ΠΟΛΥΤΕΧΝΕΙΟ ΚΡΗΤΗΣ

ΣΧΟΛΗ ΜΗΧΑΝΙΚΩΝ ΠΑΡΑΓΩΓΗΣ ΚΑΙ ΔΙΟΙΚΗΣΗΣ

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Μεθευρετικός αλγόριθμος αποδοχής κατωφλίου για το ανοιχτό πρόβλημα δρομολόγησης οχημάτων.

ΑΛΜΠΑΝΗΣ ΕΥΑΓΓΕΛΟΣ

Αριθμός Μητρώου : 2014010144

Επιβλέπων Καθηγητής : Ιωάννης Μαρινάκης

Χανιά 2021

ΕΥΧΑΡΙΣΤΙΕΣ

Αρχικά θα ήθελα να ευχαριστήσω τον επιβλέποντα καθηγητή μου κ. Μαρινάκη Ιωάννη και τον κ. Τσακιράκη Ελευθέριο για την άψογη συνεργασία που είχαμε και την καθοδήγηση και τα εφόδια που μου παρείχαν συνεχώς μέχρι την εκπόνηση της παρούσας διπλωματικής εργασίας. Στη συνέχεια θα ήθελα να ευχαριστήσω τους φίλους μου για όλες τις στιγμές και αναμνήσεις που προέκυψαν μέσα σε αυτά τα χρόνια. Τέλος την οικογένειά μου για την ψυχολογική και οικονομική στήριξη που μου παρείχαν όλα αυτά τα χρόνια.

ΠΕΡΙΕΧΟΜΕΝΑ

ΠΕΡΙΛΗΨΗ

Σε μια κοινωνία στην οποία ο ανταγωνισμός και οι απαιτήσεις των καταναλωτών συνεχώς αυξάνονται, η μελέτη της εφοδιαστικής αλυσίδας είναι πολύ σημαντική για μια επιχείρηση έτσι ώστε να είναι ανταγωνιστική, πιο αποδοτική και κερδοφόρα. Η παρούσα διπλωματική εργασία ασχολείται με το Ανοιχτό Πρόβλημα Δρομολόγησης Οχημάτων ή OVRP (Open Vehicle Routing Problem). Το συγκεκριμένο πρόβλημα βρίσκει εφαρμογή κυρίως σε εταιρίες στις οποίες δεν χρειάζεται τα οχήματα που χρησιμοποιούνται για τη διανομή να επιστρέφουν στην αποθήκη. Όλα τα οχήματα ξεκινούν από την αποθήκη και πρέπει να εξυπηρετήσουν όλους τους κόμβους (πελάτες) με τη ζήτηση του καθενός χωρίς να παραβιαστούν οι περιορισμοί χωρητικότητας των οχημάτων και μέγιστου χρόνου διαδρομής. Στόχος του προβλήματος είναι η εύρεση λύσης με όσο το δυνατόν λιγότερα οχήματα και την ελαχιστοποίηση της συνολικής απόστασης των διαδρομών. Για την εύρεση των αρχικών εφικτών λύσεων χρησιμοποιείται ο Αλγόριθμος Άπληστης Τυχαιοποιημένης Προσαρμοστικής Αναζήτησης γνωστός και ως GRASP (Greedy Randomized Adaptive Search Procedure). Στη συνέχεια για τη βελτίωση της αρχικής λύσης χρησιμοποιείται ο αλγόριθμος αποδοχής κατωφλίου (Threshold Accepted) σε συνδυασμό με τέσσερις αλγόριθμους τοπικής αναζήτησης (1-0 relocate, 1-1 exchange, opt , swap). Ο αλγόριθμος εφαρμόζεται σε παραδείγματα της βιβλιογραφίας και γίνεται σύγκριση με τα βέλτιστα αποτελέσματα που έχουν δημοσιευτεί. Για την υλοποίηση του κώδικα χρησιμοποιήθηκε η προγραμματιστική γλώσσα Matlab.

Κεφάλαιο 1 – Εισαγωγή

1.1 Εφοδιαστική

Ετυμολογικά, ο όρος Εφοδιαστική/Logistics έχει προέλευση από τον ελληνικό όρο «λόγος», που σημαίνει λογική, με την έννοια της εκλογίκευσης και σκοπό την επίτευξη ορισμένων συγκεκριμένων στόχων. Η εφοδιαστική έχει συμβάλλει ουσιαστικά στην παγκόσμια ανάπτυξη ήδη από τα αρχαία χρόνια και πιο συγκεκριμένα από την κατασκευή των πυραμίδων στην αρχαία Αίγυπτο. Τα Logistics αποτελούσαν στρατιωτική ανάγκη από τα αρχαία κιόλας χρόνια. Πάνω σε αυτά στηρίχθηκε η εκστρατεία του Μεγάλου Αλεξάνδρου . Κατά την διάρκεια του δευτέρου παγκοσμίου πολέμου, οι στρατιωτικές δυνάμεις εκμεταλλεύτηκαν τα υποδείγματα των logistics και τις μορφές ανάλυσης συστημάτων προκειμένου να διασφαλίσουν την ασφαλή μετακίνηση υλικού στα σημεία όπου χρειαζόταν. Ο όρος logistics εξακολουθεί ακόμα και στην σημερινή εποχή να χρησιμοποιείται ευρέως σε στρατιωτικές και στρατιωτικού τύπου καταστάσεις. Κατά την δεκαετία του ’60 και μετά την μεγάλη ανάπτυξη του εμπορίου μέσα από την εφαρμογή του συγκριτικού πλεονεκτήματος τα Logistics ξεκίνησαν να εφαρμόζονται και στον οικονομικό τομέα. Μερικοί από τους σύγχρονους κλάδους της εφοδιαστικής είναι οι εξής:

- επιχειρηματική εφοδιαστική (business logistics)
- διαχείριση υλικού (materials management)
- συστήματα ταχείας απόκρισης (quick-response systems)
- διαχείριση διανομών (distribution management)
- διαχείριση αποθηκών και κέντρων διανομών (ware-housing and storage)
- διαχείριση εφοδιαστικής αλυσίδας (supply chain management)

Η παρούσα εργασία ανήκει στον κλάδο της διαχείρισης της εφοδιαστικής αλυσίδας [1][6][7].

1.2 Εφοδιαστική αλυσίδα

Μια εφοδιαστική αλυσίδα (supply chain) ή και δίκτυο εφοδιαστικής (logistics network) αποτελείται από όλα τα στάδια που εμπλέκονται μέχρι να ικανοποιηθούν οι απαιτήσεις του πελάτη. Συνεπώς, μια εφοδιαστική αλυσίδα αποτελείται κατά σειρά από τους προμηθευτές, τους κατασκευαστές, τις εγκαταστάσεις και τις αποθήκες διανομής, τους πωλητές λιανικής και τους πελάτες καθώς και τις πρώτες ύλες ή και έτοιμα προϊόντα που χρησιμοποιούνται για την εξυπηρέτηση των πελατών. Η αλυσίδα είναι δυναμική και κάθε στάδιό της εκτελεί διαφορετικές εργασίες. Το κάθε στάδιο έχει επίδραση, όπως και επιδρούν σε αυτό τα υπόλοιπα στάδια της αλυσίδας. Οι εφοδιαστικές αλυσίδες διαφέρουν μεταξύ τους. Επομένως, μπορεί να παρατηρηθεί μια αλυσίδα η οποία δεν αποτελείται από όλα τα στάδια που αναφέρθηκαν παραπάνω. Αυτό οφείλεται στο διαφορετικό σχεδιασμό της κάθε αλυσίδας [1].

1.3 Απόδοση της εφοδιαστικής αλυσίδας

Σε κάθε στάδιο ή και αλληλεπίδραση μεταξύ δύο ή περισσοτέρων σταδίων μιας εφοδιαστικής αλυσίδας δημιουργούνται έξοδα (κόστος υλικών, έξοδα μεταφορών , κόστος παραγωγής, κόστος αποθεμάτων κλπ). Τα μόνα έσοδα που προκύπτουν στην αλυσίδα είναι ο πελάτης. Η απόδοση μιας εφοδιαστικής αλυσίδας χαρακτηρίζεται από τρία κριτήρια:

- Το ολικό κόστος
- Το ολικό κέρδος
- Τον χρονικό κύκλο της αλυσίδας

Σκοπός της κάθε αλυσίδας είναι να είναι όσο το δυνατόν αποδοτικότερη. Αυτό επιτυγχάνεται με τη μεγιστοποίηση της ολικής της αξίας (overall value). Για να επιτευχθεί αυτό το ολικό κόστος (overall cost) μέσα στην αλυσίδα (εξοδα μεταφορών,κόστος παραγωγής κλπ) πρέπει να ελαχιστοποιηθεί είτε από κάθε στάδιο ξεχωριστά είτε μέσω των αλληλεπιδράσεων που λαμβάνουν χώρα. Αντίθετα, το ολικό κέρδος (overall profitability) που προκύπτει από τον πελάτη πρέπει να μεγιστοποιηθεί έτσι ώστε να κατανεμηθεί πάλι μέσα στην αλυσίδα. Τέλος, ο χρονικός κύκλος (cycle time) ορίζεται ως ο συνολικός χρόνος που χρειάζεται για να πραγματοποιηθεί η ολική διαδικασία από τις πρώτες ύλες στο έτοιμο προϊόν στον καταναλωτή. Υπολογίζεται ότι μόνο το 5% του κύκλου χρησιμοποιείται για την εκτέλεση της πραγματικής διαδικασίας. Μια εφοδιαστική αλυσίδα είναι πιο αποδοτική όσο αυξάνεται και η απόδοση του χρονικού της κύκλου. Η αύξηση της αποδοτικότητας μιας εφοδιαστικής αλυσίδας είναι αρκετά δύσκολη και πολύπλοκη διότι διάφορα στοιχεία της αλυσίδας έχουν διαφορετικούς και πολλές φορές αντικρουόμενους στόχους [1].

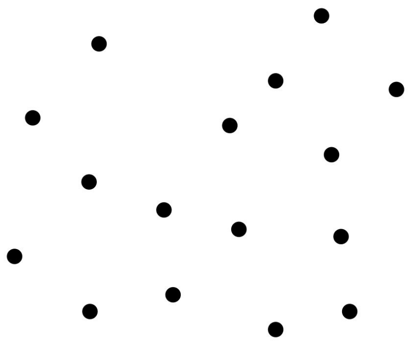
1.4 Το πρόβλημα δρομολόγησης οχημάτων στην εφοδιαστική αλυσίδα

Το τελευταίο στάδιο της εφοδιαστικής αλυσίδας πριν τον πελάτη είναι η διανομή. Η διανομή των αγαθών από τις επιχειρήσεις στους πελάτες γίνεται με οχήματα και ο λάθος προγραμματισμός αυτών μπορεί να επιφέρει πολλά έξοδα για την επιχείρηση ή και κακή εξυπηρέτηση του πελάτη. Έτσι η δρομολόγηση των οχημάτων είναι πολύ σημαντική μέσα στην εφοδιαστική αλυσίδα. Η παρούσα εργασία εστιάζει σε μια συγκεκριμένη κατηγορία προβλημάτων δρομολόγησης οχημάτων, το ανοικτό πρόβλημα δρομολόγησης οχημάτων γνωστο και ως OVRP (Open Vehicle Routing Problem) με περιορισμούς αυτούς της χωρητικότητας των οχημάτων και του μέγιστου επιτρεπτού χρονικού ορίου της κάθε διαδρομής. Στόχος είναι η ελαχιστοποίηση του κόστους και η χρήση όσο το δυνατόν λιγότερων οχημάτων. Τα παραπάνω εμφανίζονται αναλυτικότερα στα επόμενα κεφάλαια.

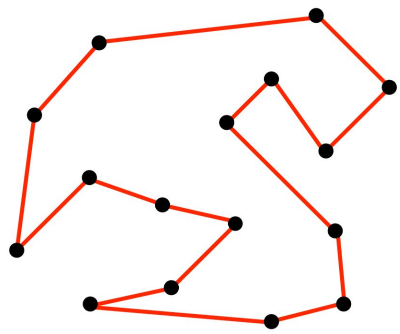
Κεφάλαιο 2 – Το πρόβλημα Δρομολόγησης Οχημάτων

2.1 Το πρόβλημα του πλανόδιου πωλητή

Το πρόβλημα του πλανόδιου πωλητή γνωστό και ως TSP (Travelling Salesman Problem) έχει ως στόχο την δημιουργία της καλύτερης διαδρομής με το μικρότερο δυνατό κόστος επισκεψιμότητας από όλους κόμβους. Στην διαδρομή αυτή ορίζεται ένα αρχικό σημείο ως αφετηρία, στο οποίο θα επιστρέψει το όχημα όταν εξυπηρετήσει ολόκληρο το σύνολο των διαθέσιμων πελατών. Ο πωλητής πρέπει να εξυπηρετήσει μέσα σε αυτή τη διαδρομή κάθε πελάτη ακριβώς μία φορά. Μια γραφική αναπαράσταση του προβλήματος του πλανόδιου πωλητή παρουσιάζεται στο κάτωθι σχήμα [8].



Σχήμα 1.1



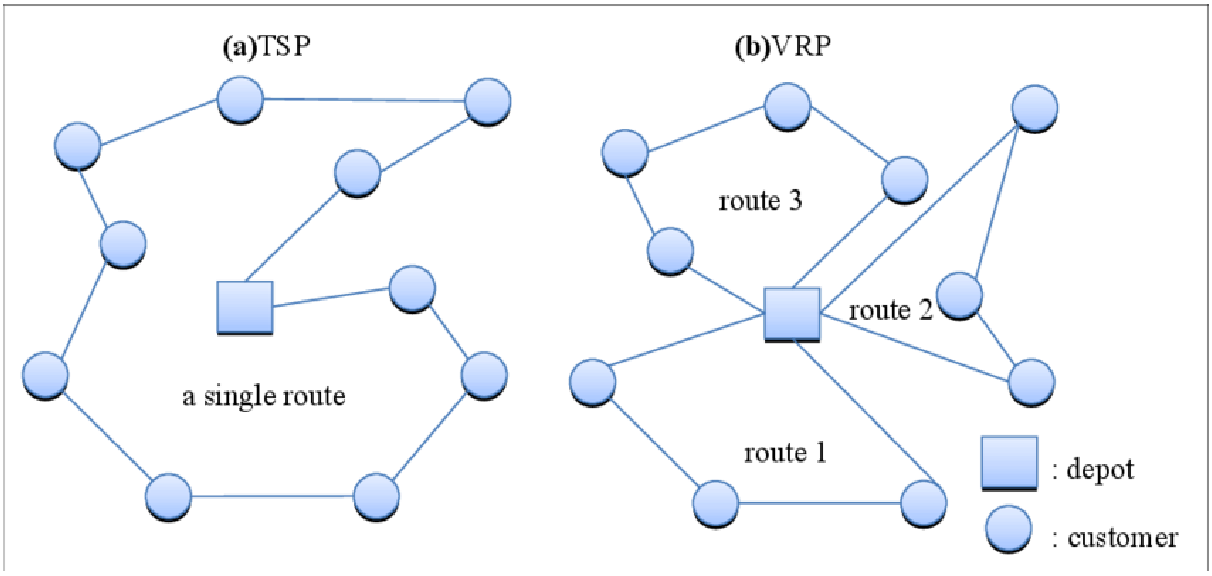
Σχήμα 1.2

Σχήμα 1: Επίλυση του προβλήματος του πλανόδιου πωλητή.
Πηγή:https://algorist.com/problems/Traveling_Salesman_Problem.html

Στο σχήμα 1.1 παρουσιάζονται οι κόμβοι (πελάτες) που είναι προς εξυπηρέτηση από τον πωλητή (όχημα). Στο σχήμα 1.2 παρουσιάζεται η λύση του προβλήματος αυτού με όλη τη διαδρομή που ακολουθεί ο πωλητής (όχημα). Παρατηρούμε ότι όλοι εξυπηρετούνται ακριβώς μία φορά και η διαδρομή τελειώνει στο σημείο αφετηρίας.

2.2 Το πρόβλημα δρομολόγησης οχημάτων

Το πρόβλημα δρομολόγησης οχημάτων γνωστό και ως VRP (Vehicle Routing Problem) είναι η επέκταση του προβλήματος του πλανόδιου πωλητή. Πιο συγκεκριμένα σε περιπτώσεις στις οποίες δεν μπορούν να εξυπηρετηθούν όλοι οι κόμβοι σε μια διαδρομή (από ένα όχημα) λόγω περιορισμών (χωρητικότητα οχήματος , μέγιστο επιτρεπτό όριο διαδρομής ή άλλος περιορισμός) και χρειάζονται περισσότερες, δημιουργούνται πολλαπλά TSP υποπροβλήματα έτσι ώστε να εξυπηρετηθούν όλοι οι κόμβοι. Σε αυτή την περίπτωση όλα τα TSP ξεκινούν από το ίδιο σημείο αφετηρίας και καταλήγουν στο ίδιο σημείο από το οποίο ξεκίνησαν. Το κάθε TSP εξυπηρετεί συγκεκριμένους κόμβους. Ο κάθε κόμβος εξυπηρετείται ακριβώς μία φορά άρα ανήκει μόνο σε ένα TSP. Στόχος του προβλήματος δρομολόγησης οχημάτων είναι όπως και στο πρόβλημα του πλανόδιου πωλητή η εξυπηρέτηση όλων των κόμβων ακριβώς μία φορά με το ελάχιστο κόστος χωρίς να παραβιάζεται κάποιος από τους περιορισμούς.



Σχήμα 2 a) Παράδειγμα TSP (μία διαδρομή) b) Παράδειγμα VRP (πολλαπλές διαδρομές)
Πηγή:https://www.researchgate.net/figure/Illustration-of-the-traveling-salesman-problem-TSP-and-vehicle-route-problem-VRP_fig1_277673931

Η μαθηματική μοντελοποίηση του βασικού προβλήματος δρομολόγησης οχημάτων όπως έχει παρουσιαστεί από τους Fisher & Jaikumar [3] παρουσιάζεται κάτωθι.

Έστω:

$$x_{i,j,k} = \begin{cases} 1, \text{ εάν το όχημα } k \text{ επισκέπτεται τον πελάτη } j \text{ αμέσως μετά τον πελάτη } i \\ 0, \text{αλλιώς} \end{cases}$$

$$y_{j,k} = \begin{cases} 1, \text{ εάν ο πελάτης } j \text{ επισκέπτεται από το σχήμα } k \\ 0, \text{ αλλιώς} \end{cases}$$

Αντικειμενική συνάρτηση:

$$min \sum_{i,j} c_{i,j} \sum_{k} x_{i,j,k}$$

Υπό περιορισμούς:

$$\sum_{k} y_{i,k} = \begin{cases} 1, & i=2,\ldots,n \\ m, & i=1 \end{cases} \quad (1)$$

$$\sum_{j} x_{i,j,k} = \sum_{j} x_{j,i,k} = y_{i,k} \quad , i = 1, \ldots, n \; , \; k = 1, \ldots, m \quad (2)$$

$$\sum_{i,j \in S} x_{i,j,k} \leq |S|-1 \quad , \text{για όλα τα } S \subseteq \{2,\ldots,n\} \quad (3)$$

$$y_{i,k} \in \{0, 1\} \quad , i=1,\ldots,n \; , k=1,\ldots,m \quad (4)$$

$$x_{i,j,k} \in \{0, 1\} \quad , i=1,\ldots,n \; , j=1,\ldots,n \; , k=1,\ldots,m \quad (5)$$

Ο περιορισμός (1) κατοχυρώνει ότι κάθε πελάτης εκχωρείται σε ένα μόνο όχημα εκτός από το σημείο αφετηρίας το οποίο επισκέπτονται όλα τα οχήματα. Ο περιορισμός (2) εξασφαλίζει ότι ένα όχημα που επισκέπτεται έναν κόμβο φεύγει από τον κόμβο αυτό [1] [8].

2.3 Παραλλαγές του προβλήματος δρομολόγησης οχημάτων

Ανάλογα με τις ανάγκες που έχουν δημιουργηθεί από τις επιχειρήσεις έχουν προκύψει πολλές παραλλαγές του προβλήματος δρομολόγησης οχημάτων. Οι αλλαγές αυτές αφορούν τους περιορισμούς που προσθέτει η εκάστοτε αλλαγή έτσι ώστε να προσαρμόζεται και να είναι επιλύσιμο. Μερικές από τις παραλλαγές του VRP αφορούν την επιστροφή ή μη των οχημάτων στο σημείο εκκίνησης. Αυτές είναι οι εξής:

- Το ανοικτό πρόβλημα δρομολόγησης οχημάτων ή OVRP (Open Vehicle Routing Problem)
- Το κλειστό πρόβλημα δρομολόγησης οχημάτων ή CVRP (Close Vehicle Routing Problem)
- Το ανοικτό-κλειστό πρόβλημα δρομολόγησης οχημάτων ή COVRP (Close Open Vehicle Routing Problem)

Μερικά από τα VRP προβλήματα που έχουν προκύψει βάσει άλλων περιορισμών είναι τα εξής:

- Το πρόβλημα δρομολόγησης οχημάτων με χρονικά παράθυρα ή VRPTW (Vehicle Routing Problem with Time Windows)
- Το πρόβλημα δρομολόγησης οχημάτων με πολλαπλές αποθήκες ή MDVRP (Multi Depot Vehicle Routing Problem)
- Το πρόβλημα δρομολόγησης οχημάτων με πολλαπλές αποθήκες και χρονικά παράθυρα ή MDVRPTW (Multi Depot Vehicle Routing Problem with Time Windows)

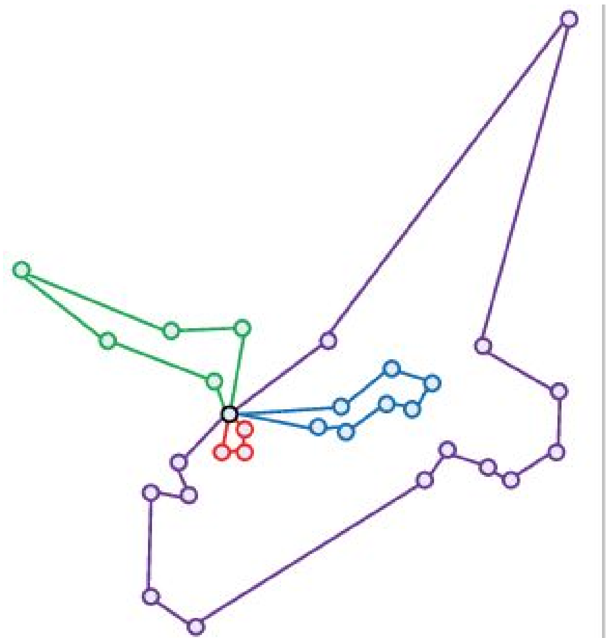
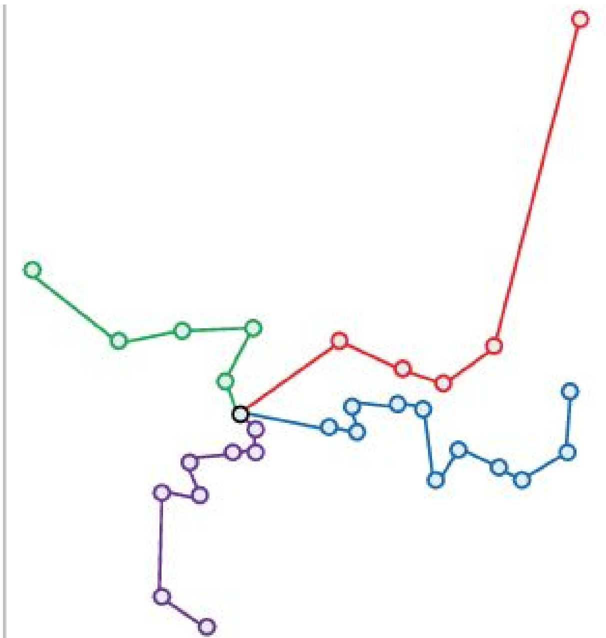
Προβλήματα δρομολόγησης οχημάτων προκύπτουν και από τον συνδυασμό ενός VRP προβλήματος με την επιλογή επιστροφής στην αποθήκη ή μη. Στην παρούσα διπλωματική εργασία μελετάται το ανοικτό πρόβλημα δρομολόγησης οχημάτων ή OVRP με περιορισμούς στην χωρητικότητα των οχημάτων αλλά και απόστασης που μπορεί να διανύσει το κάθε όχημα.

2.4 Οι περιορισμοί χωρητικότητας και απόστασης στο VRP

Το πρόβλημα δρομολόγησης οχημάτων με περιορισμούς αυτούς της χωρητικότητας και της απόστασης είναι ένα από τα πιο συνήθη προβλήματα δρομολόγησης οχημάτων καθώς οι περισσότερες επιχειρήσεις έρχονται αντιμέτωπες με αυτό. Η διαφορά του από το κλασσικό πρόβλημα δρομολόγησης οχημάτων έγκειται στους δύο περιορισμούς που προστίθενται. Ο πρώτος αφορά την χωρητικότητα των οχημάτων και έτσι το κάθε όχημα μπορεί να εξυπηρετεί πελάτες με τη ζήτηση του καθενός μέχρι το σύνολο της ζήτησης των πελατών που εξυπηρετούνται να μην ξεπερνά τη μέγιστη χωρητικότητα του οχήματος. Ο περιορισμός της απόστασης ή του χρόνου διαδρομής έχει να κάνει με τη χιλιομετρική απόσταση που διανύει το κάθε όχημα ή το χρόνο που χρησιμοποιείται αντίστοιχα. Έτσι στο πρόβλημα δρομολόγησης οχημάτων με περιορισμούς στη ζήτηση και στην απόσταση ή το χρόνο διαδρομής το όχημα που χρησιμοποιείται εξυπηρετεί πελάτες όσο η ζήτηση και ο χρόνος διαδρομής ή η χιλιομετρική απόσταση δεν υπερβαίνουν τα μέγιστα. Σκοπός του συγκεκριμένου προβλήματος είναι η ελαχιστοποίηση του κόστους. Ως συνολικό κόστος μπορεί να θεωρηθεί η συνολική απόσταση, ο συνολικός χρόνος, το σύνολο των οχημάτων που χρησιμοποιούνται ή και άλλες παράμετροι μέχρι να εξυπηρετηθούν όλοι οι πελάτες. Τα δεδομένα που χρειάζονται για την επίλυση του προβλήματος είναι το σύνολο των πελατών (κόμβων) με τη ζήτηση του καθενός και τη θέση που βρίσκεται, η αποθήκη με τη θέση της και το σύνολο των οχημάτων με την χωρητικότητα του καθενός και τη μέγιστη απόσταση που μπορεί να διανύσει ή το συνολικό χρόνο που μπορεί να χρησιμοποιείται [1].

2.5 Το ανοικτό πρόβλημα δρομολόγησης οχημάτων

Το ανοικτό πρόβλημα δρομολόγησης οχημάτων ή OVRP (Open Vehicle Routing Problem) δε διαφέρει πολύ από το κλασικό πρόβλημα δρομολόγησης οχημάτων. Η διαφορά τους βρίσκεται στην επιστοφή ή μη των οχημάτων στο σημείο αφετηρίας (αποθήκη). Στο OVRP τα οχήματα δεν επιστρέφουν στο σημείο από όπου ξεκίνησαν και χρησιμοποιούνται μέχρι να εξυπηρετηθεί και ο τελευταίος πελάτης (κόμβος). Όταν γίνει αυτό το πρόβλημα έχει φτάσει στη λύση του. Το συγκεκριμένο πρόβλημα βρίσκει εφαρμογή κυρίως σε εταιρείες οι οποίες δεν έχουν ιδιόκτητα οχήματα και για τη διανομή των προϊόντων τους χρησιμοποιούν ενοικιαζόμενα οχήματα ή σε περιπτώσεις στις οποίες τα ιδιόκτητα οχήματα που διαθέτει η εταιρεία δεν είναι κατάλληλα για τις συγκεκριμένες διανομές.

Σχήμα 3.1 Σχήμα 3.2

Πηγή:https://www.semanticscholar.org/paper/A-Heuristic-Approach-Based-on-Clarke-Wright-for-Pichpibul-Kawtummachai/5a245af049513b171e5667224a789c9c875834c5/figure/0

Στο σχήμα 3.1 απεικονίζεται η λύση CVRP προβλήματος της βιβλιογραφίας ενώ στο σχήμα 3.2 απεικονίζεται η λύση OVRP του ίδιου προβλήματος. Παρατηρείται εύκολα ότι στο 3.1 τα οχήματα επιστρέφουν στο σημείο αφετηρίας ενώ στο 3.2 οι διαδρομές τελειώνουν όταν εξυπηρετήσουν και τον τελευταίο πελάτη (κόμβο).

2.6 Μοντελοποίηση του ανοικτού προβλήματος δρομολόγησης οχημάτων με περιορισμούς στη χωρητικότητα και την απόσταση

Το ανοικτό πρόβλημα δρομολόγησης οχημάτων με περιορισμούς αυτούς της χωρητικότητας και της μέγιστης απόστασης διαδρομής είναι το πρόβλημα που μελετάται στην παρούσα διπλωματική εργασία. Το συγκεκριμένο πρόβλημα καθώς και οι περιορισμοί του μαζί με το στόχο του περιγράφηκε στις δυο προηγούμενες υποενότητες. Οι μεταβλητές που χρησιμοποιούνται για τη μαθηματική μοντελοποίηση του προβλήματος είναι οι εξής:

N: αριθμός πελατών (κόμβων συμπεριλαμβανομένης της αποθήκης)

K: ο αριθμός των οχημάτων

Q_k : μέγιστη χωρητικότητα του κάθε οχήματος

d_i : ζήτηση του πελάτη i

T_k : μέγιστη απόσταση διαδρομής κάθε οχήματος

$$x_{i,j,k} = \begin{cases} 1, \text{ εάν το όχημα } k \text{ επισκέπτεται τον πελάτη } j \text{ αμέσως μετά τον πελάτη } i \\ 0, \text{αλλιώς} \end{cases}$$

$$y_{j,k} = \begin{cases} 1, \text{ εάν ο πελάτης } j \text{ επισκέπτεται από το σχήμα } k \\ 0, \text{ αλλιώς} \end{cases}$$

$c_{i,j}$: η απόσταση (κόστος) μεταξύ των κόμβων i και j

Η μαθηματική μοντελοποίηση του προβλήματος που προκύπτει είναι η εξής:

Αντικειμενική συνάρτηση:

$$min \sum_{i,j} c_{i,j} \sum_k x_{i,j,k}$$

Υπό περιορισμούς:

$$\sum_k y_{i,k} = \begin{cases} 1, & i=2,...,n \\ m, & i=1 \end{cases} \quad (1)$$

$$\sum_i q_i y_{i,j} \le Q_k \quad , k = 1, ..., m \quad (2)$$

$$\sum_j x_{i,j,k} = \sum_j x_{j,i,k} = y_{i,k} \quad , i = 1, ..., n \; , \; k = 1, ..., m \quad (3)$$

$$\sum_i c_{i,j} y_{i,j} \le \mathrm{T}_k \quad , k = 1, ..., m \quad (4)$$

$$\sum_{i,j \in S} x_{i,j,k} \le |S|-1 \quad , \text{ για όλα τα } S \subseteq \{2,...,n\} \quad (5)$$

$$y_{i,k} \in \{0, 1\} \quad , i=1,...,n \; , k=1,...,m \quad (6)$$

$$x_{i,j,k} \in \{0, 1\} \quad , i=1,...,n \; , j=1,...,n \; , k=1,...,m \quad (7)$$

Ο περιορισμός (1) μας εξασφαλίζει ότι ο κάθε πελάτης εξυπηρετείται από ένα μόνο όχημα εκτός από την αποθήκη την οποία επισκέπτονται όλα τα οχήματα. Ο περιορισμός (2) είναι ο περιορισμός χωρητικότητας των οχημάτων. Ο περιορισμός (3) δείχνει ότι ένα όχημα που επισκέπτεται έναν πελάτη (κόμβο) φεύγει από αυτόν και ο περιορισμός (4) εξασφαλίζει να μην υπερβαίνεται η μέγιστη απόσταση που μπορεί να διανύσει το κάθε όχημα.

Κεφάλαιο 3- Αλγόριθμοι βελτιστοποίησης

3.1 Εισαγωγή στους αλγόριθμους βελτιστοποίησης

Με την πάροδο των χρόνων έχουν εμφανιστεί προβλήματα προς επίλυση της διαχείρισης της εφοδιαστικής αλυσίδας στα οποία η εύρεση της βέλτιστης λύσης είναι αρκετά δύσκολη και πολλές φορές είναι πρακτικά αδύνατο να βρεθεί. Στην συντριπτική πλειονότητα μάλιστα των περιπτώσεων είναι αδύνατο να βρεθεί η βέλτιστη λύση λόγω της πολυπλοκότητας του προβλήματος, του μεγέθους του ή κάποιου άλλου παράγοντα. Στόχος των αλγορίθμων βελτιστοποίησης είναι η εύρεση μιας λύσης της οποίας η ποιότητα είναι αρκετά καλή, αν όχι η βέλτιστη. Με τον όρο ποιότητα λύσης εννοείται μια λύση η οποία δεν έχει μεγάλη απόκλιση από τη βέλτιστη. Πολλές φορές βρισκόμαστε αντιμέτωποι με το ερώτημα εάν η λύση που προέκυψε είναι καλής ποιότητας. Για να εξετάσουμε αν είναι καλής ποιότητας ή όχι δημιουργούμε μικρότερα παραδείγματα από αυτά που είναι προς επίλυση στα οποία μπορούμε να βρούμε τη βέλτιστη λύση με κάποια ακριβή μέθοδο. Ο κάθε αλγόριθμος προσεγγίζει διαφορετικά το εκάστοτε πρόβλημα και έτσι οι λύσεις που παίρνουμε είναι διαφορετικής ποιότητας. Επομένως, έχουν δημιουργηθεί τρεις κατηγορίες αλγορίθμων βελτιστοποίησης που έχουν στόχο τη βέλτιστη λύση του εκάστοτε προβλήματος. Οι κατηγορίες αυτές είναι οι εξής:

- Ευρετικοί αλγόριθμοι
- Μεθευρετικοί αλγόριθμοι
- Εξελικτικοί αλγόριθμοι

Στην παρούσα εργασία ο προτεινόμενος αλγόριθμος έχει προκύψει από το συνδυασμό ευρετικών και μεθευρετικών αλγορίθμων, οι κατηγορίες αυτές καθώς και αλγόριθμοι που ανήκουν σε αυτές αναλύονται στη συνέχεια του κεφαλαίου [1].

3.2 Ευρετικοί αλγόριθμοι βελτιστοποίησης

Οι ευρετικοί αλγόριθμοι βελτιστοποίησης για τη διαχείριση της εφοδιαστικής αλυσίδας που έχουν αναπτυχθεί χωρίζονται σε τρεις κατηγορίες. Αυτές είναι οι εξής:

- Αλγόριθμοι απληστίας (greedy algorithms)
- Προσεγγιστικοί αλγόριθμοι (approximation algorithms)
- Αλγόριθμοι τοπικής αναζήτησης (local search algorithms)

Πιο συγκεκριμένα οι αλγόριθμοι απληστίας προσπαθούν να εξάγουν μια εφικτή λύση του προβλήματος, πολλές φορές όμως για να φτάσουν στη λύση αυτή απαιτείται πολύς χρόνος. Αυτό συμβαίνει διότι είναι μυωπικοί αλγόριθμοι, δηλαδή βλέπουν μόνο μπροστά. Οι προσεγγιστικοί αλγόριθμοι προσπαθούν να λύσουν το πρόβλημα χρησιμοποιώντας επιπλέον πληροφορία. Τέλος, οι αλγόριθμοι τοπικής αναζήτησης ως στόχο έχουν τη βελτίωση μιας αρχικής λύσης με κάποια συγκεκριμένη μέθοδο αναζήτησης. Στον προτεινόμενο αλγόριθμο οι εξελικτικοί αλγόριθμοι που χρησιμοποιούνται είναι αλγόριθμοι τοπικής αναζήτησης. Ανάλυση των αλγορίθμων αυτών γίνεται στις επόμενες υποενότητες [1].

3.2.1 Αλγόριθμοι τοπικής αναζήτησης

Η εφαρμογή των αλγορίθμων τοπικής αναζήτησης βασίζεται στην μέθοδο δοκιμής και σφάλματος η οποία αποτελεί και την αρχαιότερη μέθοδο βελτιστοποίησης. Οι αλγόριθμοι αυτοί είναι πάρα πολύ απλοί αλλά και πολύ επιτυχημένοι και βρίσκουν εφαρμογή σε πολλά προβλήματα της εφοδιαστικής αλυσίδας. Όπως αναφέρθηκε και παραπάνω η διαδικασία που ακολουθούν οι αλγόριθμοι αυτοί είναι οι εξής: ξεκινώντας από μία αρχική λύση επαναληπτικά υλοποιείται μια μεθοδολογία αναζήτησης μέχρι να βρεθεί μια καλύτερη λύση από την προηγούμενη. Προγραμματιστικά η γενική μεθοδολογία των αλγορίθμων τοπικής αναζήτησης αποτυπώνεται παρακάτω.

```
διαδικασία local_search
begin
        t μια αρχική λύση του προβλήματος
        do while βρίσκεται μια βελτιωμένη λύση (improve(t))
                t=improve(t)
        return t
end
```

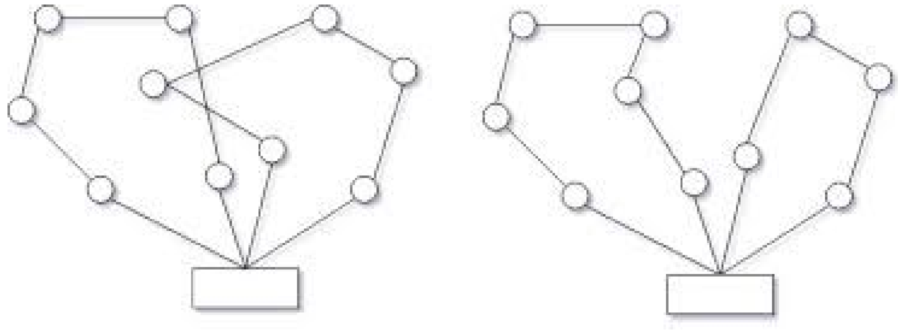
Μερικοί από τους αλγόριθμους τοπικής αναζήτησης είναι οι εξής:

- relocate (1-0 relocate, 2-0 relocate,...)
- exchange (1-1 exchange, 2-2 exchange,...)
- opt (1-opt,2-opt,3-opt,...)
- swap

Οι αλγόριθμοι τοπικής αναζήτησης που χρησιμοποιήθηκαν στη συγκεκριμένη εργασία είναι οι εξής: 1-0 relocate, 1-1 exchange, opt,swap [1].

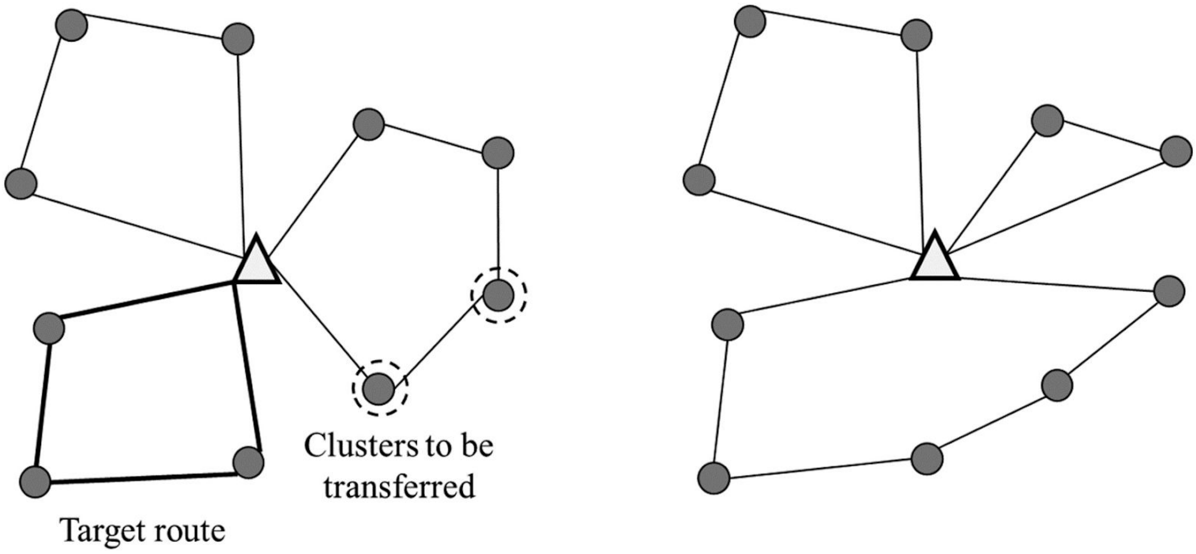
3.2.1.1 Ο αλγόριθμος relocate (επανατοποθέτησης)

Ο αλγόριθμος επανατοποθέτησης (relocate) προτάθηκε από τον Waters και έχει ως στόχο την απαλοιφή περιττών διαδρομών που έχουν δημιουργηθεί στην αρχική λύση αλλά και την ελαχιστοποίηση του συνολικού κόστους που προκύπτει από τη λύση του προβλήματος. Η εφαρμογή του συγκεκριμένου αλγορίθμου γίνεται μεταξύ δύο διαδρομών και γίνεται ως εξής: διαγράφεται ένας κόμβος από μια διαδρομή και τοποθετείται σε μια άλλη μειώνοντας έτσι το συνολικό κόστος των δυο αυτών διαδρομών και κατ' επέκταση το συνολικό κόστος. Ανάλογα με τον αριθμό των κόμβων που διαγράφονται από μια διαδρομή έχουν προκύψει και οι αντίστοιχοι αλγόριθμοι 1-0 relocate, 2-0 relocate και ούτω καθεξής. Στην παρούσα εργασία εφαρμόζεται ο 1-0 relocate.



Σχήμα 4: Παράδειγμα 1-0 relocate

Πηγή:https://www.google.com/url?sa=i&url=https%3A%2F%2Fdias.library.tuc.gr%2Fview%2Fmanf%2F82371&psig=AOvVaw36y5UGmTAVQ2IQl2xiQD6x&ust=1606481306034000&source=images&cd=vfe&ved=0CAIQjRxqFwoTCKj93P-goO0CFQAAAAAdAAAAABAD

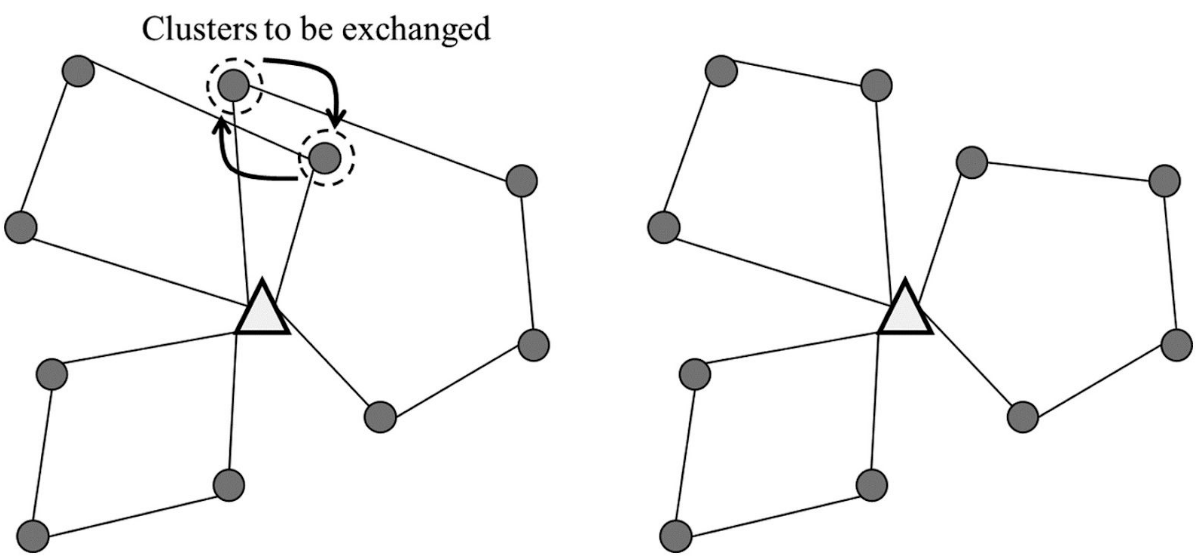


Σχήμα 5 Παράδειγμα 2-0 relocate

Πηγή: https://www.scielo.br/scielo.php?pid=S0104-530X2016000200279&script=sci_arttext&tlng=en

3.2.1.2 Ο αλγόριθμος exchange (ανταλλαγής)

Ο αλγόριθμος ανταλλαγής (exchange) προτάθηκε από τον Waters και εφαρμόζεται σε δύο διαδρομές ως εξής: διαγράφεται ένας κόμβος από κάθε διαδρομή και τοποθετείται στη θέση του άλλου. Με την κατάλληλη ανταλλαγή επιτυγχάνεται μείωση του αθροίσματος του κόστους των δυο αυτών διαδρομών άρα και του συνολικού κόστους στη λύση του προβλήματος που είναι και ο σκοπός του συγκεκριμένου αλγορίθμου όπως και όλων των αλγορίθμων τοπικής αναζήτησης. Παραπάνω περιγράφηκε ο αλγόριθμος 1-1 exchange. Ο αντίστοιχος 2-2 exchange προκύπτει αν από κάθε διαδρομή διαγραφούν δύο κόμβοι και γίνει ανταλλαγή μεταξύ αυτών και των δύο κόμβων της δεύτερης διαδρομής που εμπλέκεται στη διαδικασία. Στη συγκεκριμένη εργασία έγινε χρήση του αλγορίθμου 1-1 exchange.

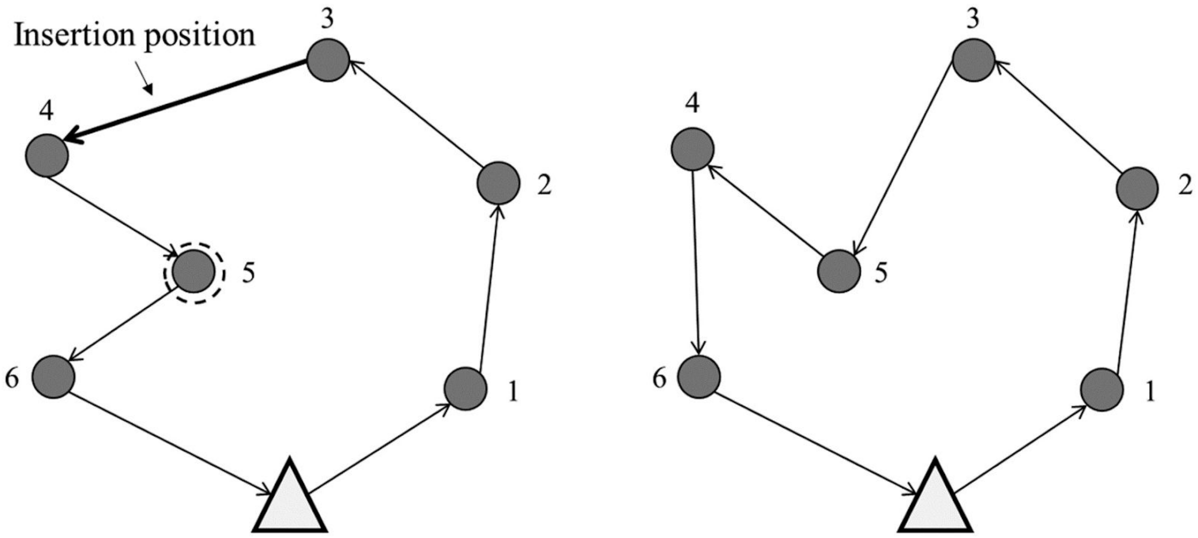


Σχήμα 6 Γραφική αναπαράσταση του αλγορίθμου 1-1 exchange

Πηγή: https://www.scielo.br/scielo.php?pid=S0104-530X2016000200279&script=sci_arttext&tlng=en

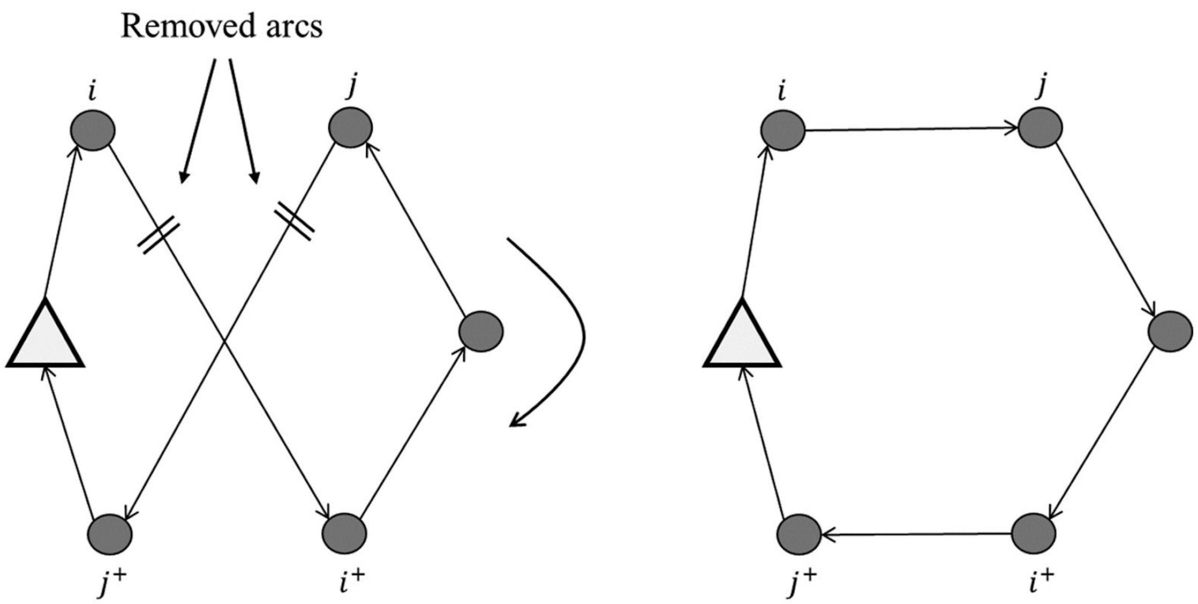
3.2.1.3 Ο αλγόριθμος opt

Ο αλγόριθμος τοπικής αναζήτησης opt προτάθηκε από τον Croes το 1958. Ο αλγόριθμος opt εφαρμόζεται σε μια διαδρομή. Οι ενέργειες που γίνονται για την βελτιστοποίηση της διαδρομής είναι με σειρά οι εξής: διαγράφεται ένα ή περισσότερα τόξα μέσα στη διαδρομή και δημιουργούνται καινούριες ακμές που ενώνουν τους κόμβους με αποτέλεσμα τη μείωση του κόστους της διαδρομής. Στην περίπτωση όπου δημιουργείται μόνο ένα καινούριο τόξο έχουμε τον αλγόριθμο 1-opt. Σε περίπτωση που δημιουργούνται δύο καινούρια τόξα έχουμε τον αλγόριθμο 2-opt και ούτω καθεξής. Στην παρούσα εργασία εφαρμόζεται ο αλγόριθμος opt με διάφορους αριθμούς τόξων που δημιουργούνται.



Σχήμα 7: Παράδειγμα Or-opt

Πηγή: https://www.scielo.br/scielo.php?pid=S0104-530X2016000200279&script=sci_arttext&tlng=en



Σχήμα 8: Παράδειγμα 2-opt

Πηγή: https://www.scielo.br/scielo.php?pid=S0104-530X2016000200279&script=sci_arttext&tlng=en

3.2.1.4 Ο αλγόριθμος swap

Ο αλγόριθμος τοπικής αναζήτησης swap είναι αρκετά παρόμοιος με τον opt. Πιο συγκεκριμένα ανταλλάσει τη θέση δύο κόμβων μέσα σε μια διαδρομή και έτσι δημιουργούνται καινούρια τόξα για να συνδεθούν όλοι οι κόμβοι και κατ' επέκταση προκύπτει ένα καινούριο κόστος διαδρομής. Στόχος είναι η ελαχιστοποίηση του κόστους αυτού.

3.3 Μεθευρετικοί αλγόριθμοι

Οι μεθευρετικοί αλγόριθμοι είναι μέθοδοι που χρησιμοποιούνται για την επίλυση προβλημάτων της εφοδιαστικής αλυσίδας και συνδυάζουν διαδικασίες τοπικής αναζήτησης με στρατηγικές υψηλότερου επιπέδου στρατηγικές με σκοπό τη δημιουργία μιας διαδικασίας που μπορεί να ξεφύγει από κάποιο τοπικό ελάχιστο. Τα περισσότερα σύγχρονα προβλήματα της εφοδιαστικής αλυσίδας λύνονται με μεθευρετικούς αλγόριθμους. Οι μεθευρετικοί αλγόριθμοι στις περισσότερες περιπτώσεις χρησιμοποιούν ευρετικούς αλγόριθμους ως υποδιαδικασίες τους. Επίσης πολλές φορές επιτρέπονται και βήματα στην ενδιάμεση διαδικασία τα οποία οδηγούν σε κάποια μη εφικτή λύση. Αυτό συμβαίνει με σκοπό να αποφευχθούν τοπικά ελάχιστα και να οδηγηθούμε σε μια καλύτερη τελική λύση. Οι αλγόριθμοι αυτοί συνήθως είναι βασισμένοι σε διαδικασίες οι οποίες έχουν εφαρμογή στη φύση. Τα στοιχεία που παρατηρούνται στους μεθευρετικούς αλγόριθμους αλλά και μεταφορικά στη φύση είναι τα εξής:

- Χρησιμοποιούν ένα αριθμό από επαναληπτικές δοκιμές
- Περιλαμβάνουν ένα ή περισσότερους πράκτορες (νευρώνες, μόρια, χρωμοσώματα,..)
- Λειτουργούν βάση ενός μηχανισμού συνεργασίας και ανταγωνισμού (στην περίπτωση που έχουμε πολλαπλούς πράκτορες)
- Περιλαμβάνουν διαδικασίες αυτο-τροποποιήσεων των ευρετικών παραμέτρων ή ακόμα και της αναπαράστασης του προβλήματος

Τα χαρακτηριστικά των αλγορίθμων αυτών είναι τα εξής:

- Μοντελοποιούν ένα φαινόμενο που υπάρχει στη φύση
- Μπορούν να μεταφερθούν εύκολα σε παράλληλη μορφή
- Είναι προσαρμοστικοί αλγόριθμοι

Μερικοί μεθευρετικοί αλγόριθμοι παρουσιάζονται κάτωθι:

- Προσομοιωμένη Ανόπτηση (Simulated Annealing (SA))
- Μέθοδος Αποδοχής Κατωφλίου (Threshold Accepted (TA))
- Περιορισμένη Αναζήτηση (Tabu Search (TS))
- Διαδικασία Άπληστης Τυχαιοποιημένης Προσαρμοστικής Αναζήτησης (Greedy Randomized Adaptive Search Procedure (GRASP))
- Επανασύνδεση Διαδρομών (Path Relinking (PR))
- Αλγόριθμος Επαναληπτικής Τοπικής Αναζήτησης (Iterated Local Search (ILS))
- Αλγόριθμος Καθοδηγούμενης Τοπικής Αναζήτησης (Guided Local Search (GLS))
- Αλγόριθμος Προσαρμοστικής Μνήμης (Adaptive Memory (AM))

Στον προτεινόμενο αλγόριθμο έγινε εφαρμογή του GRASP και της μεθόδου αποδοχής κατωφλίου (TA) , μεθευρετικοί αλγόριθμοι που αναλύονται στις επόμενες υποενότητες [1].

3.3.1 Η Διαδικασία Άπληστης Τυχαιοποιημένης Προσαρμοστικής Αναζήτησης (Greedy Randomized Adaptive Search Procedure- GRASP)

Η διαδικασία άπληστης τυχαιοποιημένης προσαρμοστικής αναζήτησης, γνωστή και ως GRASP είναι μια επαναληπτική διαδικασία για την εύρεση προσεγγιστικών λύσεων σε προβλήματα συνδυαστικής βελτιστοποίησης. Κάθε επανάληψη αποτελείται από δύο φάσεις, η πρώτη φάση είναι η κατασκευή μιας αρχικής λύσης και η δεύτερη είναι μια διαδικασία τοπικής αναζήτησης με σκοπό τη βελτιστοποίηση της αρχικής λύσης. Στη φάση κατασκευής, μια τυχαιοποιημένη συνάρτηση απληστίας χρησιμοποιείται για να κατασκευαστεί μια αρχική λύση. Αυτή η αρχική λύση στη συνέχεια βελτιώνεται με τη χρήση της διαδικασίας τοπικής αναζήτησης. Η υλοποίηση της φάσης της κατασκευής της αρχικής λύσης γινεται με την δημιουργία μιας λίστας υποψηφίων, που ονομάζεται λίστα περιορισμού των υποψηφίων και την τυχαία επιλογή του επόμενου κόμβου μέσα από αυτή τη λίστα. Εφόσον γίνει η επιλογή αυτή η λίστα ανανεώνεται για να επιλεγεί ο επόμενος κόμβος με τον ίδιο τρόπο. Η επιλογή των υποψηφίων που βρίσκονται μέσα σε αυτή τη λίστα γίνεται βάσει μιας συνάρτησης απληστίας. Στοιχεία που έχουν ήδη επιλεγεί σαν επόμενοι κόμβοι δεν είναι διαθέσιμα για την επιλογή σε επόμενες λίστες. Η κατασκευή της λίστας περιορισμού των υποψηφίων είναι ίσως το πιο σημαντικό κομμάτι της παρούσας μεθόδου, εφόσον από αυτή ελέγχεται η διασπορά των λύσεων που θα προκύψουν. Πιο συγκεκριμένα αν για παράδειγμα η λίστα επιλογής είναι πολύ μικρή τότε οι λύσεις που θα προκύψουν θα είναι αρκετά όμοιες μεταξύ τους, ενώ αν η λίστα είναι πολύ μεγάλη οι λύσεις θα παρουσιάζουν μεγάλη ποικιλομορφία εφόσον οι λύσεις θα κατασκευάζονται με τυχαίο τρόπο. Η αρχική λύση που δημιουργείται στην πρώτη φάση του αλγορίθμου δεν εγγυάται ότι είναι τοπικό ελάχιστο και για αυτό το λόγο εφαρμόζεται δεύτερη φάση του αλγορίθμου που είναι και αυτή της τοπικής αναζήτησης έτσι ώστε να παραχθεί μια λύση που έχει τοπικό ελάχιστο. Για να εφαρμοστεί η δεύτερη φάση της διαδικασίας, καθορίζεται μια συνάρτηση η οποία κάνει αναζήτηση στη γειτονιά της αρχικής λύσης. Το κάθε πρόβλημα χρειάζεται διαφορετική προσέγγιση (συναρτήσεις απληστίας, διαδικασίες τοπικής αναζήτησης, συναρτήσεις γειτονιάς που θα γίνει η τοπική αναζήτηση) για την εύρεση τοπικού ελαχίστου. Πολλές φορές για να βελτιωθεί η απόδοση της παραπάνω διαδικασίας μπορεί να εφαρμοστεί και μια τρίτη φάση στην οποία η καλύτερη ή οι καλύτερες λύσεις που έχουν προκύψει συνδυάζονται με την καινούρια λύση έτσι ώστε να προκύψουν ακόμη καλύτερες λύσεις. Παρακάτω παρουσιάζεται η αλγοριθμική διαδικασία της μεθόδου αυτής [2].

Διαδικασία Άπληστης Τυχαιοποιημένης Προσαρμοστικής Αναζήτησης

Αρχικοποίηση

c(s*)=∞

repeat

    Κατασκευή μιας αρχικής λύσης s

    Εφαρμογή τοπικής αναζήτησης στην s

    if c(s)<c(s*) **then**

        s*=s

    end if

until για όσο το κριτήριο τερματισμού δεν ικανοποιείται

Επέστρεψε την βέλτιστη λύση (s*)

3.3.2 Η μέθοδος Αποδοχής Κατωφλίου (Threshold Accepted (TA))

Η μέθοδος της αποδοχής κατωφλίου έχει ως στόχο η λύση να ξεφεύγει από κάποιο τοπικό ελάχιστο χρησιμοποιώντας ένα κατώφλι αποδοχής T ως πάνω όριο στην επιτρεπόμενη αύξηση της τιμής της αντικειμενικής συνάρτησης από τη μία κίνηση στην επόμενη. Έτσι, στη μέθοδο αποδοχής κατωφλίου, η γειτονιά N(s) καθορίζει καινούριες καταστάσεις που μπορεί να μεταβεί κάποιος από την τρέχουσα κατάσταση s. Από την αρχική κατάσταση s_0 με μια τυχαία διαταραχή του συστήματος δημιουργείται μια καινούρια λύση s′. Το αποτέλεσμα της αλλαγής στην τιμή της αντικειμενικής συνάρτησης υπολογίζεται ως εξής: Εάν το $\delta < T_i$ η καινούρια κατάσταση είναι πάντοτε αποδεκτή, όπου το T_i είναι το κατώφλι αποδοχής στην επανάληψη i. Το κατώφλι αποδοχής μπορεί να είναι σταθερό σε όλες τις επαναλήψεις ή να μειώνεται κατά τη διάρκεια των επαναλήψεων [2]. Ο αλγόριθμος της αποδοχής κατωφλίου είναι ο εξής:

Αλγόριθμος Αποδοχής Κατωφλίου

Αρχικοποίηση

Επέλεξε μια αρχική λύση s_0

Επέλεξε ένα αρχικό κατώφλι T=Tmax

Επέλεξε μία συνάρτηση μείωσης του κατωφλίου α(T)

```
repeat
    repeat
        Τυχαία επιλογή μιας γειτονιάς s∈N(s0)
        δ=f(s)-f(s0)
        if δ<T then
            s0=s
        end if
    until ο μέγιστος αριθμός επαναλήψεων ολοκληρωθεί
    T=α(T)
until κάποιο κριτήριο τερματισμού να ικανοποιηθεί
Επέστρεψε τη βέλτιστη λύση
```

Κεφάλαιο 4- Ο προτεινόμενος αλγόριθμος

Το αντικείμενο της παρούσας διπλωματικής εργασίας είναι το ανοικτό πρόβλημα δρομολόγησης οχημάτων με περιορισμούς χωρητικότητας και μέγιστου κόστους μετάβασης σε μία διαδρομή. Στο πρώτο μέρος του προτεινόμενου αλγορίθμου για την εύρεση των βέλτιστων λύσεων χρησιμοποιείται ο αλγόριθμος GRASP. Στο δεύτερο μέρος χρησιμοποιείται ο αλγόριθμος αποδοχής κατωφλίου (Threshold Accepted) σε συνδυασμό με τέσσερις τοπικές αναζητήσεις (1-0 relocate, 1-1 exchange, opt, swap) έτσι ώστε να παραχθεί η τελική λύση. Στόχος είναι η ελαχιστοποίηση του συνολικού κόστους (συνολικής απόστασης διαδρομών ή συνολικού χρόνου διαδρομών) μέχρι να εξυπηρετηθεί και ο τελευταίος πελάτης με όσο το δυνατόν λιγότερα οχήματα. Περεταίρω ανάλυση του αλγορίθμου γίνεται στις επόμενες υποενότητες. Ο κώδικας του προτεινόμενου αλγόριθμου γράφτηκε και υλοποιήθηκε στην προγραμματιστική γλώσσα Matlab.

4.1 Δεδομένα και επεξεργασία αυτών

Τα δεδομένα τα οποία χρησιμοποιήθηκαν για την πειραματική διαδικασία ανήκουν σε πακέτα δεδομένων της βιβλιογραφίας των οποίων οι βέλτιστες λύσεις που έχουν βρεθεί μέχρι στιγμής είναι γνωστές έτσι ώστε να μπορεί να γίνει σύγκριση με τις λύσεις που προκύπτουν από τον προτεινόμενο αλγόριθμο. Τα δεδομένα των πακέτων αυτών αποτελούνται από τον αριθμό των κόμβων (n), τη μέγιστη χωρητικότητα του κάθε οχήματος (capacity), το μέγιστο κόστος μετάβασης σε μια διαδρομή που δε μπορεί να ξεπεραστεί (Tmax), το χρόνο εξυπηρέτησης που είναι ίδιος για όλους τους κόμβους (service_time) τη ζήτηση του κάθε κόμβου i ($demand_i$) και τέλος τις x και τις y συντεταγμένες του κάθε κόμβου (x_i και y_i αντίστοιχα) συμπεριλαμβανομένης της αποθήκης. Τα δεδομένα για το κάθε πρόβλημα είναι συγκεκριμένα και διαφορετικά μεταξύ τους. Αρχικά τα παραπάνω δεδομένα αποθηκεύτηκαν σε αρχεία xlsx, ένα για το κάθε πρόβλημα και στη συνέχεια εισήχθησαν στην προγραμματιστική γλώσσα Matlab μέσω γνωστών συναρτήσεων. Οι αποστάσεις μεταξύ των κόμβων στο συγκεκριμένο πρόβλημα αποτελούν και το κόστος μετάβασης από τον ένα στον άλλο και θεωρούμε ότι είναι ευθείες γραμμές, έτσι το κόστος μετάβασης από τον κόμβο i στον κόμβο j υπολογίζεται βάσει της ευκλείδειας απόστασης ως εξής Cost(i,j) = $\sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}$. Έτσι δημιουργήθηκε ο πίνακας Cost διαστάσεων (N*N) του οποίου τα στοιχεία είναι τα κόστη μετάβασης από τον εκάστοτε κόμβο i στον εκάστοτε κόμβο j. Το στοιχείο i=1 ή j=1 θεωρείται η αποθήκη και η ζήτησή της είναι μηδενική.

4.2 Δημιουργία αρχικών λύσεων (πρώτη φάση αλγορίθμου)

Για την εξαγωγή αρχικών λύσεων στην παρούσα εργασία εφαρμόστηκε η Διαδικασία Άπληστης Τυχαιοποιημένης Προσαρμοστικής Αναζήτησης (Greedy Randomized Adaptive Search Procedure- GRASP). Οι διαδρομές των λύσεων αυτών ξεκινούν από την αποθήκη (κόμβος 1), εξυπηρετούν όσο το δυνατόν περισσότερους πελάτες (κόμβους) και τελειώνουν στον τελευταίο πελάτη που εξυπηρετείται. Στον αλγόριθμο που χρησιμοποιήθηκε δημιουργούνται διαδρομές μέχρι να εξυπηρετηθούν όλοι οι κόμβοι και η κατασκευή μιας καινούριας διαδρομής ξεκινά όταν τελειώσει η προηγούμενη. Ο κάθε επόμενος κόμβος που εισέρχεται στην τρέχουσα διαδρομή επιλέγεται τυχαία (ισοπίθανα) μέσα από μια λίστα (grasp_list) η οποία αποτελείται από έναν συγκεκριμένο αριθμό πιθανών κόμβων. Η εισαγωγή κόμβων σε αυτή τη λίστα δεν γίνεται τυχαία. Πιο συγκεκριμένα στη λίστα εισέρχονται οι κόμβοι που προσθέτουν το λιγότερο κόστος στη διαδρομή (απέχουν όσο το δυνατόν λιγότερο από τον τελευταίο κόμβο της τρέχουσας διαδρομής) και ταυτόχρονα δεν παραβιάζεται κάποιος από τους περιορισμούς (μέγιστης χωρητικότητας οχήματος και μέγιστου κόστους μετάβασης) με την εισαγωγή ενός από αυτούς στη διαδρομή που δημιουργείται. Όταν επιλεγεί ένας κόμβος εισέρχεται στην τελευταία θέση της διαδρομής και η λίστα αδειάζει. Έπειτα γίνεται εισαγωγή νέων κόμβων σε αυτή με τον ίδιο τρόπο που προαναφέρθηκε. Όταν δεν υπάρχει κάποιος πιθανός επόμενος κόμβος και επομένως η λίστα είναι κενή ξεκινά η επόμενη διαδρομή από την αποθήκη. Η εισαγωγή ενός οποιουδήποτε κόμβου σε μια οποιαδήποτε διαδρομή τον καθιστά εξυπηρετημένο και δεν είναι δυνατή η επιλογή του σε κάποια επόμενη λίστα πιθανών κόμβων. Η διαδικασία κατασκευής αρχικών λύσεων τελειώνει με την εξυπηρέτηση και του τελευταίου κόμβου. Με την τυχαιότητα που χαρακτηρίζει την δημιουργία των αρχικών διαδρομών παίρνουμε διαφορετικές εικόνες (διαδρομές) οι οποίες στη δεύτερη φάση του αλγορίθμου βελτιώνονται έτσι ώστε να φτάσουμε σε κάποιο τοπικό ελάχιστο.

4.3 Βελτίωση αρχικών λύσεων (δεύτερη φάση αλγορίθμου)

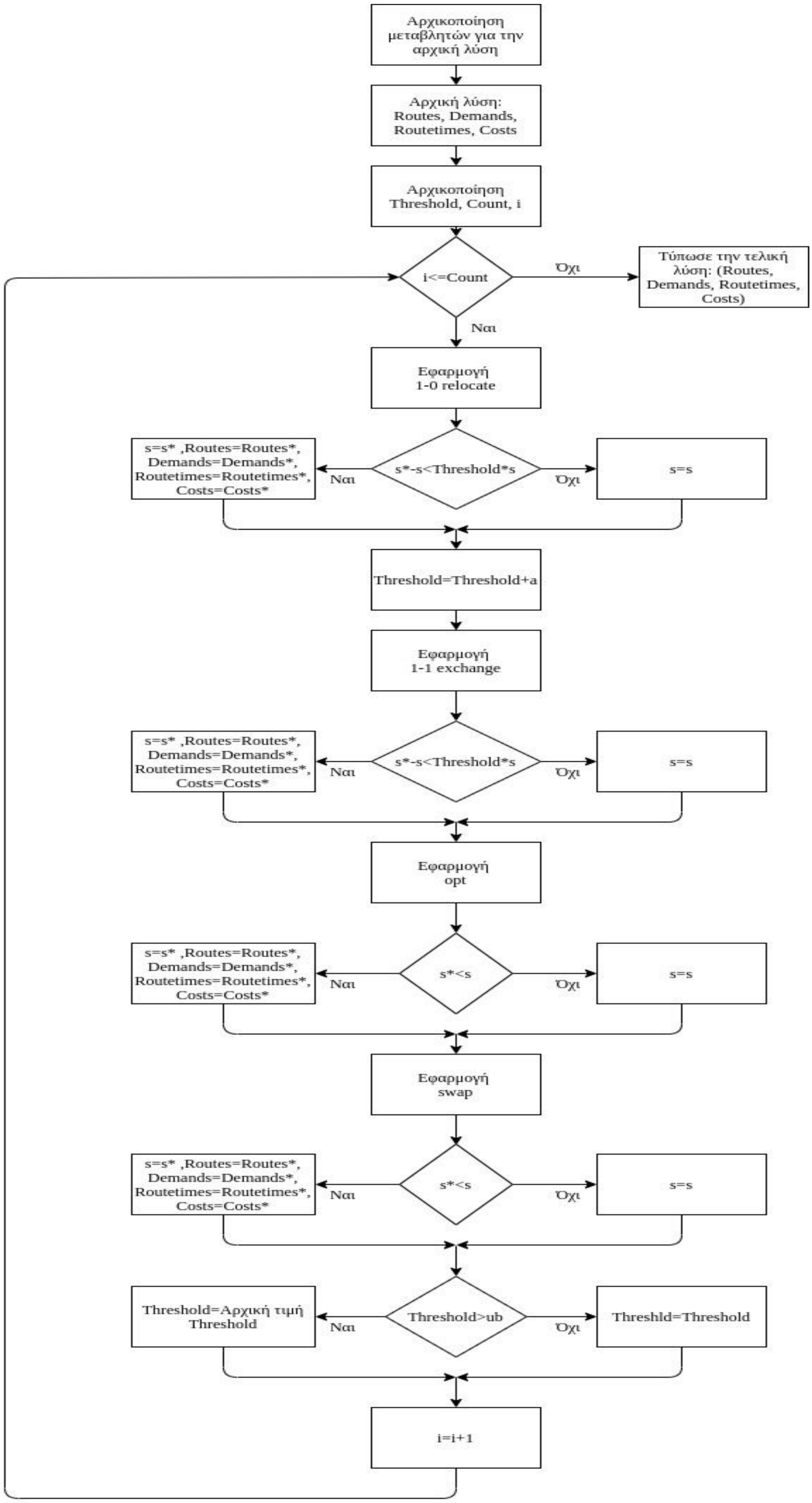
Η δεύτερη φάση του προτεινόμενου αλγορίθμου στοχεύει στη βελτίωση των αρχικών λύσεων που δημιουργούνται και δεν αποτελούν τοπικά ελάχιστα, φαινόμενο που οφείλεται στην τυχαιότητα που υπάρχει στην διαδικασία κατασκευής τους. Για τη βελτίωση αυτή συνδυάζονται τοπικές αναζητήσεις μαζί με την μέθοδο αποδοχής κατωφλίου (Threshold Accepted) στις αρχικές διαδρομές, με σκοπό την εύρεση κάποιου τοπικού ελαχίστου το οποίο θα αποτελέσει και την τελική λύση στο εκάστοτε πρόβλημα. Πιο συγκεκριμένα οι τοπικές αναζητήσεις που εφαρμόζονται κατά σειρά είναι οι εξής: 1-0 relocate, 1-1 exchange ,opt ,swap. Η κάθε τοπική αναζήτηση προσεγγίζει διαφορετικά τη βελτίωση των λύσεων. Αναλυτικότερα η τοπική αναζήτηση 1-0 relocate διαγράφει έναν κόμβο από μια διαδρομή και τον τοποθετεί σε κάποια άλλη με στόχο στην απαλοιφή περιττών διαδρομών καθώς και την μείωση του συνολικού κόστους των δυο αυτών διαδρομών. Ο αλγόριθμος 1-1 exchange όπως και ο 1-0 relocate εφαρμόζεται μεταξύ δύο διαδρομών και κατά την εφαρμογή του ανταλλάσσονται δύο κόμβοι, ένας από κάθε διαδρομή με στόχο τη μείωση του συνολικού κόστους των δύο διαδρομών. Με την εφαρμογή των δυο αυτών τοπικών αναζητήσεων επιτυγχάνεται η δημιουργία νέων διαδρομών οι οποίες έχουν περιθώριο βελτίωσης με τους δυο επόμενους αλγόριθμους τοπικής αναζήτησης. Τέλος η εφαρμογή των αλγορίθμων opt και swap γίνεται σε μία διαδρομή και μέσω της επανατοποθέτησης των κόμβων μέσα στην ίδια διαδρομή προκύπτει κόστος μικρότερο από το τρέχον. Η ακριβής επαναληπτική διαδικασία λοιπόν που ακολουθείται έχει ως εξής. Αρχικά ορίζεται ένα κατώφλι (Threshold) το οποίο καθορίζει αν η αλλαγή που προκύπτει από την κάθε τοπική αναζήτηση θα γίνει αποδεκτή ή όχι. Το κατώφλι αυτό αφορά ένα ποσοστό, το οποίο συνδέεται με τη μεταβολή του κόστους στη διαδρομή ή τις διαδρομές που εφαρμόζεται η κάθε τοπική αναζήτηση. Αναλυτικότερα, αν το καινούριο κόστος που προκύπτει ικανοποιεί τη σχέση new_cost-old_cost<Threshold*old_cost τότε η αλλαγή που προκύπτει από την συγκεκριμένη τοπική αναζήτηση γίνεται δεκτή και η τρέχουσα λύση αποτελείται από τις διαδρομές, τα κόστη, τις ζητήσεις και τους χρόνους διαδρομών που μολις προέκυψαν, αλλιώς δεν γίνεται κάποια αλλαγή. Το κατώφλι ορίζεται αρχικά σε κάποια τιμή μικρότερη της μονάδας και σε κάθε επανάληψη αυξάνεται με σταθερό ρυθμό μέχρι ένα συγκεκριμένο άνω φράγμα το οποίο είναι μεγαλύτερο της μονάδας. Όταν η τιμή του ξεπεράσει το άνω φράγμα το κατώφλι

ξαναπαίρνει την αρχική του τιμή. Αυτό έχει ως αποτέλεσμα την εξαγωγή πολλών και διαφόρων εικόνων στις λύσεις με στόχο την εύρεση της βέλτιστης. Ως βέλτιστη λύση αρχικά ορίζεται πριν από αυτή τη επαναληπτική διαδικασία η αρχική λύση που έχει προκύψει στο πρώτο στάδιο του αλγορίθμου. Η βέλτιστη αυτή λύση αντικαθίσταται κάθε φορά που προκύπτει κάποια λύση με συνολικό κόστος μικρότερο από αυτή μέσα στην παραπάνω επαναληπτική διαδικασία. Στο τέλος του αλγορίθμου παρουσιάζεται η λύση που έχει αποθηκευτεί ως βέλτιστη.

4.4 Μεταβλητές και πίνακες που χρησιμοποιούνται

1. n: αριθμός πελατών (κόμβων) συμπεριλαμβανομένης της αποθήκης
2. capacity: μέγιστη χωρητικότητα του κάθε οχήματος
3. Tmax: μέγιστο συνολικό κόστος μετάβασης σε μια διαδρομή
4. service_time: χρόνος εξυπηρέτησης του κάθε πελάτη (κόμβου)
5. x(n*1): x-συντεταγμένη του κάθε κόμβου
6. y(n*1): y-συντεταγμένη του κάθε κόμβου
7. demand(n*1): ζήτηση του κάθε πελάτη
8. Cost(n*n): πίνακας αποστάσεων μετάβασης μεταξύ κόμβων όπου Cost(i,j) κόστος μετάβασης από τον κόμβο i στον κόμβο j
9. used(1*n): διάνυσμα κόμβων που έχουν τοποθετηθεί σε διαδρομές (1 αν έχει τοποθετηθεί ο κόμβος, 0 αλλιώς)
10. used_grasp(1*n): διάνυσμα κόμβων που έχουν εισαχθεί ήδη στην τρέχουσα λίστα έτσι ώστε να μην ξανατοποθετηθούν (1 εάν ο κόμβος έχει τοποθετηθεί, 0 αλλιώς)
11. grasp_list_cost: λίστα με το κόστος μετάβασης από τον τρέχον κόμβο στον πιθανό επόμενο
12. grasp_list: λίστα υποψηφίων επόμενων κόμβων
13. vehicle: αύξων αριθμός τρέχουσας διαδρομής (αριθμός οχήματος που χρησιμοποιείται στην τρέχουσα διαδρομή)
14. routes: διαδρομές (grasp_routes , relocate_routes, exchange_routes, opt_routes, swap_routes, temp_routes, est_routes)
15. costs: κόστη διαδρομών (grasp_costs, relocate_costs, exchange_costs, opt_costs, swap_costs, temp_costs, best_costs)
16. demands: ζητήσεις διαδρομών (grasp_demands, relocate_demands, exchange_demands, opt_demands, swap_demans, temp_demands, best_demands)
17. routetimes: χρόνοι ολοκλήρωσης διαδρομών (grasp_routetimes, relocate_routetimes, exchange_routetimes, opt_routetimes, swap_routetimes, temp_routetimes, best_routetimes)
18. best_cost: βέλτιστο κόστος που έχει προκύψει

4.5 Διάγραμμα ροής του προτεινόμενου αλγορίθμου



Σχημα 9: Διάγραμμα ροής του προτεινόμενου αλγορίθμου όπου s το συνολικό κόστος της λύσης s* το συνολικό κόστος της καινούριας λύσης

Το διάγραμμα ροής της προηγούμενης σελίδας αφορά τον προτεινόμενο αλγόριθμο. Πιο συγκεκριμένα μέχρι το πρώτο στάδιο επιλογής υλοποιείται το πρώτο μέλος του αλγορίθμου που αφορά τα δεδομένα και την εξαγωγή των αρχικών λύσεων. Το υπόλοιπο διάγραμμα ροής αφορά το δεύτερο μέλος του αλγορίθμου με όλες τις διαδικασίες για την παραγωγή των τελικών λύσεων. Αναλυτικότερα απεικονίζεται όλη η αλγοριθμική διαδικασία που ακολουθείται μέχρι την εξαγωγή της τελικής λύσης, η οποία περιλαμβάνει τις τοπικές αναζητήσεις, την εφαρμογή του αλγορίθμου αποδοχής κατωφλίου και τις μεταβλητές με τις αλλαγές των τιμών τους.

Κεφάλαιο 5- Αποτελέσματα πειραματικής διαδικασίας

5.1 Αποτελέσματα πειραματικής διαδικασίας και σύγκριση

Πρόβλημα	Ελάχιστος αριθμός οχημάτων (διαδρομών) σύμφωνα με τη βιβλιογραφία	Αριθμός οχημάτων (διαδρομών) που χρησιμοποιούνται στη βέλτιστη λύση
vrpnc1	5	5
vrpnc2	10	10
vrpnc3	8	8
vrpnc4	12	12
vrpnc5	16	17 (+1)
vrpnc6	5	7 (+2)
vrpnc7	10	11 (+1)
vrpnc8	8	9 (+1)
vrpnc9	12	14 (+2)
vrpnc10	16	19 (+3)
vrpnc11	7	7
vrpnc12	10	10
vrpnc13	7	11 (+4)
vrpnc14	10	11 (+1)
F-n45-k4	4	4

Πίνακας 1: Σύγκριση αριθμού οχημάτων που χρησιμοποιούνται για τη λύση με τον αντίστοιχο της βιβλιογραφίας

Στον παραπάνω πίνακα που αφορά τον αριθμό των οχημάτων που χρησιμοποιούνται στην τελική λύση καθώς και τη διαφορά από τα οχήματα που χρησιμοποιούνται στην βέλτιστη λύση της βιβλιογραφίας, παρατηρούμε ότι στα προβλήματα τα οποία δεν έχουν περιορισμό στο μέγιστο κόστος μετάβασης μιας διαδρομής, δεν υπάρχει διαφορά στα οχήματα που χρησιμοποιούνται ενώ στα υπόλοιπα στις τελικές λύσεις που προέκυψαν από τον προτεινόμενο αλγόριθμο χρησιμοποιούνται από ένα έως τέσσερα οχήματα παραπάνω από αυτά στις λύσεις της βιβλιογραφίας.

Πρόβλημα	Βέλτιστη λύση βιβλιογραφίας	Βέλτιστη λύση	Απόκλιση
vrpnc1	406.994	432.9733	6.38%
vrpnc2	543.676	623.0794	14.6%
vrpnc3	617.0	686.385	11.25%
vrpnc4	729.684	770.6957	5.59%
vrpnc5	841.672	953.2388	13.26%
vrpnc6	412.95	428.7219	3.82%
vrpnc7	549.562	593.595	8.0%
vrpnc8	640.89	696.6763	8.7%
vrpnc9	741.44	839.5359	13.23%
vrpnc10	871.58	994.0996	14.06%
vrpnc11	672.582	760.2023	13.03%
vrpnc12	526.823	552.0246	4.78%
vrpnc13	836.55	924.1898	10.48%
vrpnc14	549.372	579.0659	5.41%
F-n45-k4	422.323	475.6407	12.62%

Πίνακας 2: Σύγκριση βέλτιστων λύσεων με τις αντίστοιχες της βιβλιογραφίας

Τα βέλτιστα αποτελέσματα με τα οποία έγιναν και οι παραπάνω συκγρίσεις παρουσιάζονται στη δημοσίευση OVRP_GELS: solving open vehicle routing problem using the gravitational emulation local search algorithm. Ali Asghar Rahmani Hosseinabadi• Javad Vahidi • Valentina Emilia Balas Seyed Saeid Mirkamali (2016). Παρατηρούμε ότι οι αποκλίσεις των τελικών λύσεων από τις βέλτιστες της βιβλιογραφίας κυμαίνονται από 3.82% έως και 14.6%. Η μικρότερη απόκλιση παρατηρείται στο πρόβλημα vrpnc6 ενώ η μεγαλύτερη στο vrpnc10.

5.2 Παραμετροποίηση προβλημάτων

Μετά από μεγάλο αριθμό επαναλήψεων για το μήκος της λίστας των υποψηφίων κόμβων καθώς και τον καθορισμό των φραγμάτων του κατωφλίου και τον ρυθμό αύξησης αυτού, για το κάθε πρόβλημα, καταλήξαμε σε συγκεκριμένη παραμετροποίηση για τις τελικές λύσεις. Οι τιμές που χρησιμοποιήθηκαν για την παραγωγή των τελικών λύσεων για το κάθε πρόβλημα παρουσιάζεται στον παρακάτω πίνακα.

Πρόβλημα	Μήκος λίστας υποψηφίων κόμβων	Κατώφλι (κάτω φράγμα/ άνω φράγμα)	Ρυθμός αύξησης φράγματος
vrpnc1	4	-0.01/0.2	0.0001 για κατώφλι<0 0.01 αλλιώς
vrpnc2	4	-0.01/0.2	0.0001 για κατώφλι<0 0.01 αλλιώς
vrpnc3	3	-0.01/0.2	0.0001 για κατώφλι<0 0.01 αλλιώς
vrpnc4	4	-0.01/0.2	0.0001 για κατώφλι<0 0.01 αλλιώς
vrpnc5	4	-0.01/0.2	0.0001 για κατώφλι<0 0.01 αλλιώς
vrpnc6	4	-0.01/0.2	0.0001 για κατώφλι<0 0.01 αλλιώς
vrpnc7	4	-0.01/0.2	0.0001 για κατώφλι<0 0.01 αλλιώς
vrpnc8	2	-0.01/0.2	0.0001 για κατώφλι<0 0.01 αλλιώς
vrpnc9	4	-0.01/0.2	0.0001 για κατώφλι<0 0.01 αλλιώς
vrpnc10	3	-0.01/0.2	0.0001 για κατώφλι<0 0.01 αλλιώς
vrpnc11	4	-0.01/0.2	0.0001 για

			κατώφλι<0 0.01 αλλιώς
vrpnc12	4	-0.01/0.2	0.0001 για κατώφλι<0 0.01 αλλιώς
vrpnc13	4	-0.01/0.2	0.0001 για κατώφλι<0 0.01 αλλιώς
vrpnc14	2	-0.01/0.2	0.0001 για κατώφλι<0 0.01 αλλιώς
F-n45-k4	1	-0.01/0.2	0.0001 για κατώφλι<0 0.01 αλλιώς

Πίνακας 3: Παραμετροποίηση προβλημάτων

Παρατηρούμε ότι το μήκος λίστας κυμαίνεται από μία έως τέσσερις θέσεις. Όσων αφορά το κατώφλι η παραμετροποίηση παραμένει ίδια για όλα τα προβλήματα. Μετά από μεγάλο αριθμό επαναλήψεων καταλήξαμε στο ότι οι συγκεκριμένες τιμές μας επιτρέπουν την εμφάνιση νέων εικόνων για την εύρεση νέων τοπικών ελαχίστων για όλα τα προβλήματα και με αυτό τον τρόπο ο προτεινόμενος αλγόριθμος έγινε πιο γενικός.

5.3 Παρουσίαση αποτελεσμάτων

Σε αυτή την ενότητα παρουσιάζονται οι βέλτιστες διαδρομές, με τις ζητήσεις, τις διάρκειες και τα κόστη τους όπως προέκυψαν από τον προτεινόμενο αλγόριθμο σε κάποια από τα προβλήματα στα οποία εφαρμόστηκε. Οι λύσεις παρουσιάζονται και γραφικά.

5.3.1 Το πρόβλημα vrpnc1

Αριθμός κόμβων: 50

Μέγιστη χωρητικότητα οχημάτων: 160

Μέγιστη απόσταση διαδρομής: -

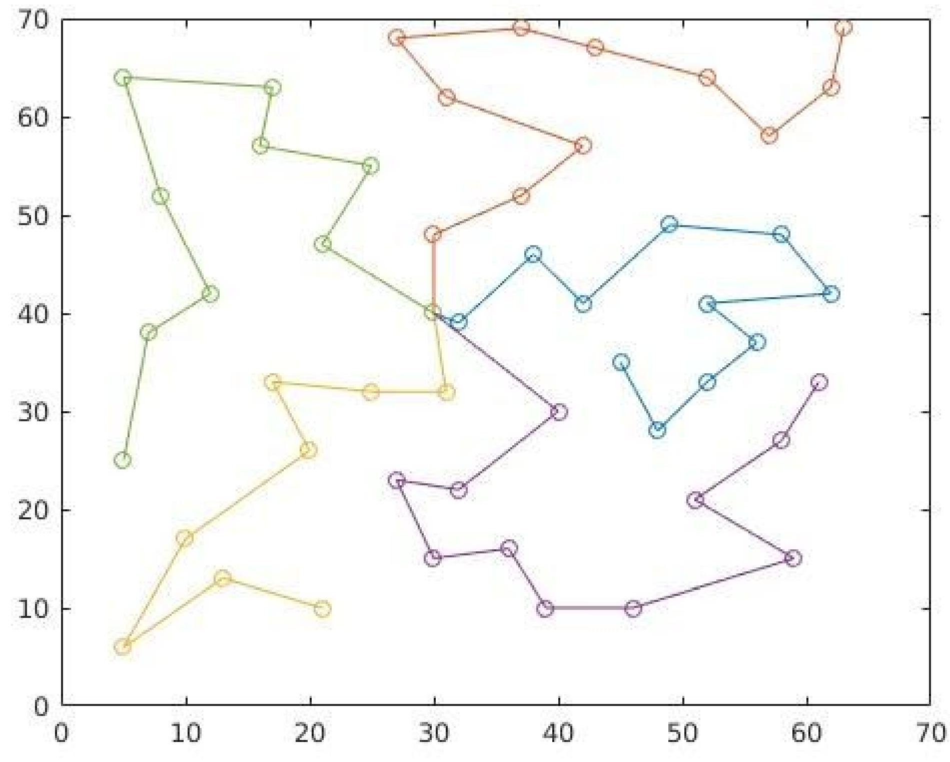
Χρόνος εξυπηρέτησης πελάτη: -

Διαδρομές:

1	47	33	12	3	30	22	17	51	10	50	39
1	28	2	23	9	27	32	29	4	21	36	37
1	13	48	19	5	42	41	20	43			
1	6	38	18	45	16	46	34	40	11	31	35

1	7	49	24	8	44	25	15	26	14

Αύξων αριθμός διαδρομής	**Ζήτηση**	**Κόστος μετάβασης**	**Κόστος**
1	149	80.1378	80.1378
2	152	89.2528	89.2528
3	160	74.4511	74.4511
4	156	98.7459	98.7459
5	160	90.3856	90.3856
		Συνολικό κόστος	**432.9733**



Σχήμα 10: Γραφική αναπαράσταση λύσεων για το πρόβλημα vrpnc1

5.3.2 Το πρόβλημα vrpnc4

Αριθμός κόμβων: 150

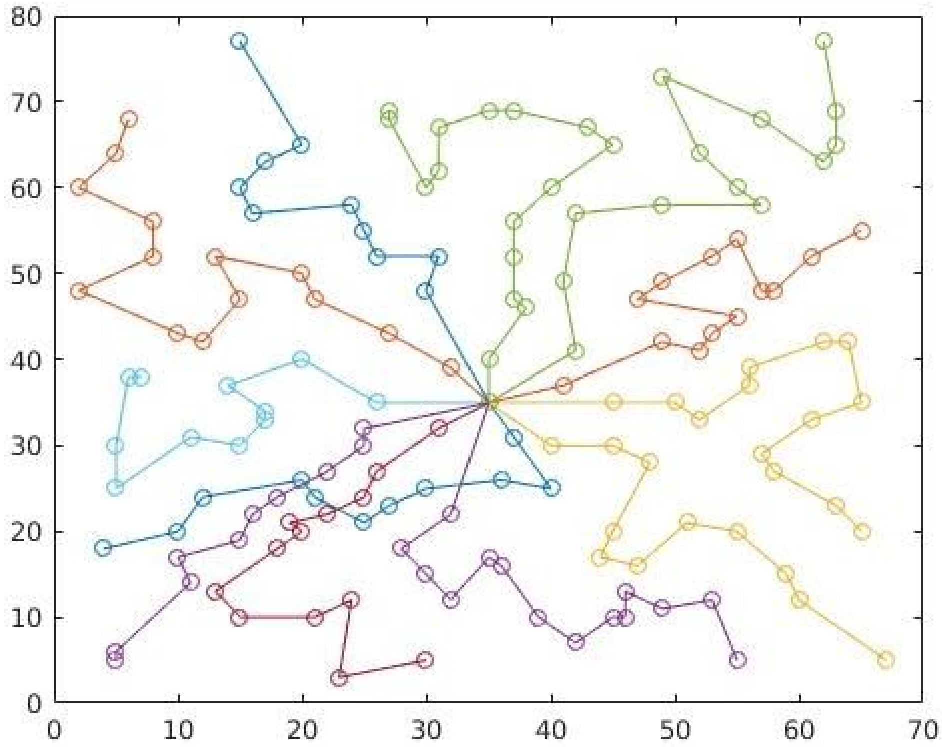
Μέγιστη χωρητικότητα οχημάτων: 200
Μέγιστη απόσταση διαδρομής: -
Χρόνος εξυπηρέτησης πελάτη: -

Διαδρομές:

1	54	41	59	14	118	98	60	105	100	62	17	87		
1	29	77	117	78	4	51	103	34	82	80	130	79	35	
1	139	13	110	81	151	69	122	30	25	135	55	131	56	26
1	138	88	145	58	3	116	146	42	23	134	75	76	57	24
1	28	133	70	102	71	31	21	129	132	33	91	109	11	127
1	90	19	84	61	119	6	85	114	18	46	126			
1	113	95	96	93	99	38	101	120	15	143	43	44	16	
1	128	32	89	149	63	124	20	108	12	65				
1	147	53	107	8	49	83	115	9	47	125	48	37	144	50
1	106	27	150	22	74	73	111	5	140	40	68			
1	148	7	97	94	86	92	142	45	141	39				
1	112	2	123	52	121	10	104	67	72	136	36	137	66	

Αύξων αριθμός διαδρομής	**Ζήτηση**	**Κόστος μετάβασης**	**Κόστος**
1	192	54.0991	54.0991
2	197	60.2129	60.2129
3	188	63.6382	63.6382
4	192	67.6570	67.6570
5	183	66.5986	66.5986
6	184	59.0372	59.0372
7	175	62.8614	62.8614
8	160	60.8116	60.8116
9	200	79.4880	79.4880
10	195	60.5363	60.5363

11	171	47.2210	47.2210
12	198	88.5337	88.5337
		Συνολικό κόστος	**770.6957**



Σχήμα 11: Γραφική αναπαράσταση λύσεων για το πρόβλημα vrpnc4

5.3.3 Το πρόβλημα vrpnc6

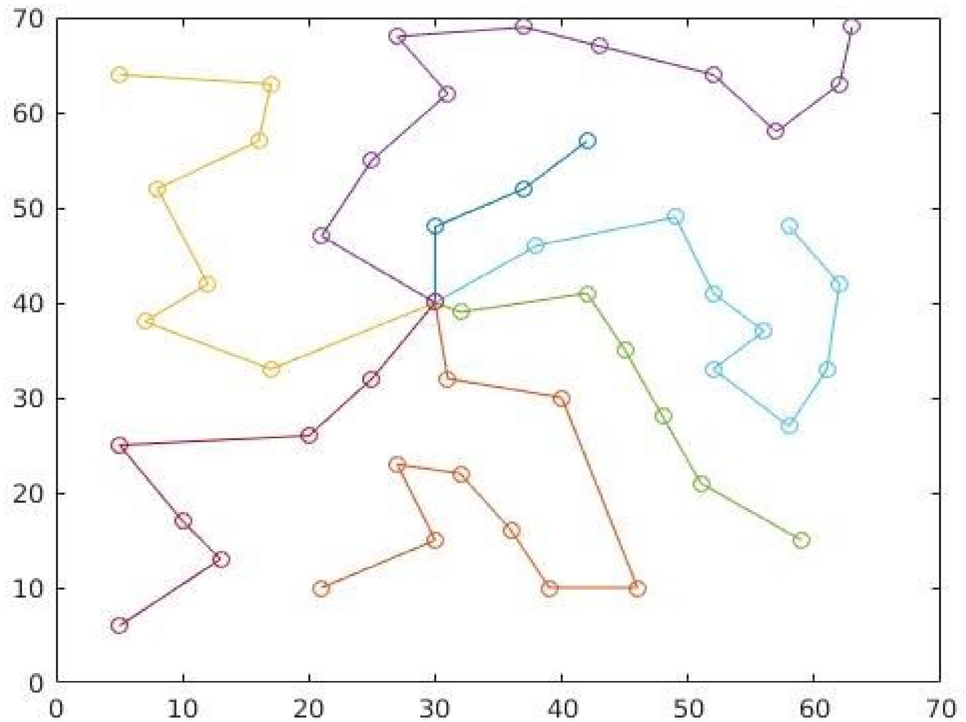
Αριθμός κόμβων: 50
Μέγιστη χωρητικότητα οχημάτων: 160
Μέγιστη απόσταση διαδρομής: 200
Χρόνος εξυπηρέτησης πελάτη: 10

Διαδρομές:

1	28	49	9	27	32	29	4	21	36	37
1	13	6	39	17	51	10	50	11	31	40
1	7	24	8	44	25	15	26			

1	19	14	42	41	20	43		
1	48	5	18	38	45	16	46	34
1	47	12	3	30	22	35		
1	33	2	23					

Αύξων αριθμός διαδρομής	**Ζήτηση**	**Κόστος μετάβασης**	**Κόστος**
1	154	179.8583	79.8583
2	157	180.1661	80.1661
3	120	140.2492	70.2492
4	120	129.8782	69.8782
5	105	137.0300	57.0300
6	94	108.3861	48.3861
7	27	53.1538	23.1538
		Συνολικό κόστος	**428.7219**



Σχήμα 12: Γραφική αναπαράσταση λύσεων για το πρόβλημα vrpnc6

5.3.4 Το πρόβλημα vrpnc7

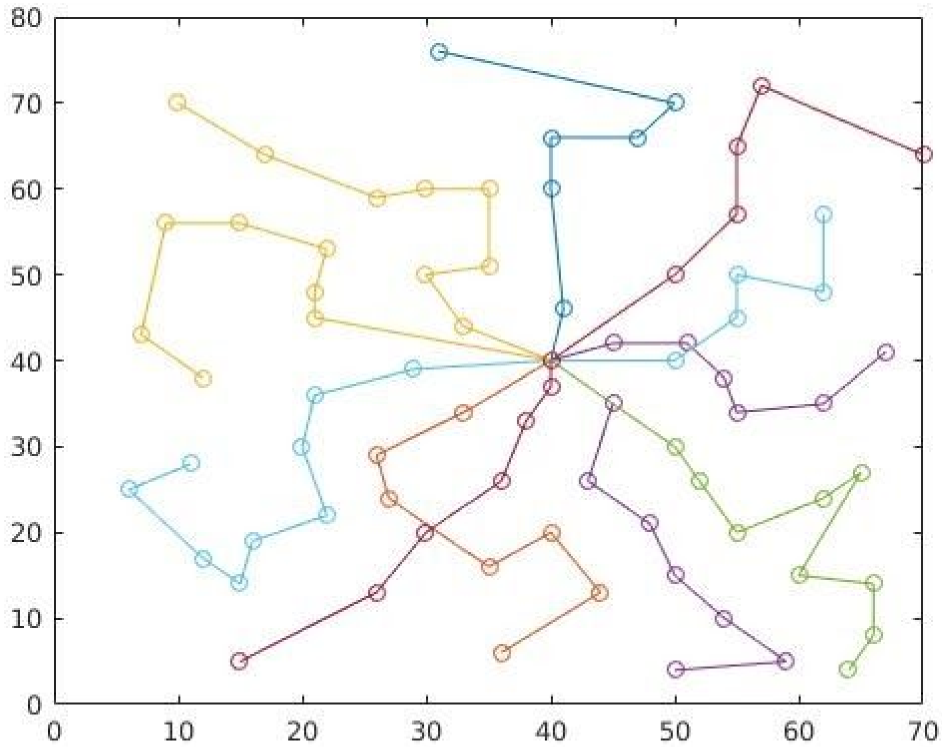
Αριθμός κόμβων: 75
Μέγιστη χωρητικότητα οχημάτων: 140
Μέγιστη απόσταση διαδρομής: 160
Χρόνος εξυπηρέτησης πελάτη: 10

Διαδρομές:

1	27	59	11	39	66	32			
1	18	41	13	73	40	10	26	56	
1	5	31	49	48	37	72	70		
1	46	30	6	16	58	38	21	71	61
1	52	17	64	2	44	43	42	57	24
1	8	54	12	67	60				
1	7	34	74	29	75	22	62		
1	4	45	33	51	19	25	50		
1	68	47	53	28	14	55			
1	35	9	36	20	15				
1	76	69	3	63	23	65			

Αύξων αριθμός διαδρομής	**Ζήτηση**	**Κόστος μετάβασης**	**Κόστος**
1	123	118.0432	58.0432
2	136	137.5077	57.5077
3	114	122.2158	52.2158
4	137	157.1822	67.1822
5	136	155.7991	65.7991
6	135	103.2889	53.2889
7	134	129.3301	59.3301
8	123	131.5856	61.5856
9	121	95.3895	35.3895

10	91	88.3511	38.3511
11	114	104.9012	44.9012
		Συνολικό κόστος	**593.5950**



Σχήμα 13: Γραφική αναπαράσταση λύσεων για το πρόβλημα vrpnc7

5.3.5 Το πρόβλημα vrpnc8

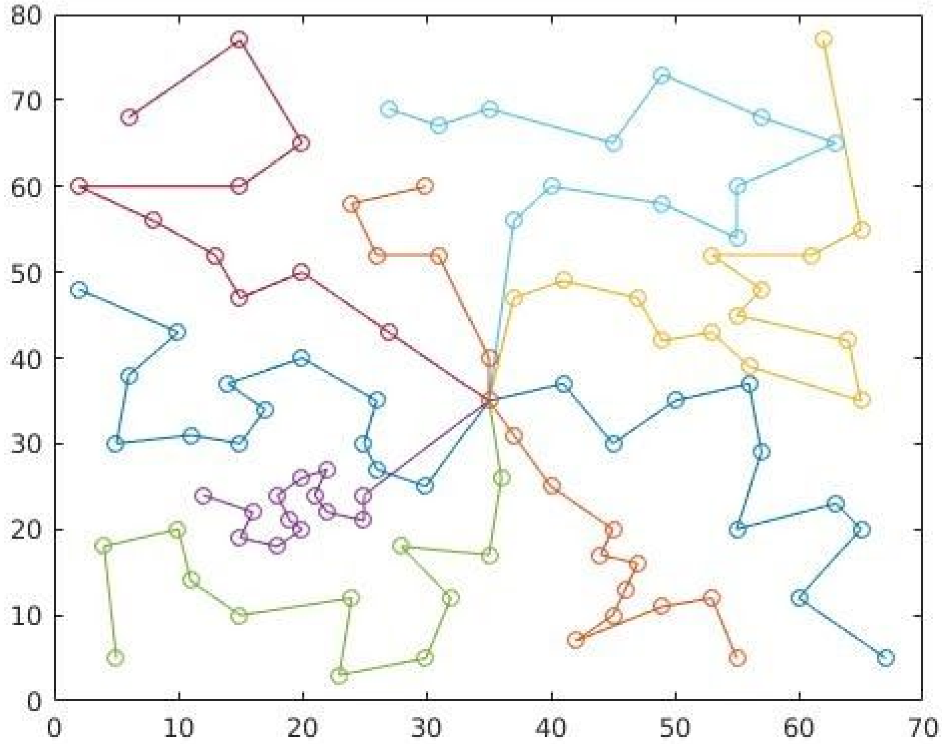
Αριθμός κόμβων: 100
Μέγιστη χωρητικότητα οχημάτων: 200
Μέγιστη απόσταση διαδρομής: 230
Χρόνος εξυπηρέτησης πελάτη: 10

Διαδρομές:

1	14	95	7	90	19	84	61	6	85	18	46	9	47	
1	54	41	22	74	73	75	23	42	76	57	24			
1	70	2	51	77	78	69	25	30	4	80	34	79	35	66
1	96	98	93	60	97	100	94	99	38	101	92	86	62	

1	59	3	88	58	16	44	43	15	45	17	87	39
1	71	31	52	82	10	36	72	67	21	33	91	64
1	53	8	83	49	48	37	20	12	65	50		
1	29	27	13	81	55	5	56	26	40	68		
1	28	32	89	63	11							

Αύξων αριθμός διαδρομής	**Ζήτηση**	**Κόστος μετάβασης**	**Κόστος**
1	155	211.2522	81.2522
2	152	164.6086	54.6086
3	188	248.3432	108.3432
4	194	179.6927	49.6927
5	186	216.3306	96.3306
6	171	222.7607	102.7607
7	166	191.8425	91.8425
8	159	176.5472	76.5472
9	87	85.2982	35.2982
		Συνολικό κόστος	**696.6763**



Σχήμα 14: Γραφική αναπαράσταση λύσεων για το πρόβλημα vrpnc8

5.3.6 Το πρόβλημα vrpnc12

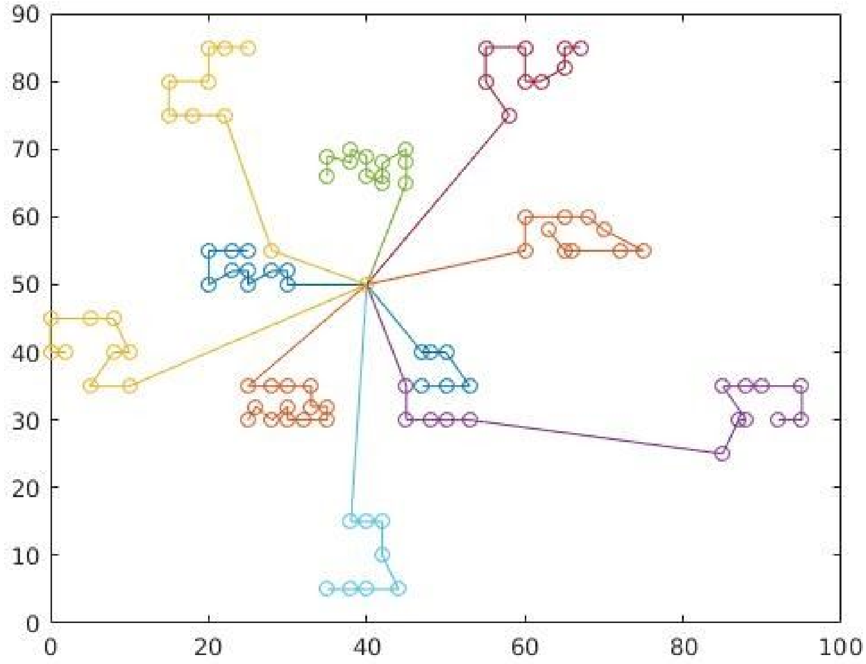
Αριθμός κόμβων: 100
Μέγιστη χωρητικότητα οχημάτων: 200
Μέγιστη απόσταση διαδρομής: -
Χρόνος εξυπηρέτησης πελάτη: -

Διαδρομές:

1	21	22	23	25	26	28	30	31	29	27				
1	53	50	48	44	43	42	41	45	46	47	49	51	52	
1	32	36	34	33	35	37	40	39	38					
1	70	69	65	62	73	81	80	78	82	79	77	72	71	74
1	76	2	3	5	4	6	8	7	10	9	12	11		
1	60	58	56	55	54	57	59	61						
1	99	100	101	98	97	96	95	94	93					
1	68	66	64	75	63	67								

1	91	92	89	86	85	83	84	87	88	90
1	24	14	18	19	20	16	17	15	13	

Αύξων αριθμός διαδρομής	**Ζήτηση**	**Κόστος μετάβασης**	**Κόστος**
1	160	35.2111	35.2111
2	160	51.1061	51.1061
3	200	66.7571	66.7571
4	200	91.4157	91.4157
5	180	43.0513	43.0513
6	200	58.4422	58.4422
7	190	62.2423	62.2423
8	150	27.0375	27.0375
9	170	55.8804	55.8804
10	200	60.8806	60.8806
		Συνολικό κόστος	**552.0246**



Σχήμα 15: Γραφική αναπαράσταση λύσεων για το πρόβλημα vrpnc12

5.3.7 Το πρόβλημα vrpnc14

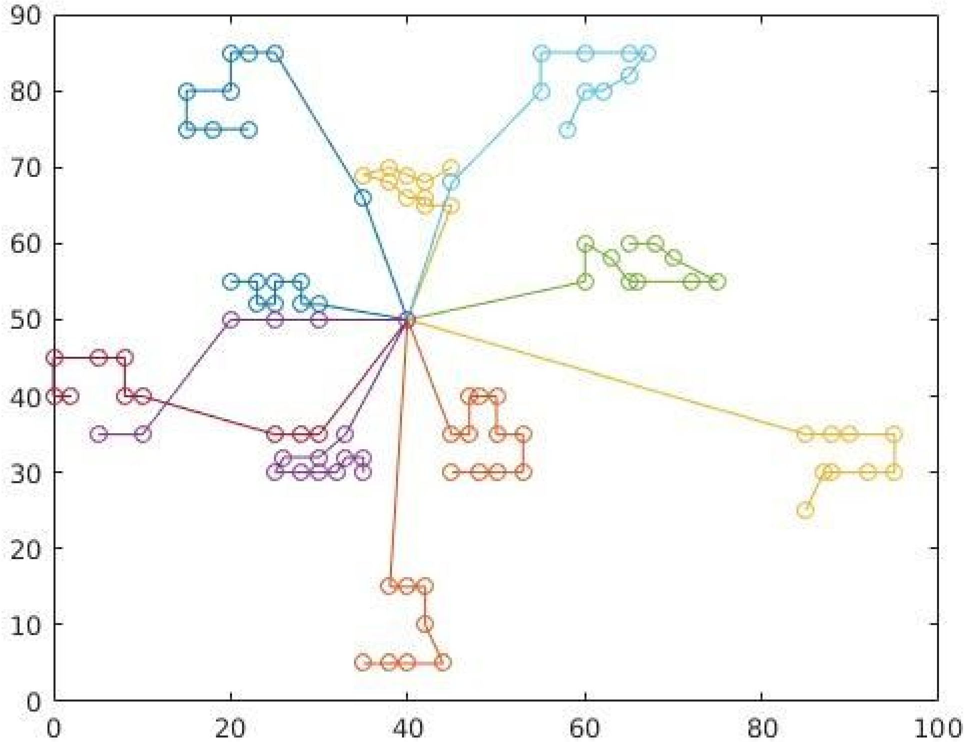
Αριθμός κόμβων: 100
Μέγιστη χωρητικότητα οχημάτων: 200
Μέγιστη απόσταση διαδρομής: 1040
Χρόνος εξυπηρέτησης πελάτη: 90

Διαδρομές:

1	22	23	24	27	26	28	29	31			
1	70	67	68	66	64	63	75	73	62	65	69
1	76	6	4	8	9	12	10	7	5	3	
1	44	47	51	52	49	46	45	43	42	41	
1	91	92	90	88	87	84	83	85	86	89	
1	2	100	101	98	94	93	95	96	97	99	
1	48	50	53	33	34	35	37	40	39	38	
1	11	13	15	17	16	20	19	18	14		
1	60	58	56	55	54	57	59	61			
1	82	79	77	72	71	74	78	80	81		
1	21	25	30	32	36						

Αύξων αριθμός διαδρομής	**Ζήτηση**	**Κόστος μετάβασης**	**Κόστος**
1	140	749.1980	29.1980
2	200	1036.811	46.8113
3	160	939.0420	39.0420
4	130	940.2677	40.2677
5	170	954.4860	54.4860
6	200	965.8983	65.8983
7	200	960.8391	60.8391
8	200	875.2339	65.2339

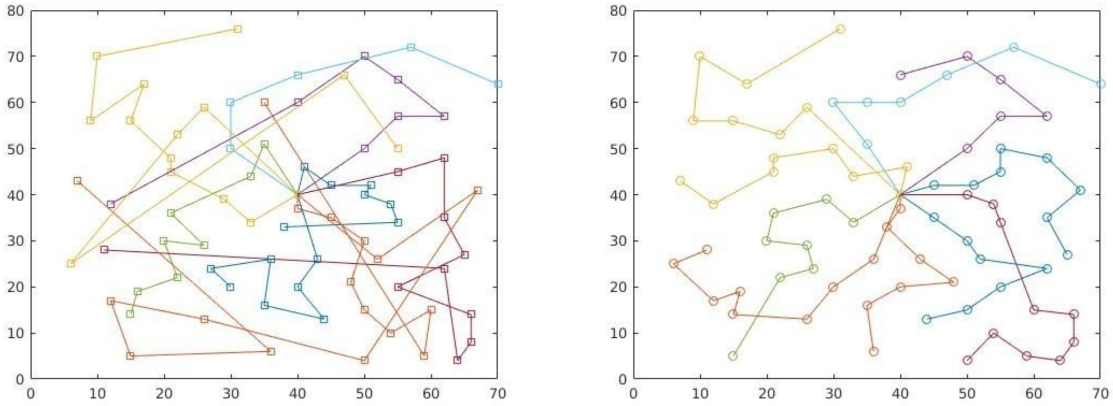
9	200	778.4422	58.4422
10	150	885.8193	75.8193
11	60	493.0277	43.0277
		Συνολικό κόστος	**579.0659**



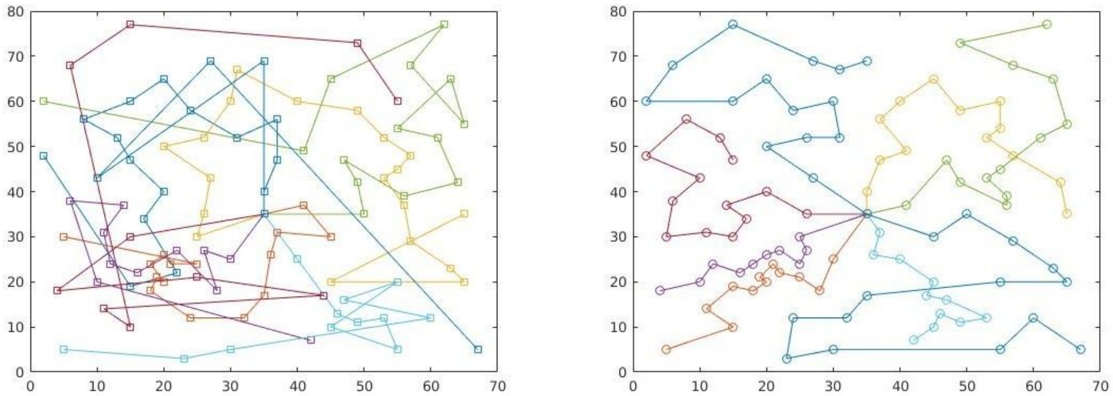
Σχήμα 16: Γραφική αναπαράσταση λύσεων για το πρόβλημα vrpnc14

5.4 Σύγκριση αρχικών-τελικών λύσεων

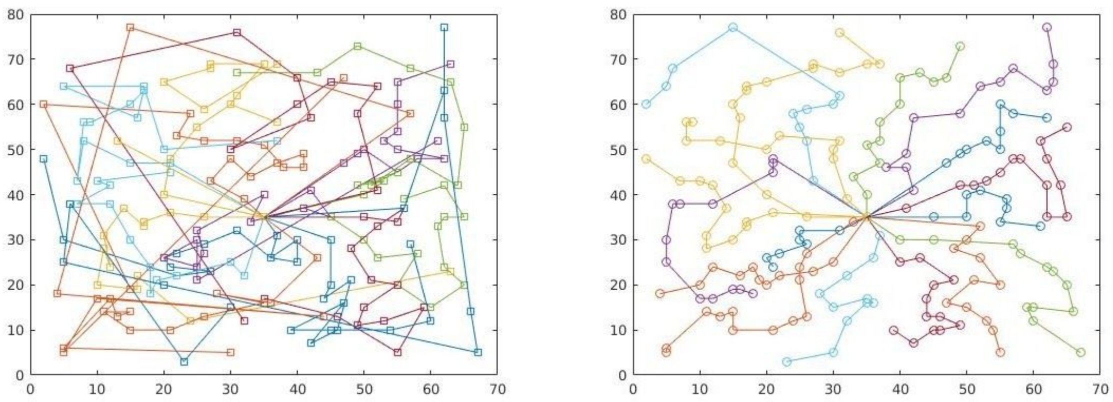
Στη συγκεκριμενη υποενότητα παρουσιάζονται γραφικά οι αρχικές λύσεις για το κάθε πρόβλημα που προκύπτουν από το πρώτο μέλος του αλγορίθμου καθώς και οι τελικές λύσεις που προέκυψαν από τις αρχικές στο δεύτερο μέλος του. Στο αριστερό μέρος εμφανίζονται οι αρχικές ενώ στο δεξί οι τελικές.



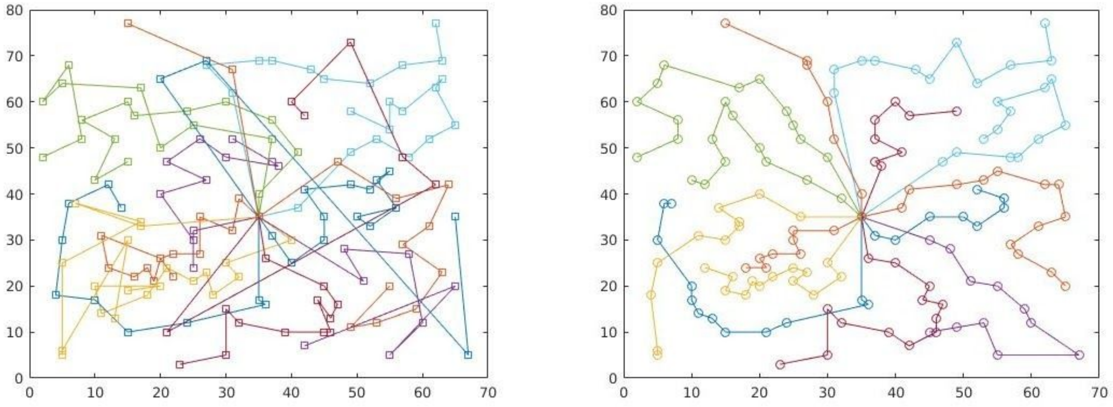
Σχήμα 17: Αρχικές και τελικές λύσεις του προβλήματος vrpnc2



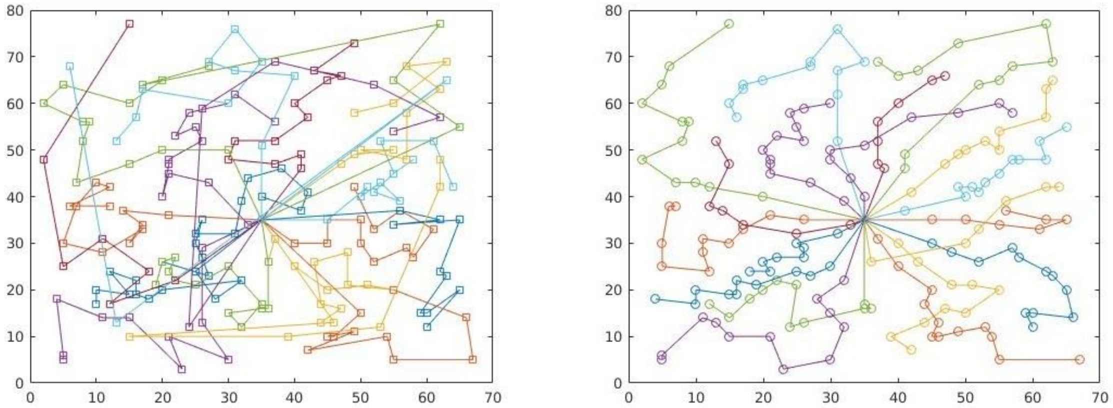
Σχήμα 18: Αρχικές και τελικές λύσεις του προβλήματος vrpnc3



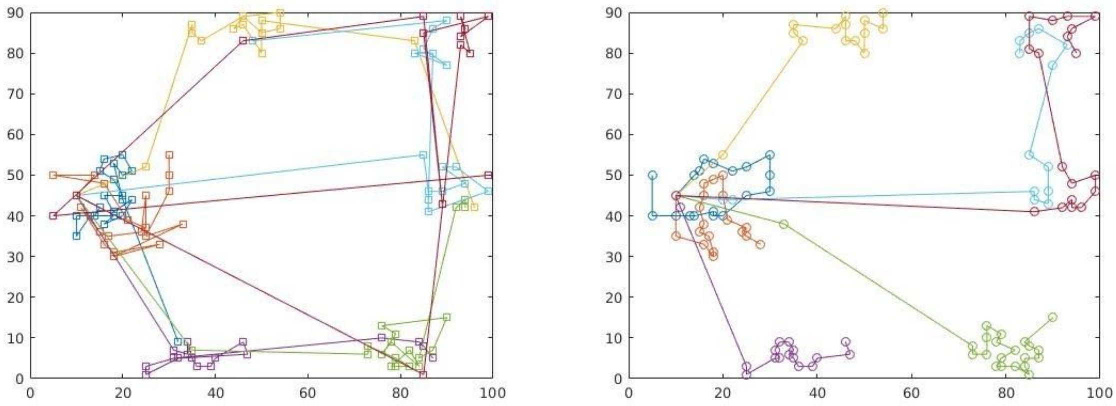
Σχήμα 19: Αρχικές και τελικές λύσεις του προβλήματος vrpnc5



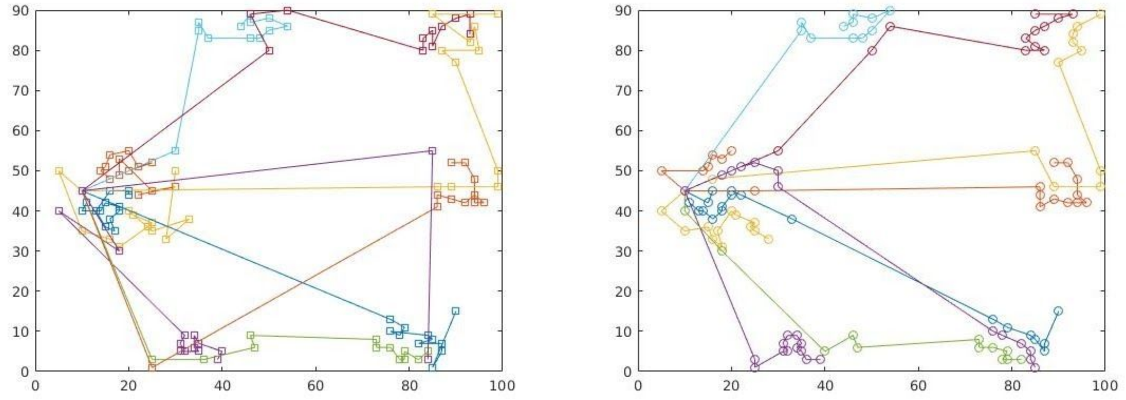
Σχήμα 20: Αρχικές και τελικές λύσεις του προβλήματος vrpnc9



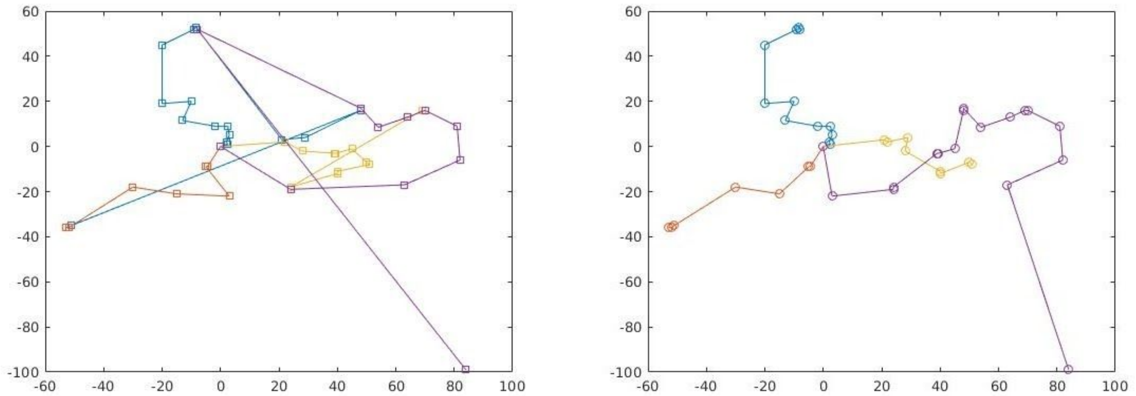
Σχήμα 21: Αρχικές και τελικές λύσεις του προβλήματος vrpnc10



Σχήμα 22: Αρχικές και τελικές λύσεις του προβλήματος vrpnc11



Σχήμα 23: Αρχικές και τελικές λύσεις του προβλήματος vrpnc13



Σχήμα 23: Αρχικές και τελικές λύσεις του προβλήματος F-n45-k4

Κεφάλαιο 6- Συμπεράσματα

Για να φτάσουμε στις λύσεις που παρουσιάστηκαν στο παραπάνω κεφάλαιο έγιναν πολλές δοκιμές με διαφορετική παραμετροποίηση για το κάθε πρόβλημα. Όσων αφορά το πρώτο μέλος του αλγορίθμου, οι λίστες που χρησιμοποιήθηκαν για την παραγωγή των αρχικών λύσεων είχαν από μία έως τέσσερεις θέσεις για πιθανούς κόμβους ανάλογα με το πρόβλημα. Στη διαδικασία αποδοχής κατωφλίου μετά από μεγάλο αριθμό δοκιμών για το άνω φράγμα, το κάτω φράγμα και την αύξηση της τιμής του κατωφλίου καταλήξαμε στο ότι οι βέλτιστες λύσεις παράγονται με την ίδια παραμετροποίηση για όλα τα προβλήματα ανεξάρτητα από τον αριθμό των κόμβων ή κάποια άλλη παράμετρο του προβλήματος. Με τη συγκεκριμένη παραμετροποίηση στη διαδικασία αποδοχής κατωφλίου που εφαρμόζεται στο δεύτερο μέλος του αλγορίθμου σε συνδυασμό με τις τοπικές αναζητήσεις, επιτυγχάνεται μέσω επαναληπτικής διαδικασίας η εύρεση κάποιου τοπικού ελαχίστου καθώς και η εύρεση κάποιου επομένου με την παραγωγή καινούριων λύσεων των οποίων το συνολικό κόστος είναι μεγαλύτερο από το τοπικό ελάχιστο το οποίο βρέθηκε.

Αναλυτικότερα, οι αρχικές λύσεις που προκύπτουν από το πρώτο μέρος του προτεινόμενου αλγορίθμου απέχουν πολύ από τις βέλτιστες και κάθε φορά που τρέχει ο αλγόριθμος παρουσιάζονται διαφορετικές λύσεις, φαινόμενο το οποίο οφείλεται στην τυχαιότητα που χαρακτηρίζει τον τρόπο δημιουργίας τους. Αυτό επιτρέπει την εύρεση διαφορετικών τοπικών ελαχίστων στο δεύτερο μέρος του αλγορίθμου. Για την εύρεση διαφορετικών τοπικών ελαχίστων εκτός από τη μέθοδο αποδοχής κατωφλίου γίνεται και εφαρμογή των τοπικών αναζητήσεων 1-0 relocate και 1-1 exchange, καθώς οι τοπικές αναζητήσεις opt και swap χρησιμοποιούνται για την εύρεση τοπικού ελαχίστου σε διαδρομές οι οποίες αποτελούνται από συγκεκριμένους κόμβους. Στις αρχικές λύσεις επίσης παρατηρείται ότι στα προβλήματα τα οποία έχουν περιορισμό στο συνολικό κόστος μετάβασης για μία διαδρομή, χρησιμοποιούνται παραπάνω οχήματα από ότι στις τελικές λύσεις. Οι επιπλέον αυτές διαδρομές εξαλείφονται στο δεύτερο μέρος του αλγορίθμου με την εφαρμογή της τοπικής αναζήτησης 1-0 relocate με αποτέλεσμα τη σημαντική μείωση του συνολικού κόστους. Σε προβλήματα στα οποία ο περιορισμός διάρκειας διαδρομής δεν υπάρχει, ο αριθμός των οχημάτων που χρησιμοποιούνται στις αρχικές λύσεις συμπίπτει με τον αριθμό των οχημάτων στις τελικές λύσεις όπως και με τον αριθμό των οχημάτων που χρησιμοποιούνται στις βέλτιστες λύσεις με τις οποίες έγινε και η σύγκριση των αποτελεσμάτων.

Η απόκλιση των λύσεων που προέκυψαν από τον προτεινόμενο αλγόριθμο σε σχέση με τις βέλτιστες λύσεις της βιβλιογραφίας είναι απο 3.82% έως 14.6%.

ΒΙΒΛΙΟΓΡΑΦΙΑ

[1] Ιωάννης Μαρινάκης, Αθανάσιος Μυγδαλάς, Σχεδιασμός και Βελτιστοποίηση της Εφοδιαστικής Αλυσίδας, Εκδόσεις Σοφία, 2008, Σελίδες 536.

[2] Ιωάννης Μαρινάκης, Ειδικά Θέματα Συστημάτων Υποστήριξης Αποφάσεων Ευρετικοί – Μεθευρετικοί Αλγόριθμοι, Σχολή Μηχανικών Παραγωγής και Διοίκησης, Πολυτεχνείο Κρήτης, 2015, Σελίδες 118.

[3] Fisher, M. L., Jaikumar, R., and Wassenhove, L. N. V (1986). A Multiplier Adjustment Method for the Generalized Assigment Problem, Management Science, 32 (9), pages 1095-1103.

[4] Ali Asghar Rahmani Hosseinabadi, Javad Vahidi, Valentina Emilia Balas, Seyed Saeid Mirkamali, OVRP_GELS: solving open vehicle routing problem using the gravitational emulation local search algorithm, The Natural Computing Applications Forum 2016.
https://www.readcube.com/articles/10.1007/s00521-016-2608-x

[5] U Derigs, K Reuter, A simple and efficient tabu search heuristic for solving the open vehicle routing problem, Journal of the Operational Research Society, 2009.
https://www.tandfonline.com/doi/abs/10.1057/jors.2008.107

[6] Alexandros Papaspyrou, Τι είναι τα Logistics?, Engineering, 2015.
https://www.slideshare.net/alexandrospapaspurou/logistics-48011400

[7] Γεώργιος Μαλινδρέτος, Εφοδιαστική Αλυσίδα, Logistics & Εξυπηρέτηση Πελατών, ΣΕΑΒ, 2015, Σελίδες 17.
https://repository.kallipos.gr/bitstream/11419/5392/1/02_chapter01.pdf

[8] Σκευοφύλαξ Παναγιώτης, Εξελικτικοί αλγόριθμοι για το ανοικτό – κλειστό πρόβλημα δρομολόγησης οχημάτων ,Διπλωματική Εργασία, Πολυτεχνείο Κρήτης, Ιούλιος 2019.