ΠΑΝΕΠΙΣΤΗΜΙΟ ΜΑΚΕΔΟΝΙΑΣ
ΣΧΟΛΗ ΕΠΙΣΤΗΜΩΝ ΠΛΗΡΟΦΟΡΙΑΣ
ΤΜΗΜΑ ΕΦΑΡΜΟΣΜΕΝΗΣ ΠΛΗΡΟΦΟΡΙΚΗΣ

Computational Methodologies & Operations Research

artificial intelligence and data analytics

Εργασία: week - (05)
Ακαδημαϊκό Έτος : 2022-2023
Διδάσκων : Καθηγητής Νικόλαος Σαμαράς

Η ανακάλυψη του αλγορίθμου simplex οδήγησε σε μια σειρά από σημαντικά ερωτήματα και αποτελέσματα. Ο Borgwardt [1], [2] απέδειξε ότι ο αναμενόμενος αριθμός των επαναλήψεων για την επίλυση ενός γραμμικού προβλήματος από έναν αλγόριθμο τύπου simplex είναι πολυωνυμικός όταν χρησιμοποιείται σε πραγματικά προβλήματα. Νωρίτερα οι Klee-Minty [3] έδειξαν ότι η συμπεριφορά χειρότερης περίπτωσης του simplex είναι *εκθετική (exponential).*
Έχουν γίνει από τότε πολλές προσπάθειες για την εύρεση καλύτερων (πιο γρήγορων) τρόπων για την επίλυση γραμμικών προβλημάτων. Οι προσπάθειες αυτές οδήγησαν στην ανακάλυψη των αποτελεσματικών μεθόδων εσωτερικών σημείων [4], και [5]. Οι μέθοδοι αυτοί είναι ταχύτεροι από τον αλγόριθμο simplex σε προβλήματα μεγάλης διάστασης.
Παρόλα αυτά, η ανάπτυξη *περιστροφικών αλγορίθμων (pivoting algorithms),* οι οποίοι να είναι ταχύτεροι του simplex παραμένει ένα θέμα με μεγάλο επιστημονικό ενδιαφέρον. Ένας τρόπος για την βελτίωση της υπολογιστικής συμπεριφοράς των αλγορίθμων τύπου simplex είναι η μετακίνηση σε *μη-γειτονικές (non-adjacent)* βελτιώνουσες κορυφές. Η μετακίνηση αυτή μπορεί να επιτευχθεί αποφεύγοντας το σύνορο του πολυέδρου $P = \{x \mid Ax \leq b, x \geq 0\}$ και κατασκευάζοντας βασικές λύσεις οι οποίες δεν είναι εφικτές. Ένας τέτοιος αλγόριθμος ονομάζεται *αλγόριθμος simplex εξωτερικών σημείων (exterior point simplex algorithm).*

Ερωτήματα.

[Α]. Να γράψετε κώδικα στη γλώσσα προγραμματισμού python ο οποίος θα υλοποιεί τον αλγόριθμο Εξωτερικών σημείων**,** όπως αυτός παρουσιάστηκε στη διάλεξη 5. Συγκεκριμένα, πρέπει να προγραμματίσετε μόνο τη Φάση ΙΙ του αλγορίθμου εξωτερικών σημείων. Αν για ένα πρόβλημα χρειάζεται να υπολογιστεί μια εφικτή διαμέριση, τότε να χρησιμοποιήσετε τη `scipy.optimize.linprog` (https://docs.scipy.org/doc/scipy/reference/optimize.linprog-simplex.html)και να ορίσετε ότι θέλετε να εκτελεστεί μόνο η Φάση Ι. Στη συνέχεια, ο κώδικάς σας θα δέχεται το

πρόβλημα της Φάσης I και θα εκτελεί τη Φάση II. Το όνομα αρχείου του πηγαίου κώδικα που θα παραδώσετε να είναι στη μορφή *Επίθετο_week(5).py*.

[Β]. Να τρέξετε μια mini υπολογιστική μελέτη στα παρακάτω τυχαία αραιά (με πυκνότητα περίπου 10%) βέλτιστα γραμμικά προβλήματα. Πρώτα πρέπει να τα μετατρέψετε σε μορφή μητρών με την εργασία σας από την $1^{η}$ εβδομάδα. Για κάθε πρόβλημα πρέπει να καταγράφετε τον cpu χρόνο και το πλήθος των επαναλήψεων που έκανε ο αλγόριθμος. Δεν χρειάζεται η χρήση της Φάσης I από τη βιβλιοθήκη scipy.

ID	Name	Constraints	Variables	Optimal Value
1	sdata1_100x100	500	500	-143,4570
2	sdata1_200x200	200	200	-81,2455
3	sdata1_300x300	300	300	-98,9077
4	sdata1_400x400	400	400	-73,8266
5	sdata1_500x500	500	500	-80,5277

Σημείωση. Την αντίστροφη μπορείτε να την υπολογίζετε είτε με χρήση έτοιμης συνάρτησης στην Python, είτε με χρήση της eta-matrix.

Αναφορές.

[1]. Borgwardt, H.K. (1982). "Some distribution independent results about the asymptotic order of the average number of pivot steps in the simplex method", *Mathematics of Operations Research,* Vol. 7(3), pp. 441-462.

[2]. Borgwardt, H.K. (1982). "The average number of the pivot steps required by the simplex method is polynomial*", Zeitschrift fur Operational Research, Series A: Theory*, Vol. 26(5), pp. 157-177.

[3]. Klee, V., Minty, G. (1972). "How good is the simplex algorithm?", In Inequalities III, Shisha, O. (Ed.), Academic Press, New York, pp. 159-179.

[4]. Gondzio, J. (1992). "Splitting dense columns of constraint matrix in interior point methods for large-scale linear programming", *Optimization,* Vol. 24, pp. 285-297.

[5]. Gondzio, J. (1996). "Multiple centrality corrections in a primal-dual method for linear programming", *Computational Optimization and Applications,* Vol. 6, pp. 137-156.