Μάθημα: LSO

Ομαδική Εργασία: 3-4 Άτομα

Βαθμολογία: 30% Τελικού Βαθμού

Διορία: TBA (Εντός Ιανουαρίου 2019)

Στα πλαίσια της άσκησης θα επιλύσουμε ένα πρόβλημα δρομολόγησης οχημάτων, το οποίο σχετίζεται με την παροχή βοήθειας-υλικών σε διάφορα γεωγραφικά σημεία μετά από φυσικές καταστροφές.

Το επιχειρησιακό σενάριο έχει ως εξής:

1. Μετά από μία μεγάλη φυσική καταστροφή, καλούμαστε να μεταφέρουμε όσο το δυνατό γρηγορότερα, απαραίτητες προμήθειες σε ένα σύνολο 200 γεωγραφικών σημείων.
2. Κάθε γεωγραφικό σημείο καλύπτει διαφορετικά τμήματα του πληθυσμού, έτσι ώστε κάθε ένα από αυτά να έχει διαφορετικές ανάγκες προμηθειών
3. Για τη μεταφορά των προμηθειών χρησιμοποιούνται 25 ομογενή φορτηγά αυτοκίνητα με μέγιστο φορτίο 3 tn.
4. Κάθε φορτηγό ξεκινά τη διαδρομή του από τη κεντρική αποθήκη $d = \{0\}$ και επισκέπτεται διαδοχικά κάποια από τα 200 σημεία εξυπηρέτησης $N = \{1,2,3, \dots, n\}$
5. Οι διαδρομές όλων των φορτηγών ξεκινούν ταυτόχρονα
6. Κάθε σημείο εξυπηρέτησης ικανοποιείται από μία μόνο επίσκεψη ενός αποκλειστικά οχήματος. Επομένως, όταν ένα όχημα επισκέπτεται ένα σημείο εξυπηρέτησης, παραδίδει σε αυτό το σύνολο των απαιτούμενων προμηθειών.
7. Τα οχήματα ταξιδεύουν στο οδικό δίκτυο με περίπου με 35 km/hr
8. Σε κάθε σημείο εξυπηρέτησης $i \in N$, απαιτείται ένα χρονικό διάστημα 15 λεπτών για την εκφόρτωση των προμηθειών

Ο σχεδιασμός των διαδρομών πρέπει να ελαχιστοποιεί το χρόνο ολοκλήρωσης της παραλαβής των προϊόντων από το σημείο το οποίο θα εξυπηρετηθεί πιο αργά από όλα τα υπόλοιπα σημεία.

Οι κόμβοι του προβλήματος δημιουργούνται από τον παρακάτω κώδικα ο οποίος πρέπει να ενσωματωθεί στο πρόγραμμά σας. Οι συντεταγμένες δίνονται σε χιλιόμετρα. Ο πίνακας των αποστάσεων προκύπτει από τον υπολογισμό των Ευκλείδειων αποστάσεων μεταξύ των διάφορων κόμβων:

```
def BuildModel(self):
    random.seed(1)
    depot = Node(0, 50, 50, 0)
    self.allNodes.append(depot)
    totalPoints = 200
    for i in range (0, totalPoints):
      x = random.randint(0, 100)
      y = random.randint(0, 100)
      dem = 100*random.randint(1, 5)
      serviceTime = 0.25;
      point = Node(i + 1 , x, y, dem, servTime)
      self.allNodes.append(point)
```

1. Να παρέχει την υποδομή για την επίλυση του εξεταζόμενου προβλήματος και να δημιουργεί το input (3)

    a. Κλάσεις (π.χ. Διαδρομή, Σημεία κτλ.)
    b. Πίνακες (Κόστος μεταξύ σημείων, Χρονικές Αποστάσεις…)

    Προφανώς δεν υπάρχει ένας μοναδικός (και σωστός τρόπος) για την αναπαράσταση του προβλήματος. Σκοπός που τίθεται είναι η πλήρης περιγραφή του μοντέλου, καθώς και η διευκόλυνση σας για την ανάπτυξη του αλγορίθμου βελτιστοποίησης στη συνέχεια.
2. Υλοποιείστε μία μέθοδο που θα δέχεται σαν όρισμα μία οποιαδήποτε λύση του προβλήματος και θα υπολογίζει την τιμή της αντικειμενικής συνάρτησης αυτής της λύσης. (1)
3. Κατασκευάστε έναν κατασκευαστικό αλγόριθμο, ο οποίος να δημιουργεί μία αρχική λύση. (2)
4. Σχεδιάστε ένα αλγόριθμο τοπικής έρευνας ο οποίος να χρησιμοποιεί την κίνηση μετακίνηση ενός πελάτη (relocation) (3)
    a. Σε πόσες επαναλήψεις η μέθοδός σας παγιδεύεται σε τοπικό ελάχιστο;
    b. Ποιο είναι η είναι το κόστος του τοπικού ελάχιστου και ποιο το σύνολο των διαδρομών;
5. Σχεδιάστε ένα VND αλγόριθμο ο οποίος θα χρησιμοποιεί τρεις τελεστές τοπικής έρευνας (1)