμός) γίνεται έλεγχος για την ύπαρξη των {min,max} στην αντικειμενική συνάρτηση, των πρόσημων {+,-} εκεί που πρέπει, τα δεξιά μέρη των περιορισμών και των συμβολών {>=,<=,=} στους περιορισμούς.

Ένας ακόμη σημαντικός έλεγχος που γίνεται είναι για τους περιορισμούς: Οπότε συναντάμε περιορισμούς της μορφής

π.χ. x1 + x5 >= 2

η συνάρτηση φροντίζει να θέσει στον matrix των συντελεστών A, στην γραμμή του περιορισμού αυτού, ένα μηδενικό για κάθε ένα από τα x2, x3, x4 χωρίς να χρειάζεται να γραφεί ο χρήστης 0x2 + 0x3 + 0x4

Αν για το ίδιο παράδειγμα η αντικειμενική συνάρτηση είχε την εξής μορφή: z = x1 + x2 + x3 + x4 + x5 + x6 τότε επιπροσθέτως για τον παραπάνω περιορισμό η συνάρτηση θα θέσει 0 για τον συντελεστή x6 στην γραμμή του matrix A που αντιστοιχεί στον περιορισμό αυτό.

Η αντικειμενική συνάρτηση είναι αυτή που καθορίζει την μεταβλητή με τον μέγιστο δείκτη καθώς θεωρείται ότι αυτό συμβαίνει όταν το γραμμικό πρόβλημα είναι στην γενική του μορφή και δεν υπάρχουν χαλαρές και πλεονασματικές μεταβλητές.

D) Μαθηματική μορφή συντελεστών

Οι συντελεστές των μεταβλητών μπορούν να είναι θετικοί ή αρνητικοί, ακέραιοι ή δεκαδικοί. Ο έλεγχος για το αν ένας συντελεστής είναι δεκαδικός γίνεται με το εντοπισμό της τελείας '.' αναμεσά στα ψηφιά του. Το κόμμα και τα κλάσματα δεν επιτρέπονται.

3. Σύνταξη του LP1

Η σύνταξη είναι σχεδόν ιδιά με αυτή του Parser_example.txt που ανέβηκε στο compus.

Αρχικά θα πρέπει να υπάρχει ένα από τα: {min,max} στην αρχή της αντικειμενικής συνάρτησης. Το όνομα αυτής μπορεί να είναι οποιοδήποτε αλλά αμέσως μετά θα πρέπει να υπάρχει το σύμβολο '='.

Στην σειρά του πρώτου περιορισμού θα πρέπει να υπάρχει ένα από τα: {s.t. , st , subject to}.

Π.χ. min z = x1 + 3x2

st: x1 + x2 >= 2

3x1 + 2x2 = 2

Επίσης το αρχείο θα πρέπει να τελειώνει με τη λέξη 'end'.

Για κάθε ένα από αυτούς τους κομβικούς χαρακτήρες δίνεται το απαραίτητο μήνυμα εφόσον δεν υπάρχουν και η συνάρτηση τερματίζεται.

ΠΡΟΣΟΧΗ:

- Εκτός του ονόματος της συνάρτησης όλα τα υπόλοιπα γράμματα θα πρέπει να είναι μικρά και όχι κεφάλαια.
- Οι γραμμές των περιορισμών πρέπει να χωρίζονται με το Enter και όχι κόμματα.

- Η χρήση του Tab για ευθυγράμμιση των δεδομένων στο LP1.txt ενδεχομένως να δημιουργήσει προβλήματα στην λειτουργία της συνάρτησης.
- Το δεκαδικό μέρος των συντελεστών δηλώνεται με τελειά (.) και όχι κόμμα (,).
- Πρώτα θα πρέπει να γράφεται η αντικειμενική συνάρτηση και υστέρα οι περιορισμοί.

4. Λογισμικά, γλώσσα προγ/σμού και modules

Η συνάρτηση γράφτηκε με Python 3.8 μέσω του Eclipse IDE.

Εσωτερικά στην συνάρτηση έγινε χρήση των εξής modules:

a) Linecache: για διάβασμα γραμμή προς γραμμή του αρχείου εισόδου.

b) Numpy: για μετατροπή λιστών σε matrix, transpose και save στο LP2.txt

c) Os: για τον έλεγχο της περίπτωσης που το LP1.txt είναι κενό

d) Sys: για ‘προώρους’ τερματισμούς της συνάρτησης στην περίπτωση που υπάρχουν συντακτικά λάθη και την εκτύπωση στην κονσόλα του κατάλληλου μηνύματος

e) Time: για την διεξαγωγή test του χρόνου εκτέλεσης της συνάρτησης