ΠΟΛΥΤΕΧΝΕΙΟ ΚΡΗΤΗΣ
ΣΧΟΛΗ ΜΗΧΑΝΙΚΩΝ ΠΑΡΑΓΩΓΗΣ ΚΑΙ ΔΙΟΙΚΗΣΗΣ

ΑΛΓΟΡΙΘΜΟΣ ΜΕΤΑΒΛΗΤΗΣ ΓΕΙΤΟΝΙΑΣ ΑΝΑΖΗΤΗΣΗΣ ΓΙΑ ΤΟ ΠΡΟΒΛΗΜΑ ΔΡΟΜΟΛΟΓΗΣΗΣ ΟΧΗΜΑΤΩΝ ΠΕΡΙΟΡΙΣΜΕΝΗΣ ΑΠΟΣΤΑΣΗΣ

VARIABLE NEIGHBORHOOD SEARCH ALGORITHM FOR THE DISTANCE-CONSTRAINED VEHICLE ROUTING PROBLEM

Αγγελή Βασιλική

Επιβλέπων: Δρ. Ιωάννης Μαρινάκης, Επίκουρος καθηγητής

Χανιά 2015

«Η διπλωματική αυτή εργασία αφιερώνεται

στον πατέρα μου Βαγγέλη και

στον παππού μου πατήρ Μιχαήλ…»

Ευχαριστίες

Θα ήθελα να ευχαριστήσω τον καθηγητή μου κ. Ιωάννη Μαρινάκη για την υποστήριξη και την καθοδήγησή του κατά τη διάρκεια εκπόνησης της διπλωματικής μου εργασίας. Επίσης οφείλω ένα μεγάλο ευχαριστώ στον βοηθό του κ. Ιωάννη Μαρινάκη, Ηρακλή-Δημήτριο Ψύχα, για την πολύτιμη βοήθειά του, όπως και στην κα. Μάγδα Μαρινάκη.

Ευχαριστώ πολύ την οικογένεια μου για την υποστήριξή της σε κάθε μου απόφαση. Ακόμη ευχαριστώ τους φίλους μου που αποτελούν στήριγμα για μένα.

Πίνακας Περιεχομένων

Περίληψη 6
ΚΕΦΑΛΑΙΟ 1 : ΕΙΣΑΓΩΓΗ 7
1.1 Η εφοδιαστική αλυσίδα (Supply Chain) 7
1.2 Εφοδιαστική (Logistics) 8
1.3 Διαχείριση της εφοδιαστικής αλυσίδας 10
ΚΕΦΑΛΑΙΟ 2 : ΔΡΟΜΟΛΟΓΗΣΗ ΟΧΗΜΑΤΩΝ 11
2.1 Το πρόβλημα δρομολόγησης οχημάτων (Vehicle Routing Problem) 11
2.2 Το πρόβλημα δρομολόγησης οχημάτων σε περιορισμένη απόσταση(Distance-Constrained Vehicle Routing Problem) 12
2.3 Αλλα προβλήματα δρομολόγησης οχημάτων 14
ΚΕΦΑΛΑΙΟ 3 : ΜΕΘΕΥΡΕΤΙΚΟΙ ΑΛΓΟΡΙΘΜΟΙ 18
3.1 Εισαγωγή 18
3.2 Αλγόριθμος μεταβλητής γειτονιάς αναζήτησης (Variable Neighborhood Search) 19
3.3 Αλλοι μεθευρετικοί αλγόριθμοι. 20
ΚΕΦΑΛΑΙΟ 4 : ΠΕΡΙΓΡΑΦΗ ΚΑΙ ΕΠΙΛΥΣΗ ΤΟΥ ΠΡΟΒΛΗΜΑΤΟΣ ΔΡΟΜΟΛΟΓΗΣΗΣ ΟΧΗΜΑΤΩΝ ΣΕ ΠΕΡΙΟΡΙΣΜΕΝΗ ΑΠΟΣΤΑΣΗ (DCVRP) 24
4.1 Εισαγωγή 24
4.2 Περιγραφή και μοντελοποίηση του προβλήματος 24
4.3 Εύρεση αρχικής λύσης με τον αλγόριθμο του πλησιέστερου γείτονα 26
4.4 Υλοποίηση του αλγορίθμου μεταβλητής γειτονιάς αναζήτησης 27
4.4.1 Εφαρμογή τοπικής αναζήτησης 1-1 ανταλλαγή (1-1 exchange) 28
4.4.2 Εφαρμογή τοπικής αναζήτησης 1-0 επανατοποθέτηση(1-0 relocate) 29
4.4.3 Εφαρμογή τοπικής αναζήτησης 2-opt 30
4.5 Τερματισμός αλγορίθμου και ψευδοκώδικας 31
ΚΕΦΑΛΑΙΟ 5 : ΠΕΡΙΓΡΑΦΗ ΚΑΙ ΑΝΑΛΥΣΗ ΤΩΝ ΑΠΟΤΕΛΕΣΜΑΤΩΝ 33
5.1 Παράδειγμα par6 με 51 κόμβους 33
5.2 Παράδειγμα par7 με 76 κόμβους 36
5.3 Παράδειγμα par8 με 101 κόμβους 39
5.4 Παράδειγμα par9 με 151 κόμβους 42

5.5 Παράδειγμα par10 με 200 κόμβους 45
5.6 Παράδειγμα par12 με 101 κόμβους 48
5.7 Παράδειγμα par14 με 101 κόμβους 52
5.8 Συμπεράσματα 56
ΒΙΒΛΙΟΓΡΑΦΙΑ 57

Περίληψη

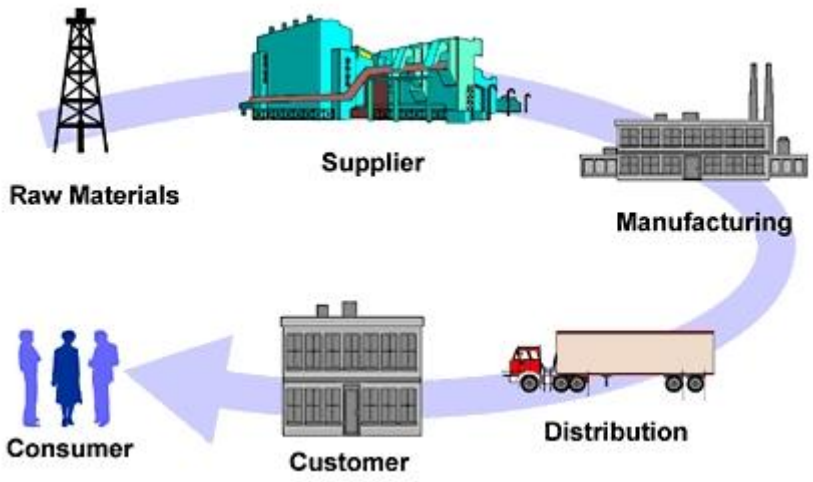
Λόγω της παγκοσμιοποίησης και των διαρκώς αυξανόμενων απαιτήσεων των πελατών, η βελτίωση της Εφοδιαστικής Αλυσίδας αποτελεί καθοριστικό παράγοντα για την ανταγωνιστικότητα ακόμα και για τη βιωσιμότητα της κάθε επιχείρησης. Ενας από τους στόχους της Εφοδιαστικής είναι η μείωση του κόστους των μεταφορών των προϊόντων και των αποθεμάτων. Η συγκεκριμένη διπλωματική εργασία έχει σαν αντικείμενο το πρόβλημα δρομολόγησης οχημάτων σε περιορισμένη απόσταση (Distance Constrained Vehicle Routing Problem). Ξεκινώντας από μία αποθήκη τα οχήματα μεταφέρουν προϊόντα στους πελάτες πηγαίνοντας κάθε φορά στον πλησιέστερο, ώστε να μην υπάρχει χαμένος χρόνος. Για την περιγραφή του προβλήματος ορίζονται κατάλληλα οι απαραίτητοι περιορισμοί για τον αριθμό και τις τοποθεσίες των πελατών, τον συνολικό χρόνο που έχει τη δυνατότητα να δαπανήσει το φορτηγό στο δρόμο, καθώς και τον χρόνο εξυπηρέτησης του κάθε πελάτη. Αρχικά υλοποιείται ένας απλός αλγόριθμος απληστίας από τον οποίο προκύπτει μία αρχική εφικτή λύση που ικανοποιεί τους περιορισμούς. Στη συνέχεια χρησιμοποιείται ο αλγόριθμος Μεταβλητής Γειτονιάς Αναζήτης (Variable Neighborhood Search Algorithm) με σκοπό τη βελτίωση της λύσης. Γίνεται διερεύνηση της κατάλληλης γειτονιάς αναζήτησης με την εφαρμογή μεθόδων τοπικής αναζήτησης (1-1 exchange, 1-0 relocate, 2-opt). Στην εργασία παρουσιάζεται η υλοποίηση του VNS αλγορίθμου, καθώς και τα αποτελέσματα από τη χρήση των αλγορίθμων. Για την ανάπτυξη του αλγορίθμου χρησιμοποιήθηκε το προγραμματιστικό περιβάλλον της MATLAB.

ΚΕΦΑΛΑΙΟ 1 : ΕΙΣΑΓΩΓΗ

1.1 Η εφοδιαστική αλυσίδα (Supply Chain)

Παρατηρείται ότι στις μέρες μας υπάρχει αυξανόμενη οικονομική αλληλεξάρτηση των χωρών παγκόσμια, μέσω του αυξανόμενου όγκου και ποικιλίας διεθνών συναλλαγών αγαθών και υπηρεσιών, της ελεύθερης ροής κεφαλαίου διεθνώς, και της γρήγορης και ευρείας διάχυσης της τεχνολογίας. Επιπρόσθετα παρατηρείται αύξηση των απαιτήσεων των καταναλωτών. Προκύπτει, έτσι, επιτακτική η ανάγκη της ανάπτυξης μιάς πλήρους και ανταποκρινόμενης Εφοδιαστικής Αλυσίδας ώστε να καλύπτει τις απαιτήσεις των πελατών και να εξασφαλίζει αύξηση τόσο των μεριδίων της αγοράς, όσο και της κερδοφορίας της εκάστοτε επιχείρησης.

Με τον όρο Εφοδιαστική Αλυσίδα (Supply Chain) εννοούμε την ροή υλικών, πληροφοριών και υπηρεσιών από τους προμηθευτές πρώτων υλών μέσα από τα εργοστάσια και τις αποθήκες, στους τελικούς πελάτες. Δηλαδή, η Εφοδιαστική Αλυσίδα αποτελείται από τους κατασκευαστές, τους προμηθευτές υλικών, τους χώρους αποθήκευσης, τα κέντρα διανομών, τους μεταφορείς, τους πωλητές, τους πελάτες, τις πρώτες ύλες αλλά και τα έτοιμα προϊόντα που μπορεί να υπάρχουν ανάμεσα.



Σχήμα 1.1 : Εφοδιαστική Αλυσίδα

Παρατηρούμε πως δεν είναι στατική η Εφοδιαστική Αλυσίδα, αλλά είναι ένα δυναμικό σύστημα που εξελίσσεται μέσα στο χρόνο.

Βασική επιδίωξή της είναι η ικανοποίηση του πελάτη. Όλα τα στάδια και οι ροές προϊόντων ή πληροφοριών που περιλαμβάνει η Εφοδιαστική Αλυσίδα δημιουργούν δαπάνες, οι οποίες προσαυξάνονται στο τελικό προϊόν. Μοναδική πηγή εσόδων είναι ο πελάτης, γι'αυτό έχει καθοριστική σημασία για την εκάστοτε επιχείρηση να μείνει ικανοποιημένος από το τελικό προϊόν που θα φτάσει στα χέρια του. Ο βασικός σκοπός της επιχείρησης, λοιπόν, είναι η βέλτιστη διαχείριση της Εφοδιαστικής Αλυσίδας για να παραλάβει το τελικό προϊόν ταχύτερα ο πελάτης, στην καλύτερη τιμή έναντι των ανταγωνιστών, παρέχοντας του την καλύτερη εξυπηρέτηση και με εμπεριεχόμενες τις νέες τεχνολογίες.

1.2 Εφοδιαστική (Logistics)

Η διαδικασία της ροής των αγαθών από το σημείο παραγωγής στο σημείο κατανάλωσης αλλά και η ενδιάμεση αποθήκευσή τους, είναι ζητήματα τα οποία απασχολούσαν κατά τη διάρκεια της ιστορίας όλες τις ανθρώπινες κοινωνίες. Παλαιότερα, η διαδικασία αυτή απαιτούσε πολύ χρόνο και κόπο, περιλαμβάνοντας και την φθορά πολλών αντικειμένων μετακίνησης. Οι άνθρωποι ήταν υποχρεωμένοι να κατοικούν στις περιοχές στις οποίες παράγονταν τα αγαθά λόγω της έλλειψης ανεπτυγμένου μεταφορικού συστήματος και συστήματος αποθήκευσης. Κατά τον 19° αιώνα, ο οποίος αποτελεί μία πολύ σημαντική περίοδο, αναπτύχθηκαν μεταφορικά συστήματα. Συγκεκριμένα, η δημιουργία σιδηροδρόμων βοήθησε στο να ελαχιστοποιηθεί ο χρόνος μεταφοράς. Λόγω αυτής της ανάπτυξης η παραγωγή μιας περιοχής που ήταν πλεονάζουσα μπορούσε να εξαχθεί σε άλλες περιοχές, ενώ τα αγαθά που δεν ήταν δυνατό να παραχθούν τοπικά πλέον μπορούσαν να εισαχθούν.

Ο κλάδος μεταφορών και logistics αποτελεί σημαντικό παράγοντα για την επιχειρηματικότητα με αξιοσημείωτη αναπτυξιακή δυναμική. Η πλεονεκτική γεωγραφική θέση της χώρας μας αποτελεί τη βασική προϋπόθεση ώστε να διαδραματίσει κομβικό σημείο στην παγκόσμια logistics. Ιδιαίτερα ο τομέας των logistics τα τελευταία χρόνια έχει αποκτήσει έντονα στοιχεία τεχνολογικής διείσδυσης, η αξιοποίηση των οποίων υπόσχεται περιορισμό του λειτουργικού κόστους και καλύτερη εκμετάλλευση των υπαρχόντων αποθηκευτικών χώρων.

Ο χώρος της διοίκησης logistics προσφέρεται, πρώτον για τη βελτίωση της ποιότητας των προσφερόμενων υπηρεσιών προς τον πελάτη και δεύτερον για τη μείωση του συνολικού κόστους του προϊόντος με τη βοήθεια στρατηγικών που στοχεύουν κυρίως να καταστήσουν πιο ορθολογικό το σύστημα της φυσικής διανομής. Ας μη ξεχνάμε ότι στις μέρες μας, η συμμετοχή του κόστους της διανομής

στο συνολικό κόστος των προϊόντων είναι αρκετά υψηλή. Όλα τα παραπάνω μπορούν να προσφέρουν σημαντικά ανταγωνιστικά πλεονεκτήματα στις επιχειρήσεις, αλλά και η συνεχής επινόηση και δημιουργία νέων πρωτότυπων προϊόντων δυσκολεύουν την παρούσα κατάσταση. Επομένως, η διαφοροποίηση στον χώρο της διανομής καθίστα αναγκαία σχετική έρευνα.

Ο όρος logistics αφορά τη συσκευασία και ομαδοποίηση, τη διοίκηση των αποθεμάτων και των αποθηκευτικών χώρων, την επιμέλεια του προϊόντος, τις μεταφορές, τη διαχείριση των πληροφοριών της λειτουργίας της διανομής καθώς και την εξυπηρέτηση του πελάτη. Σκοπός της διοίκησης logistics είναι η διαθεσιμότητα των προϊόντων, στον κατάλληλο χρόνο, στον κατάλληλο τόπο και με την κατάλληλη παροχή υπηρεσιών. Αυτό σημαίνει ότι οι δραστηριότητες της λειτουργίας της υλοποίησης πρέπει να διέπονται έντονα από τον προσανατολισμό στις ανάγκες και τις επιθυμίες των πελατών και καταναλωτών.

Η πολυπλοκότητα των logistics μπορεί να διαμορφωθεί, να αναλυθεί, ορατά, και να βελτιστοποιηθεί από ειδικό λογισμικό προσομοίωσης.

1.3 Διαχείριση της εφοδιαστικής αλυσίδας

Ως Διαχείριση Εφοδιαστικής Αλυσίδας ορίζεται ο σχεδιασμός, η οργάνωση και ο συντονισμός όλων των δραστηριοτήτων της Εφοδιαστικής Αλυσίδας.

Το Council of Logistics Management ορίζει τον όρο διαχείριση της εφοδιαστικής (logistics management) ως τη διαδικασία προγραμματισμού, υλοποίησης και ελέγχου για εφικτή και αποτελεσματική ροή και αποθήκευση αγαθών υπηρεσιών και σχετικών πληροφοριών από την πηγή στο σημείο κατανάλωσης με σκοπό τη συμμόρφωση στις απαιτήσεις του πελάτη.

Η Διαχείριση της Εφοδιαστικής Αλυσίδας αποτελεί ένα σχετικά νέο και πολλά υποσχόμενο τομέα της επιστήμης με μεγάλη επίδραση στην αποτελεσματικότητα των σημερινών επιχειρήσεων και στην ευρύτερη διασφάλιση ποιοτικών διαδικασιών, στο ιδιαίτερα ανταγωνιστικό περιβάλλον της σύγχρονης επιχειρηματικότητας. Η διάδοσή της οφείλεται κυριότερα στα ιδιαίτερα σημαντικά αποτελέσματα που επιφέρει, τόσο προς την κατεύθυνση μείωσης του κόστους των επιχειρήσεων όσο και προς την κατεύθυνση του βέλτιστου συντονισμού των διεργασιών της επιχείρησης που συνδέονται με τους προμηθευτές και τους διανομείς.

ΚΕΦΑΛΑΙΟ 2 : ΔΡΟΜΟΛΟΓΗΣΗ ΟΧΗΜΑΤΩΝ

2.1 Το πρόβλημα δρομολόγησης οχημάτων (Vehicle Routing Problem)

Η διαχείριση στόλου οχημάτων αποτελούσε πάντα πρόβλημα για τις μεταφορικές και τις logistics εταιρείες. Σήμερα όμως που τα καύσιμα έχουν μεγάλο κόστος και οι αστοχίες απαγορεύονται, η ορθή διαχείριση στοχεύει όχι μόνο στην απρόσκοπτη λειτουργία, αλλά και στον περιορισμό του κόστους.

Το πρόβλημα δρομολόγησης οχημάτων εμφανίστηκε για πρώτη φορά από τον George Dantzig και John Ramser το 1959, όπου γράφτηκε πρώτα η αλγοριθμική προσέγγιση και στη συνέχεια εφαρμόστηκε στις παραδόσεις πετρελαίου. Το 1964, οι Clarke και Wright βελτίωσαν την προσέγγιση των Dantzig και Ramser χρησιμοποιώντας μια αποτελεσματική προσέγγιση που ονόμασαν άπληστο αλγόριθμο αποταμιεύσεων.

Η διατύπωση του VRP είναι απλή. Ζητούμε τον καθορισμό του συνόλου βέλτιστων διαδρομών για ένα στόλο οχημάτων, προκειμένου να εξυπηρετηθούν οι πελάτες εντοπιζόμενοι σε καθορισμένους κόμβους ενός δικτύου. Σκοπός είναι η σχεδίαση της διαδρομής, η οποία να επιτυγχάνει την ελαχιστοποίηση του συνολικού κόστους της. Πιο συγκεκριμένα, τα δεδομένα που δίνονται είναι ένα σύνολο κόμβων που αντιπροσωπεύουν πελάτες, έναν κόμβο που αντιπροσωπεύει μία αποθήκη, τα κόστη μεταφοράς από τον ένα κόμβο στον άλλο και μία τιμή για κάθε κόμβο πελάτη που αντιπροσωπεύει τη ζήτηση σε προϊόντα του κάθε πελάτη. Το πρόβλημα είναι να βρεθούν οι διαδρομές που ελαχιστοποιούν το κόστος (άθροισμα των κοστών μεταφοράς) και να καλύπτουν τη ζήτηση των πελατών, περνώντας από κάθε πελάτη ακριβώς μία φορά. Μπορεί να θεωρηθεί ότι είναι γενίκευση του Προβλήματος του πλανόδιου πωλητή (Travelling Salesman Problem-TSP), επειδή η διαδρομή που κατασκευάζεται για κάθε επιστροφή του φορτηγού στην αποθήκη (Cycle) αποτελεί στην πράξη ένα TSP πρόβλημα. Η μελέτη ενός τέτοιου προβλήματος είναι σημαντική, αφού απαραίτητο ζήτημα αποτελεί η εξοικονόμηση χρόνου, κυρίως για τη μεταφορά αγαθών και υπηρεσιών. Οι μεταφορείς έχουν να αντιμετωπίσουν από τη μία τις πιέσεις για τη μείωση του κόστους μεταφοράς, από την άλλη την αύξηση του κόστους λόγω διαφόρων παραγόντων, με κυριότερο την αύξηση των τιμών των καυσίμων.

Οι νέες αντιλήψεις για την αποθεματοποίηση, οι συνεχώς αυξανόμενες απαιτήσεις των πελατών και προϊόντα με μικρό κύκλο ζωής, απαιτούν από τους μεταφορείς να προγραμματίζουν τις διαδρομές τους με τρόπο ώστε να επιτυγχάνουν όσο το δυνατόν την πλέον βέλτιστη διαδρομή. Έτσι μπορούν και συμβουλεύονται διάφορα

συστήματα για αυτόν τον σκοπό, όπως: τα Υπολογιστικά Συστήματα Βέλτιστης Δρομολόγησης (Computerized Vehicle Routing System), όπου μέσα από έναν μεγάλο αριθμό ζητούμενων διαδρομών, τις οποίες επεξεργάζονται, βρίσκουν τον πλέον αποδοτικό τρόπο υλοποίησης, υπολογίζοντας τον χρόνο και το καύσιμο που απαιτείται. Τα προβλήματα VRP ανήκουν στη κατηγορία των **NP-hard** προβλημάτων, **μη πολυωνυμικά δύσκολων,** και οι σχετικοί ακριβείς αλγόριθμοι που έχουν αναπτυχθεί μπορούν να επιλύσουν προβλήματα για δίκτυα μέχρι το πολύ 60-70 κόμβων σε αποδεκτό υπολογιστικό χρόνο. Γι' αυτό, για την επίλυση πραγματικών προβλημάτων με σχετικά μεγάλα δίκτυα, έχουν αναπτυχθεί ευρετικοί αλγόριθμοι, οι οποίοι έχουν την ικανότητα να βρίσκουν μία αρκετά ικανοποιητική λύση σε αποδεκτό χρόνο.

2.2 Το πρόβλημα δρομολόγησης οχημάτων σε περιορισμένη απόσταση(Distance-Constrained Vehicle Routing Problem)

Το πρόβλημα δρομολόγησης οχημάτων σε περιορισμένη απόσταση (Distance-Constrained Vehicle Routing Problem - DCVRP) αποτελεί μια παραλλαγή του γενικότερου προβλήματος δρομολόγησης οχημάτων.

Στο συγκεκριμένο πρόβλημα έχουμε ένα σύνολο κόμβων (πελατών) γεωγραφικά διασκορπισμένο σε μια συγκεκριμένη περιοχή. Η κάθε λύση του προβλήματος πρέπει να ικανοποιεί δυο δομικούς περιορισμούς. Αυτοί είναι:

- ✓ η επίσκεψη σε κάθε πελάτη να γίνεται μία μόνο φορά
- ✓ να ικανοποιείται ο περιορισμός της απόστασης

Στόχος του προβλήματος είναι η εύρεση διαδρομών εξυπηρέτησης, οι οποίες να ξεκινάνε και να τελειώνουν στην αποθήκη χωρίς να παραβιάζονται οι παραπάνω περιορισμοί.

Το πρόβλημα δρομολόγησης οχημάτων σε περιορισμένη απόσταση μπορεί να αναπαρασταθεί σε ένα μη προσανατολισμένο γράφημα **G=(V,E)** όπου ένας κόμβος **j∈V** αντιστοιχεί σε έναν πελάτη και ένα τόξο **e∈E** εκφράζει μια διαδρομή μεταξύ ενός ζεύγους πελατών. Ως **n** χαρακτηρίζεται ο αριθμός των πελατών. Η κεντρική αποθήκη συμβολίζεται με τον κόμβο 0. Κάθε όχημα του στόλου μπορεί να διανύσει συγκεκριμένη απόσταση εντός των ορίων $\mathbf{[d^k_{min}, d^k_{max}]}$.

Παράμετροι

n: αριθμός πελατών
m: αριθμός οχημάτων

V: το σύνολο των κορυφών, V = {0,…..,n} όπου το 0 είναι η αποθήκη
E: το σύνολο των τόξων E = {(i,j):i, j ∈ V}
c_{ij}^k: το κόστος της διανυθείσας απόστασης από το i στο j που εκτελείται από το όχημα k
d_{ij}: η απόσταση από τον πελάτη i στον j
$[d_{min}^k, d_{max}^k]$:Το εύρος της συνολικής απόστασης που το εκάστοτε όχημα k μπορεί να εκτελέσει

Μεταβλητές

$$\boldsymbol{x_{ij}^k} = \begin{cases} 1\ \alpha\nu\ \tau o\ \tau\acute{o}\xi o\ (i,j) \varepsilon\kappa\tau\varepsilon\lambda\varepsilon\acute{\imath}\tau\alpha\iota\ \alpha\pi\acute{o}\ \tau o\ \acute{o}\chi\eta\mu\alpha\ k \\ 0\ \sigma\varepsilon\ \kappa\acute{\alpha}\theta\varepsilon\ \acute{\alpha}\lambda\lambda\eta\ \pi\varepsilon\rho\acute{\imath}\pi\tau\omega\sigma\eta \end{cases}$$

$$\boldsymbol{y^k} = \begin{cases} 1\ \alpha\nu\ \tau o\ k\ \chi\rho\eta\sigma\iota\mu o\pi o\iota\varepsilon\acute{\imath}\tau\alpha\iota \\ 0\ \sigma\varepsilon\ \kappa\acute{\alpha}\theta\varepsilon\ \acute{\alpha}\lambda\lambda\eta\ \pi\varepsilon\rho\acute{\imath}\pi\tau\omega\sigma\eta \end{cases}$$

Η **αντικειμενική συνάρτηση** γίνεται:

MinF(x) =$\sum_{k=1}^{m}\sum_{i=0}^{n}\sum_{j=0}^{n} x_{ij}^k * c_{ij}^k$ (1)

υπό :

$\sum_{j=1}^{n} x_{0j}^k$=1 k=1,……,n (2)

$\sum_{i=1}^{n} x_{i0}^k$=1 k=1,……,n (3)

$\sum_{i=0}^{n}\sum_{k=1}^{m}\sum_{i=1\neq j,i=0}^{n} x_{ij}^k$ =1 j=1,…..,m (4)

$\sum_{i=0}^{n} x_{ij}^k$ -$\sum_{i=0}^{n} x_{ij}^k$=0 j=1,…..,m, k=1,…..n (5)

$$\sum_{i\in S}\sum_{j\in S} x_{ij}^{k} \leq |S| - 1 \text{ k=1,....,n } \forall S \subseteq V, |S| \in 2, \ldots., m \qquad (6)$$

$$d_{min}^{k}\, y^{k} \leq \sum_{i=0}^{n}\sum_{j\neq i, j=0}^{n} d_{ij} x_{ij}^{k} \leq d_{max}^{k} y^{k}, \text{k=1,....,n} \qquad (7)$$

$$x_{ij}^{k}\, y^{k} \in \{0,1\}, \text{i=0,.....,m, j=1,.....,m, k=1,.....,n}$$

Η αντικειμενική συνάρτηση (1) εκφράζει την ελαχιστοποίηση του κόστους των συνολικών διαδρομών.

Οι περιορισμοί (2) και (3) εξασφαλίζουν ότι κάθε διαδρομή θα ξεκινήσει και θα τελειώσει στην αποθήκη.

Ο (4) περιορισμός εγγυάται ότι κάθε πελάτης εξυπηρετείται από ένα μόνο όχημα.

Ο (5) αποτελεί την εξίσωση διατήρησης της ροής και διασφαλίζει τη συνέχεια της διαδρομής του κάθε οχήματος.

Η εκάστοτε διαδρομή που εκτελείται και άρα δεν μπορεί να επαναληφθεί ξανά, δηλαδή διαγράφεται, καλύπτεται από τον (6) περιορισμό.

Τέλος, η επιτρεπόμενη διανυθείσα απόσταση εκφράζεται από τον (7) περιορισμό.

[5]

2.3 Αλλα προβλήματα δρομολόγησης οχημάτων

Υπάρχουν αρκετές παραλλαγές του βασικού προβλήματος VRP. Εξαρτάται κάθε φορά από τους περιορισμούς που λαμβάνουμε υπόψιν μας, ώστε να φτάσουμε στη βέλτιστη λύση.

Κάποια άλλα προβλήματα δρομολόγησης οχημάτων παρουσιάζονται συνοπτικά παρακάτω:

- *Πρόβλημα δρομολόγησης οχημάτων με διανομή και παραλαβή προϊόντων κατά τη διάρκεια της διαδρομής. (**Vehicle Routing Problem with Pickup & Delivery - VRPD**).*

Μία ποσότητα φορτίου θα πρέπει να μετακινηθεί από ορισμένες θέσεις (pickup) σε άλλες τοποθεσίες παράδοσης. Ο στόχος είναι να βρεθούν βέλτιστες διαδρομές για το στόλο των οχημάτων, ώστε να επισκεφθούν τις τοποθεσίες παραλαβής και αποβίβασης. Ο κάθε πελάτης που παραλαμβάνει προϊόντα, μπορεί και να παραδώσει στο φορτηγό εμπορεύματα. Σημαντικός περιορισμός σε αυτή την περίπτωση είναι ότι η διανομή γίνεται πριν την παραλαβή.

- *Το περιορισμένης χωρητικότητας πρόβλημα δρομολόγησης οχημάτων (**Capacitated Vehicle Routing Problem - CVRP**)*

  Αποτελεί, και αυτό το πρόβλημα, επέκταση του **TSP.** Εδώ, όμως, δεν γίνονται οι διανομές σε όλους τους πελάτες σε μία μόνο διαδρομή. Ο λόγος που συμβαίνει αυτό είναι η περιορισμένη χωρητικότητα που έχει το όχημα. Ετσι πραγματοποιούνται πολλαπλές κυκλικές διαδρομές, όλες με αρχή και τέλος το ίδιο σημείο, την αποθήκη. Επίσης κάθε πελάτης επισκέπτεται από έναν μόνο κύκλο. Στόχος και εδώ είναι η ελαχιστοποίηση του κόστους. Βασικός περιορισμός είναι: το άθροισμα της ζήτησης των κόμβων που επισκέπτονται από έναν κύκλο δεν ξεπερνάει τη χωρητικότητα του οχήματος.

- *Δρομολόγηση οχημάτων για την εξυπηρέτηση πελατών μέσα σε δεδομένα χρονικά περιθώρια - χρονικά παράθυρα (**Vehicle Routing Problem with Time Windows - VRPTW**).*

  Αποτελεί επέκταση του **CVRP.** Ισχύουν και εδώ οι περιορισμοί χωρητικότητας. Επιπλέον η εξυπηρέτηση του κάθε πελάτη πρέπει να γίνει εντός ενός ορισμένου χρονικού διαστήματος, το οποίο ονομάζεται και χρονικό παράθυρο. Πριν ή μετά το χρονικό παράθυρο δεν μπορεί να γίνει η εξυπηρέτηση. Θα πρέπει να γίνει από ένα μόνο φορτηγό, μία μόνο φορά και θα πρέπει να παραλάβει όλο το φορτίο. Σκοπός είναι να βρεθεί ένα σύνολο από διαδρομές, όπου κάθε διαδρομή θα ξεκινά και θα τελειώνει στην αποθήκη χωρίς να παραβιάζονται οι περιορισμοί της χωρητικότητας, οι περιορισμοί από τα χρονικά παράθυρα και να ελαχιστοποιείται το συνολικό μήκος διαδρομών.

- *Η ύπαρξη πολλαπλών αποθηκών (**Multi-Depot Vehicle Routing Problem - MDVR**)*

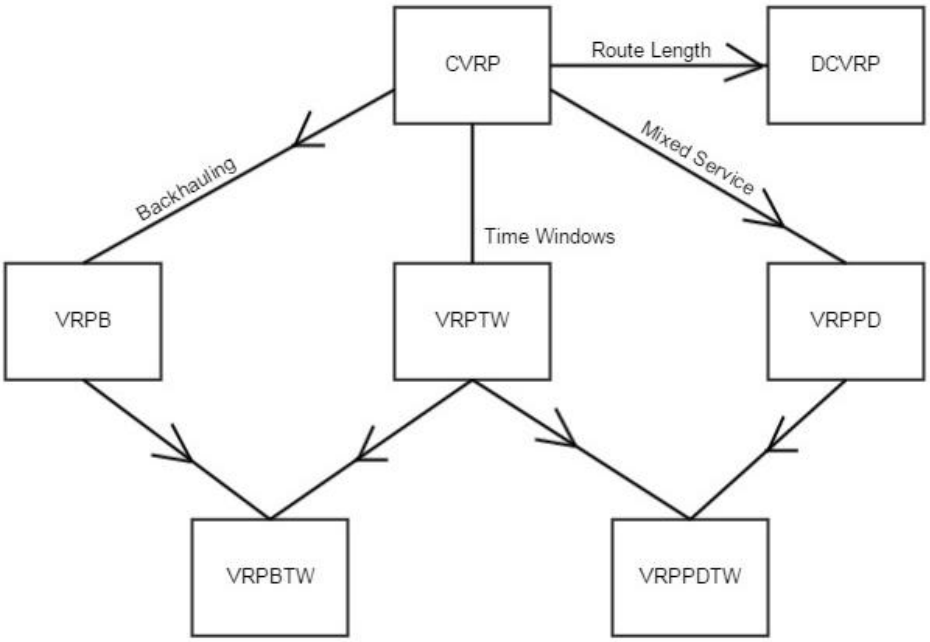
  Όπως γίνεται φανερό από τον ορισμό του προβλήματος, η εξυπηρέτηση των πελατών γίνεται από περισσότερες από μια αποθήκες. Αυτό το πρόβλημα μπορεί να επιλυθεί με δυο τρόπους:

  - είτε η κάθε αποθήκη να έχει τον δικό της αριθμό οχημάτων και πελατών όπου έτσι ουσιαστικά δημιουργούνται πολλαπλά VRP,
  - είτε το κάθε όχημα να ξεκινά από κάποια αποθήκη, να τερματίζει σε μια άλλη, ή ακόμα και να σταματά για ανεφοδιασμό σε μια τρίτη αποθήκη.

  Το πρόβλημα αυτό μπορεί να θεωρηθεί σαν πρόβλημα ομαδοποίησης αφού στόχος του είναι να βρεθεί ο αριθμός των διαδρομών των οχημάτων που ανήκουν σε κάθε μία αποθήκη. Πρώτα αναθέτουμε τους πελάτες στις αποθήκες και μετά δημιουργούμε δρομολόγια για κάθε ένα πελάτη και για κάθε ένα όχημα.

- *Πρόβλημα δρομολόγησης οχημάτων με δύο είδη πελατών κατά τη διάρκεια της διαδρομής (**Vehicle Routing Problem with Backhauls and linehauls customers - VRPB**)*

  Σε αυτό το πρόβλημα ο πελάτες χωρίζονται σε δύο υποσύνολα. Στο πρώτο ανήκουν οι πελάτες που απαιτούν διανομή εμπορευμάτων (linehauls customers) και στο δεύτερο οι πελάτες που απαιτούν εμπορεύματα να περισυλλεχθούν από αυτούς (backhauls customers).
  Ισχύουν και εδώ οι περιορισμοί αποστάσεων, χωρητικότητας καθώς και η μοναδική επίσκεψη του κάθε πελάτη. Επιπρόσθετα <u>οι πελάτες του δεύτερου υποσυνόλου επισκέπτονται **μετά** από τους πελάτες του πρώτου. Οι διαδρομές που περιλαμβάνουν μόνο πελάτες του δεύτερου τύπου (backhauls customers*)* δεν επιτρέπονται.</u>



Σχήμα 2.1 :Σχέση μεταξύ των παραλλαγών του κοινού προβλήματος VRP. (Πηγή: https://en.wikipedia.org/wiki/Vehicle_routing_problem)

- *Πρόβλημα δρομολόγησης οχημάτων χωρίς επιστροφή στην αποθήκη (**Open Vehicle Routing Problem - OVRP**)*

  Σε αυτή την κατηγορία προβλημάτων το φορτηγό δεν επιστρέφει στην αποθήκη μετά από ορισμένο χρονικό περιθώριο. Συνεχίζει και εκτελεί δρομολόγια μέχρι να αδειάσει το φορτίο του. Κάθε διαδρομή αποτελεί ένα μονοπάτι (όχι κύκλο), το οποίο περνάει από κάθε κόμβο που περιέχει μόνο μία φορά. Με αυτόν τον τρόπο γίνεται εξοικονόμηση οχημάτων. Είναι χρήσιμο, συνήθως, για επιχειρήσεις που δεν διαθέτουν δικό τους στόλο οχημάτων, ή δεν διαθέτουν τους κατάλληλους πόρους για να ικανοποιήσουν τους πελάτες τους. Μισθώνουν, λοιπόν, στόλο ή παραχωρούν μέρος της διανομής σε εξωτερικούς παράγοντες, οι οποίοι αναλαμβάνουν και το κόστος ανά όχημα. Στόχος είναι ο μικρότερος αριθμός οχημάτων για την κάλυψη των πελατών.

ΚΕΦΑΛΑΙΟ 3 : ΜΕΘΕΥΡΕΤΙΚΟΙ ΑΛΓΟΡΙΘΜΟΙ

3.1 Εισαγωγή

Με την εφαρμογή της τοπικής αναζήτησης μπορεί να εγκλωβιστούμε σε κάποιο τοπικό ελάχιστο. Οι μεθευρετικοί αλγόριθμοι έρχονται να δώσουν λύση, καθώς διερευνούν λύσεις σε μη εφικτές περιοχές ή ακόμη και σε περιοχές που χειροτερεύουν την τιμή της αντικειμενικής συνάρτησης. Σκοπός είναι η αποφυγή του εγκλωβισμού του αλγορίθμου σε τοπικά ελάχιστα, επιτυγχάνοντας λύσεις οι οποίες είναι πλησιέστερες στις βέλτιστες. Η ποιότητα των λύσεων που παράγουν οι μεθευρετικές μέθοδοι είναι αρκετά πιο υψηλή συγκριτικά με τις ευρετικές μεθόδους. Οι μεθευρετικοί αλγόριθμοι που επικεντρώνονται στην αναζήτηση γύρω από κάποιο σημείο που έχει βρεθεί το τοπικό ελάχιστο με τη μέθοδο της τοπικής αναζήτησης μπορούν να διακριθούν σε τέσσερις βασικές κατηγορίες, ανάλογα με τον τρόπο που χρησιμοποιούν για να αποφύγουν το τοπικό ελάχιστο:

1. Επαναληπτικές διαδικασίες που αρχίζουν από διαφορετικές αρχικές λύσεις.
   - ✓ Η μέθοδος της διαδικασίας άπληστης τυχαιοποιημένης προσαρμοστικής αναζήτησης (**Greedy Randomized Adaptive Search Procedure - GRASP)**
   - ✓ Αλγόριθμοι πολυεναρκτήριας τοπικής αναζήτησης (**multistart local search**)
   - ✓ Αλγόριθμοι της επαναληπτικής τοπικής αναζήτησης (**iterated local search)**
2. Αλγόριθμοι που δέχονται, υπό κάποιες συνθήκες, γειτονικές κινήσεις που δεν βελτιώνουν τη λύση. Με αυτόν τον τρόπο μπορεί στις επόμενες κινήσεις να βρούμε ένα τοπικό ελάχιστο καλύτερο από το τρέχων.
   - ✓ Προσομοιωμένη ανόπτηση (**simulated annealing**)
   - ✓ Περιορισμένη αναζήτηση (**tabu search**)
3. Αλγόριθμοι που αλλάζουν τη γειτονιά αναζήτησης. Οταν βρεθούν σε τοπικό ελάχιστο αλλάζουν τον αλγόριθμο που χρησιμοποιούν.
   - ✓ αλγόριθμος μεταβλητής γειτονιάς αναζήτησης (**Variable Neighborhood Search - VNS**)
   - ✓ αλγόριθμος επέκτασης της γειτονιάς αναζήτησης (**Expanding Neighborhood Search - ENS**)
4. Αλγόριθμοι που αλλάζουν την αντικειμενική συνάρτηση ή τους περιορισμούς του προβλήματος
   - ✓ αλγόριθμος καθοδηγούμενης τοπικής αναζήτησης (**Guided Local Search**).

Εφαρμόζονται, βέβαια, και υβρίδια των παραπάνω μεθευρετικών μεθόδων.

3.2 Αλγόριθμος μεταβλητής γειτονιάς αναζήτησης (Variable Neighborhood Search)

Στην παρούσα διπλωματική δουλέψαμε με τον μεθευρετικό αλγόριθμο μεταβλητής γειτονιάς αναζήτησης (**VNS**) , ώστε να βρούμε καλύτερη λύση από την αρχική λύση που προκύπτει από την εφαρμογή της μεθόδου του πλησιέστερου γείτονα.

Ο *αλγόριθμος μεταβλητής γειτονιάς αναζήτησης* προτάθηκε από τους Hansen και Mladenovic το 1997. Η βασική ιδέα είναι η χρήση πολλών μεθόδων τοπικής αναζήτησης για να βρεθεί μια καλύτερη λύση ή για να ξεφύγει ο αλγόριθμος από κάποιο τοπικό βέλτιστο. Σχεδιάστηκε για την προσέγγιση των λύσεων διακριτών και συνεχών προβλημάτων βελτιστοποίησης και σύμφωνα με αυτό στοχεύει στη λύση προβλημάτων γραμμικού προγραμματισμού, ακέραιου, μικτών προβλημάτων ακέραιου προγραμματισμού, μη γραμμικών προβλημάτων κ.λ.π..

Η μέθοδος αυτή εκμεταλλεύεται το γεγονός ότι διαφορετικές μέθοδοι τοπικής αναζήτησης μπορούν να οδηγήσουν σε διαφορετικά τοπικά βέλτιστα. Είναι ένας στοχαστικός αλγόριθμος, ο οποίος επιλέγει ένα σύνολο γειτονιών (Nl, όπου l=1,…,lmax) και στη συνέχεια η κάθε επανάληψη αλγορίθμου ακολουθεί τα εξής βήματα: την **ανακίνηση** (shaking) , την **τοπική αναζήτηση** (local search) και την **κίνηση** (move) . Στην ουσία σε κάθε επανάληψη μία αρχική λύση s δημιουργείται από την τρέχουσα γειτονιά αναζήτησης (για παράδειγμα με τυχαίο τρόπο). Μία διαδικασία τοπικής αναζήτησης εφαρμόζεται στη λύση s με στόχο να παράγουμε τη λύση s. ΄Επειτα ελέγχεται η αντικειμενική συνάρτηση και αν η λύση οδηγεί σε βελτίωση του κόστους τότε η καινούρια λύση αντικαθιστά την αρχική λύση. Στη συνέχεια η ίδια διαδικασία αρχίζει από την αρχή με τη χρήση της γειτονιάς N1 και τη καινούρια λύση s που βρήκαμε προηγουμένως. Αν δεν βρεθεί μία καλύτερη λύση τότε ο αλγόριθμος προχωρά στην επόμενη γειτονιά αναζήτησης. Στη συνέχεια δίνεται ένας ψευδοκώδικας του αλγόριθμου της μεταβλητής γειτονιάς αναζήτησης.

Αλγόριθμος Μεταβλητής Γειτονιάς Αναζήτησης

Αρχικοποίηση

Επέλεξε ένα σύνολο γειτονιών (Nl, l = 1, ⋯ , lmax)

Επέλεξε μια αρχική λύση s_0

l = 1

repeat

Δημιούργησε μία λύση s στη γειτονιά του Nl

s’’= LS(s), Εφάρμοσε μία διαδικασία τοπικής αναζήτησης στο s’

if f(s’’) < f(s’) **then**

s = s''

l = 1

else

l = l + 1

end if

until l ≤ lmax

Επέστρεψε τη βέλτιστη λύση. [10]

Οι εφαρμογές του VNS αυξάνονται με ταχείς ρυθμούς και αφορούν πολλούς τομείς. Κάποιοι από αυτούς είναι: η θεωρία της τοποθεσίας, η ανάλυση συστάδων, ο προγραμματισμός, η δρομολόγηση οχημάτων, ο σχεδιασμός των δικτύων, η τεχνητή νοημοσύνη, η μηχανική, η ομαδοποίηση των προβλημάτων, η βιολογία, η γεωμετρία, ο σχεδιασμός των τηλεπικοινωνιών .

3.3 Αλλοι μεθευρετικοί αλγόριθμοι.

Παρακάτω περιγράφονται κάποιοι από τους πιο σημαντικούς μεθευρετικούς αλγορίθμους:

*Διαδικασία άπληστης τυχαιοποιημένης προσαρμοστικής αναζήτησης (**Greedy Randomized Adaptive Search Procedure – GRASP)***

Η διαδικασία άπληστης τυχαιοποιημένης προσαρμοστικής αναζήτησης δραστηριοποιείται στην εύρεση προσεγγιστικών λύσεων σε προβλήματα συνδυαστικής βελτιστοποίησης. Η τεχνική αυτή παρέχει μια εφικτή λύση σε κάθε επανάληψη. Οι επαναλήψεις της διαδικασίας σταματούν όταν κάποιο κριτήριο τερματισμού ικανοποιείται. Το τελικό αποτέλεσμα είναι η πιο βέλτιστη λύση που βρέθηκε από όλες τις επαναλήψεις. Οι δύο φάσεις της επανάληψης είναι οι εξής: μια φάση κατασκευής μιας αρχικής λύσης (construction phase), και μια διαδικασία τοπικής αναζήτησης (local search phase) για βελτιστοποίηση αυτής της λύσης. Στη φάση κατασκευής, μια τυχαιοποιημένη συνάρτηση απληστίας χρησιμοποιείται για να κατασκευαστεί μια λύση. Αυτή η αρχική λύση στη συνέχεια βελτιώνεται με τη χρήση της διαδικασίας τοπικής αναζήτησης. Η στρατηγική επιλογής του επόμενου στοιχείου βασίζεται στην τυχαία επιλογή από μια λίστα υποψηφίων, που ονομάζεται λίστα περιορισμού των υποψηφίων για εισαγωγή στη λύση την οποία κάθε στοιχείο κατατάσσεται βάσει μιας συνάρτησης απληστίας. Ένα μειονέκτημα

του GRASP είναι η ανεξαρτησία των επαναλήψεων, ο βασικός αλγόριθμος απορρίπτει πληροφορίες για οποιαδήποτε λύση βρέθηκε και δεν είναι η βέλτιστη.

*Αλγόριθμος επαναληπτικής τοπικής αναζήτησης (**Iterated Local Search – ILS**)*

Η μέθοδος αυτή δραστηριοποιείται στη βελτίωση των υπαρχόντων μεθόδων πολυεναρκτήριας τοπικής αναζήτησης, εισάγοντας μια διαταραχή στη λύση που οδηγεί σε τοπικό ελάχιστο. Έτσι η νέα βέλτιστη λύση αποθηκεύεται σε περίπτωση που κινδυνεύσει να χαθεί. Ο αλγόριθμος της επαναληπτικής τοπικής αναζήτησης περιέχει τρία βασικά συστατικά. **1)** Τον αλγόριθμο τοπικής αναζήτησης. Ο αλγόριθμος αυτός έχει την δυνατότητα να χρησιμοποιεί διαφορετικές μεθόδους σε κάθε επανάληψη, ώστε να μην χειροτερέψει ο φόρτος του υπολογισμού. **2)** Την διαδικασία διαταραχής. Η διαδικασία της διαταραχής αναλαμβάνει την κράτηση κάποιων στοιχείων της αρχικής λύσης σε συνδυασμό με την μετάλλαξη των υπολοίπων σημείων. **3)** Το κριτήριο αποδοχής. Το κριτήριο αυτό περιγράφει τις προϋποθέσεις κάτω από τις οποίες το νέο τοπικό ελάχιστο μπορεί να αντικαταστήσει το παλιό τοπικό ελάχιστο. Για να επιλεγεί μία λύση, πρέπει να υπάρχει η ικανότητα εντατικοποίησης του αλγορίθμου γύρω από κάποια σημεία και στη συνέχεια η ικανότητα της εξερεύνησης του αλγορίθμου περισσότερων μερών του χώρου λύσεων.

*Προσομοιωμένη Ανόπτηση (**Simulated Annealing – SA**)*

Στη μεταλλουργία ανόπτηση ονομάζεται η θερμική κατεργασία, στην οποία υπόκεινται τα μέταλλα ή κράματά τους, που έχουν υποστεί νωρίτερα άλλη κατεργασία, ώστε να μειωθούν οι ατέλειές τους. Είναι, δηλαδή, η διαδικασία κατά την οποία ένα στερεό υλικό θερμαίνεται μέχρι το σημείο τήξης του και στη συνέχεια ακολουθεί αργά η ψύξη. Το κατώτερο επίπεδο ενέργειας είναι στο τέλος της ψύξης. Η στρατηγική μείωσης της ψύξης μπορεί να επιφέρει ατέλειες στο σύστημα. Ο αλγόριθμος της προσομοιωμένης ανόπτησης αποτυπώνει στην ενέργεια ενός συστήματος το οποίο ψύχεται, τη σύγκλιση, δηλαδή σε κάποιο σημείο ισορροπίας. Αυτό συμβαίνει όταν ο αλγόριθμος σταματήσει την κίνηση στον εφικτό χώρο αναζήτησης, τότε δηλαδή έχει εντοπιστεί το τελικό σημείο ισορροπίας.

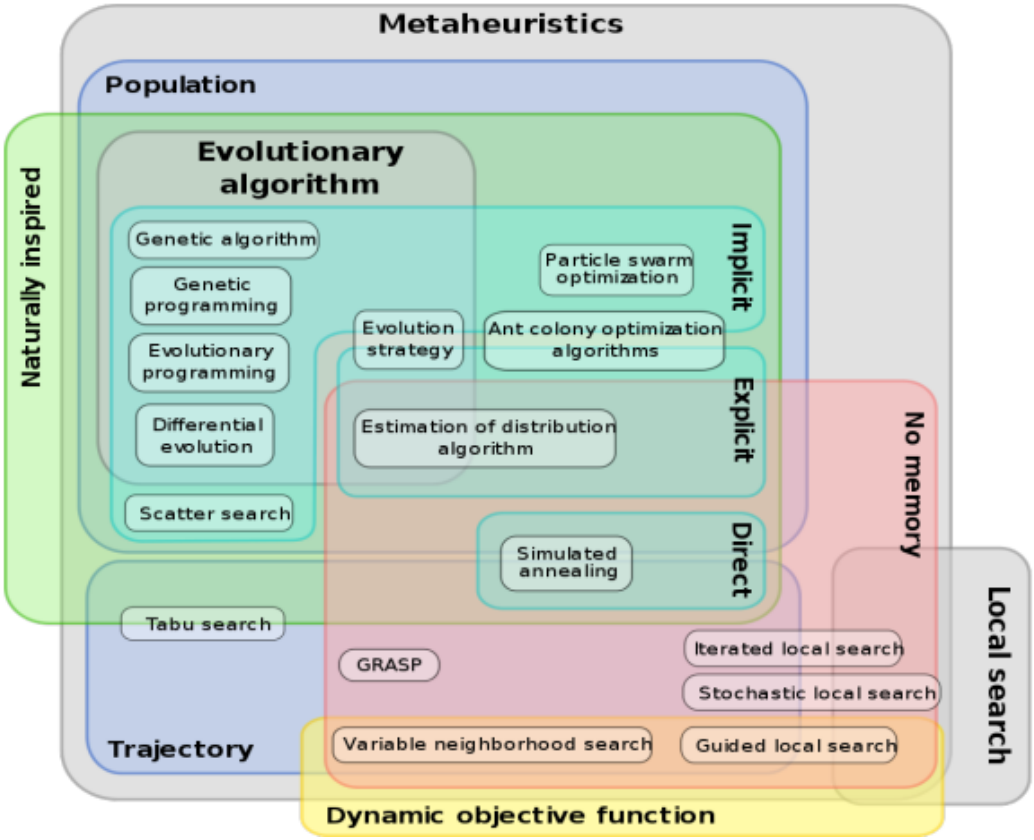
*Περιορισμένη Αναζήτηση (**Tabu Search - TS**)*

Ο μεθευρετικός αυτός αλγόριθμος παρουσιάστηκε από τον Fred Glover το 1986 και επισημοποιήθηκε το 1989. Η μέθοδος αυτή, χρησιμοποιεί έναν ευρετικό αλγόριθμο, για να μετακινηθεί από τη μία λύση στην άλλη, η οποία λύση ενέχει τον κίνδυνο να παγιδευτεί σε τοπικό ελάχιστο. Για την αποφυγή του κινδύνου αυτού, γίνεται χρήση μιας στρατηγικής που επιλέγει την επόμενη λύση και όχι μια τυχαία. Η στρατηγική αυτή χρησιμοποιεί μνήμη. Οι τελευταίες κινήσεις αναγράφονται σε μια λίστα, την

λίστα της περιορισμένης αναζήτησης (tabulist) και οι συγκεκριμένες κινήσεις απαγορεύεται να επιστρέψουν στη λύση για ένα συγκεκριμένο αριθμό επαναλήψεων. Η περιορισμένη αναζήτηση δεν συγκλίνει με φυσικό τρόπο, έτσι είναι αναγκαίο να καθοριστούν κριτήρια τερματισμού.

*Αλγόριθμος επανασύνδεσης διαδρομών (**Path Relinking-PR**)*

Ο αλγόριθμος της επανασύνδεσης διαδρομών δημιουργεί καινούριες λύσεις εξετάζοντας διαφορετικές τροχιές που αποτελούνται από υψηλές ποιοτικά λύσεις. Ο αλγόριθμος ξεκινάει από μία από αυτές τις λύσεις που ονομάζεται εναρκτήρια λύση (**starting solution**) και μια λύση στόχου, η τελική λύση (**target solution**). Σύμφωνα με το Path Relinking χρειάζεται να δημιουργηθούν τροχιές στο χώρο που να συνδέουν την αρχική με την τελική λύση. Πολλές φορές η καλύτερη από τις δύο λύσεις παίζει το ρόλο της αρχικής λύσης και η χειρότερη παίζει το ρόλο της τελικής λύσης, αλλά μπορεί να συμβεί και το αντίστροφο. Η μέθοδος στοχεύει στο τέλος των επαναλήψεων να μην υπάρχει απόσταση μεταξύ των δύο λύσεων.



*Σχήμα 3.1 :*Παραδείγματα μεθευρετικών αλγορίθμων και η συσχέτισή τους. (Πηγή:Wikipedia)

ΚΕΦΑΛΑΙΟ 4 : ΠΕΡΙΓΡΑΦΗ ΚΑΙ ΕΠΙΛΥΣΗ ΤΟΥ ΠΡΟΒΛΗΜΑΤΟΣ ΔΡΟΜΟΛΟΓΗΣΗΣ ΟΧΗΜΑΤΩΝ ΣΕ ΠΕΡΙΟΡΙΣΜΕΝΗ ΑΠΟΣΤΑΣΗ (DCVRP)

4.1 Εισαγωγή

Σε αυτό το Κεφάλαιο παρουσιάζονται αναλυτικά τα δεδομένα που χρησιμοποιήσαμε για το συγκεκριμένο πρόβλημα. Θα γίνουν γνωστά τα επιμέρους χαρακτηριστικά των οχημάτων και των πελατών καθώς και οι δοθέντες περιορισμοί. Ακόμη, περιγράφεται η μοντελοποίηση του προβλήματος και η αναπαράσταση τους στο προγραμματιστικό περιβάλλον της MATLAB. Στη συνέχεια, αναλύονται τα επιμέρους βήματα του αλγορίθμου της Μεταβλητής Γειτονιάς Αναζήτησης (VNS) με αρχικό στάδιο την εύρεση αρχικής λύσης με κάποιον αλγόριθμο απληστίας και τελικό στάδιο την εύρεση βέλτιστης λύσης με εφαρμογή τοπικών αναζητήσεων.

4.2 Περιγραφή και μοντελοποίηση του προβλήματος

Το πρόβλημα που καλούμαστε να επιλύσουμε είναι η δρομολόγηση οχημάτων σε περιορισμένη απόσταση. Στο συγκεκριμένο πρόβλημα διαθέτουμε μία αποθήκη, ένα συγκεκριμένο αριθμό πελατών με συγκεκριμένη ζήτηση ο καθένας και ένα στόλο όμοιων οχημάτων άπειρης χωρητικότητας το καθένα. Στόχος είναι η εξυπηρέτηση όλων των πελατών ικανοποιώντας πάντα τον περιορισμό της απόστασης και ταυτόχρονα η αναζήτηση της βέλτιστης λύσης, ώστε τα οχήματα να διανύσουν την ελάχιστη ευκλείδεια απόσταση. Αρχικά βρίσκουμε μία αρχική υποβέλτιστη λύση χρησιμοποιώντας τον αλγόριθμο του πλησιέστερου γείτονα. Στη συνέχεια καλούμαστε να βελτιστοποιήσουμε αυτή την αρχική λύση χρησιμοποιώντας τον μεθευρετικό αλγόριθμο μεταβλητής γειτονιάς αναζήτησης.

Οι αλγόριθμοι επιλύθηκαν στο πραγραμματιστικό περιβάλλον της Matlab. Πρώτα πρέπει να μελετήσουμε το πρόβλημα ώστε να καταλήξουμε στο ποιες μεταβλητές πρέπει να αρχικοποιήσουμε και ποιες θα κατασκευάσουμε στην πορεία του αλγορίθμου. Ένα από τα δεδομένα που έχουμε είναι η αρίθμηση των πελατών, με τον πελάτη 0 να είναι η αποθήκη. Στο προγραμματιστικό περιβάλλον της Matlab, όμως, η αποθήκη αναπαρίσταται με τον αριθμό 1, ο πελάτης 1 με τον αριθμό 2, κ.ο.κ.. Ένα άλλο δεδομένο που μας δίνεται είναι οι συντεταγμένες χ και y που αντιστοιχούν σε κάθε πελάτη καθώς και ο χρόνος εξυπηρέτησης. Οι συντεταγμένες χ και y αποθηκεύτηκαν σε φύλλα excel και διαβάστηκαν από την Matlab με την εντολή xlsread, καταχωρώντας τες σε ένα δισδιάστατο πίνακα «A». Η εντολή που χρησιμοποιήθηκε ήταν «A = xlsread(filename)», όπου στο filename είχε εκχωρηθεί το όνομα του παραδείγματος που πρόκειται να χρησιμοποιηθεί.

Πιο συγκεκριμένα,

n: αριθμός κόμβων

periorismos: ο περιορισμός της απόστασης ή του χρόνου που λαμβάνεται υπόψη σε κάθε παράδειγμα

epistrofh: η απόσταση ή ο χρόνος επιστροφής στην αποθήκη που ισχύει σε κάθε παράδειγμα

A: δισδιάστατος πίνακας στον οποίο είναι αποθηκευμένη η τετμημένη χ και η τεταγμένη y του κάθε πελάτη.

Προκειμένου να επιλύσουμε το πρόβλημα που έχουμε θα πρέπει να κατασκευάσουμε κάποιους επιπλέον πίνακες. Ο πρώτος πίνακας που χρειαζόμαστε είναι αυτός που θα υπολογίζει τις αποστάσεις μεταξύ όλων των κόμβων. Ο πίνακας αυτός συμβολίζεται ως C(i,j). Η απόσταση που υπολογίζεται είναι η ευκλείδεια, ο πίνακας είναι συμμετρικός που σημαίνει ότι η απόσταση που διανύουμε για να πάμε από τον κόμβο i στον κόμβο j, ισούται με την απόσταση από τον j στον i. Επίσης, η παραμονή σε ένα κόμβο απαγορεύεται για αυτό το λόγο η απόσταση C(i,i) μηδενίζεται έτσι ώστε να μην ληφθεί υπόψη στην επίλυση του προβλήματος. Η διάσταση του πίνακα C θα είναι nxn δηλαδή όσο είναι και το πλήθος των κόμβων (πελάτες και αποθήκη μαζί). Ο υπολογισμός της ευκλείδειας απόστασης έγινε ως εξής:

$$C(i,j)=\sqrt{((A(i,1)-A(j,1))^2-(A(i,2)-A(j,2))^2}$$

Θεωρούμε ότι ο χρόνος που θα χρειαστούμε για να μεταβούμε από τον πελάτη i στον πελάτη j ισούται με την απόσταση που θα διανύσουμε. Στην συνέχεια, όταν θα χρησιμοποιούμε τα δεδομένα του πίνακα C, θα αναφερόμαστε στο χρόνο.

Τώρα δίνονται όλες οι μεταβλητές και οι πίνακες που χρειάζονται για την επίλυση του προβλήματος.

4.3 Εύρεση αρχικής λύσης με τον αλγόριθμο του πλησιέστερου γείτονα

Μετά την αποθήκευση των δεδομένων το επόμενο βήμα είναι η υλοποίηση ενός απλού αλγόριθμου απληστίας για τον εντοπισμό της αρχικής λύσης. Στη συγκεκριμένη διπλωματική εργασία χρησιμοποιείται ο αλγόριθμος του πλησιέστερου γείτονα. Βάση, λοιπόν, του αλγορίθμου αυτού, ξεκινάμε από την αποθήκη και επισκεπτόμαστε τον πλησιέστερο πελάτη σε αυτήν. Από αυτόν συνεχίζουμε στον πλησιέστερό του, τον οποίο δεν έχουμε επισκεφτεί ξανά, και ολοκληρώνεται η διαδικασία όταν επισκεφτούμε όλους τους πελάτες και επιστρέψουμε πίσω στην αποθήκη, στο σημείο εκκίνησης. Ο βασικός επαναληπτικός βρόγχος ο οποίος έχει ως στόχο να ταξινομήσει τους πελάτες με βάση το διάνυσμα της απόστασης/χρόνου εκτελείται μέχρι να ελεγχθούν όλοι οι πελάτες. Ο αλγόριθμός ελέγχει σε κάθε βήμα αν το όχημα μπορεί να εξυπηρετήσει έναν νέο πελάτη χωρίς να παραβιάσει τους χρονικούς περιορισμούς. Αν οι περιορισμοί παραβιάζονται το όχημα επιστρέφει στην αποθήκη και ένα νέο όχημα ξεκινά. Η εκτέλεση συνεχίζεται μέχρι να εξυπηρετηθούν όλοι οι πελάτες.

Αρχικά αποθηκεύουμε τον πίνακα αποστάσεων **C** σε έναν νέο πίνακα **c**, ώστε να δουλεύουμε σε αυτόν. Ολες οι αλλαγές αποθηκεύονται στον πίνακα **c**. Θέλουμε να φτιάξουμε ένα διάνυσμα (**route**) ταξινομημένων πελατών. Κάθε φορά που θα επισκεπτόμαστε έναν πελάτη θα εξισώνουμε με το άπειρο την απόστασή του (**min=inf**), ώστε να είναι πάντα μεγαλύτερη της ελάχιστης και να μην τον ξαναεπισκεφτούμε. Μετά την ταξινόμηση του πίνακα θα πρέπει να ελεγχθούν οι περιορισμοί. Για να ελέγξουμε τον χρονικό περιορισμό θα πρέπει να δημιουργήσουμε μια μεταβλητή (**total**) η οποία θα κρατά το χρόνο που δαπανάται για να φτάσει το όχημα αρχικά από την αποθήκη στον πρώτο πελάτη, από τον πρώτο στον δεύτερο κ.ο.κ. συν το χρόνο εξυπηρέτησης (ίδιος για κάθε πελάτη). Βέβαια στον περιορισμό αυτό θα πρέπει να λαμβάνεται υπόψη και ο χρόνος επιστροφής στην αποθήκη. Για παράδειγμα έστω ότι έχουμε έναν συμμετρικό πίνακα αποστάσεων 5x5 (όπου d η αποθήκη) . Έστω, ότι ο συνολικός χρόνος που θα πρέπει να διανύσει το όχημα σε κάθε διαδρομή είναι 160.

	D	1	2	3	4
D	_	40	72	65	64
1	40	_	27	71	41
2	72	27	_	30	61
3	65	71	30	_	24
4	64	41	61	24	_

Ξεκινώντας από την αποθήκη πλησιέστερος στην αποθήκη πελάτης είναι ο 1 και από τον 1 πλησιέστερος είναι ο 2. Πλησιέστερος στον 2 είναι ο πελάτης 4. Από τον 4 δεν μπορούμε να μεταβούμε στον 3 διότι θα παραβιαστεί ο περιορισμός του χρόνου. Επομένως, επιστρέφουμε στην αποθήκη και ελέγχουμε ποιος είναι ο επόμενος πλησιέστερος πελάτης που δεν έχουμε επισκεφθεί. Είναι ο 3, ο οποίος είναι και ο τελευταίος πελάτης που δεν είχαμε επισκεφθεί έως τώρα. Επιστρέφουμε στην αποθήκη. Άρα, οι διαδρομές διαμορφώνονται ως εξής:

$$D \rightarrow 1 \rightarrow 2 \rightarrow 4 \rightarrow D$$

$$D \rightarrow 3 \rightarrow D$$

Με τον τρόπο αυτό κατασκευάζουμε ένα νέο διάνυσμα (**ROUTE**), το οποίο περιλαμβάνει όλες τις διαδρομές που θα πραγματοποιηθούν. Περιλαμβάνει τον γυρισμό στην αποθήκη όποτε κρίνεται αναγκαίο, λόγω του περιορισμού. Υπολογίζουμε, ακόμη, το συνολικό κόστος όλων των διαδρομών (**COST).** Στην συνέχεια ακολουθεί σύγκριση των λύσεων του αλγορίθμου της μεταβλητής γειτονιάς αναζήτησης και κρατάμε κάθε φορά την καλύτερη. Εκχωρούμε επίσης στην μεταβλητή **bestk** τους επισκέψιμους κόμβους που μας δίνει κάθε φορά η λύση.

4.4 Υλοποίηση του αλγορίθμου μεταβλητής γειτονιάς αναζήτησης

Αφού έχουμε βρει μια αρχική εφικτή λύση με τον αλγόριθμο του πλησιέστερου γείτονα, ακολουθεί ο αλγόριθμος μεταβλητής γειτονιάς αναζήτησης για την εύρεση καλύτερης λύσης.

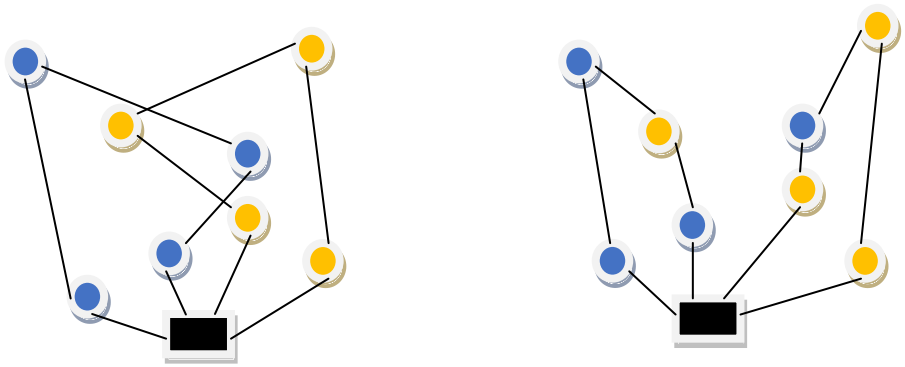
Ο αλγόριθμος VNS χαρακτηρίζεται από το γεγονός ότι αναζητά πιθανές βέλτιστες λύσεις σε όλο και μεγαλύτερες γειτονιές μιας τρέχουσας λύσης. Στην περίπτωση αυτή, θα αναζητήσουμε διαδοχικά λύσεις που διαφέρουν από την τρέχουσα λύση σε k=1,2,3,… θέσεις και στη συνέχεια θα εφαρμόσουμε τοπική αναζήτηση γύρω από τη νέα υποψήφια λύση. Στην περίπτωση που προκύψει λύση που δίνει μικρότερο κόστος και ικανοποιεί και τον περιορισμό τότε η αναζήτηση ξεκινά ξανά με k=1 και έχοντας τη νέα λύση σαν αρχική.

Αρχικά, αποθηκεύουμε σε ένα διάνυσμα **temproute** το ταξινομημένο διάνυσμα των πελατών, **route**, ώστε να αποθηκευτούν σε αυτό όποιες αλλαγές πραγματοποιήσουμε. Επομένως, η σύγκριση των κόστων των νέων παραγόμενων λύσεων θα γίνεται με τη μεταβλητή TEMPCOST στην οποία αποθηκεύεται το κόστος και κάθε νέας παραγόμενης λύσης. Αν είναι καλύτερη κάποια νέα λύση θα αποθηκεύουμε το κόστος της και το διάνυσμα της, στις μεταβλητές **COST** και **ROUTE,** αντίστοιχα.

Για κάποιον καθορισμένο αριθμό επαναλήψεων πραγματοποιούνται οι τοπικές αναζητήσεις που αναλύουμε παρακάτω. Ξεκινάμε από την 1-1 ανταλλαγή, συνεχίζουμε με την 1-0 επανατοποθέτηση και ολοκληρώνουμε με την 2-opt. Σε περίπτωση που βρεθεί κάποια καλύτερη λύση αποθηκεύουμε αυτή και το αντίστοιχο κόστος της στις μεταβλητές **COST** και **ROUTE,** αντίστοιχα.

4.4.1 Εφαρμογή τοπικής αναζήτησης 1-1 ανταλλαγή (1-1 exchange)

Η συνάρτηση αυτή κάνει ανταλλαγές κόμβων μεταξύ διαφορετικών δρομολογίων, είτε κόμβων του ίδιου δρομολογίου.



Σχήμα 4.1: 1-1 exhange

Αρχικά, αποθηκεύουμε το διάνυσμα **route** σε έναν καινούριο διάνυσμα **temproute.** Χρησιμοποιώντας δυο βοηθητικές μεταβλητές, εκχωρούμε σε αυτές, με την βοήθεια της συνάρτησης randi(n-1), δυο τυχαίες τιμές από το 1 μέχρι το n, όπου n είναι ο αριθμός των κόμβων. Με αυτόν τον τρόπο δεν λαμβάνουμε υπόψη την αποθήκη. Εδώ, πρέπει να σημειωθεί ότι οι μεταβλητές αυτές αντιπροσωπεύουν θέσεις στο διάνυσμα temproute και όχι στοιχεία. Στην συνέχεια, κάνουμε αντιμετάθεση των δύο μεταβλητών. Για παράδειγμα, έστω το διάνυσμα:

6	5	**8**	2	9	**3**	1

αν το randi(7)=3 και το άλλο randi(7)=6, τότε θα γίνει αντιμετάθεση του κελιού 3, δηλαδή του πελάτη 8, με το κελί 6, δηλαδή τον πελάτη 3, οπότε το διάνυσμα θα γίνει:

6	5	**3**	2	9	**8**	1

Εφόσον έχει γίνει η ανταλλαγή μεταξύ δυο κελιών είτε του ίδιου δρομολογίου είτε διαφορετικών ελέγχεται και πάλι ο περιορισμός του χρόνου και κατασκευάζεται το **TEMPROUTE** το οποίο περιλαμβάνει όλες τις διαδρομές και το **TEMPCOST.** Αν το

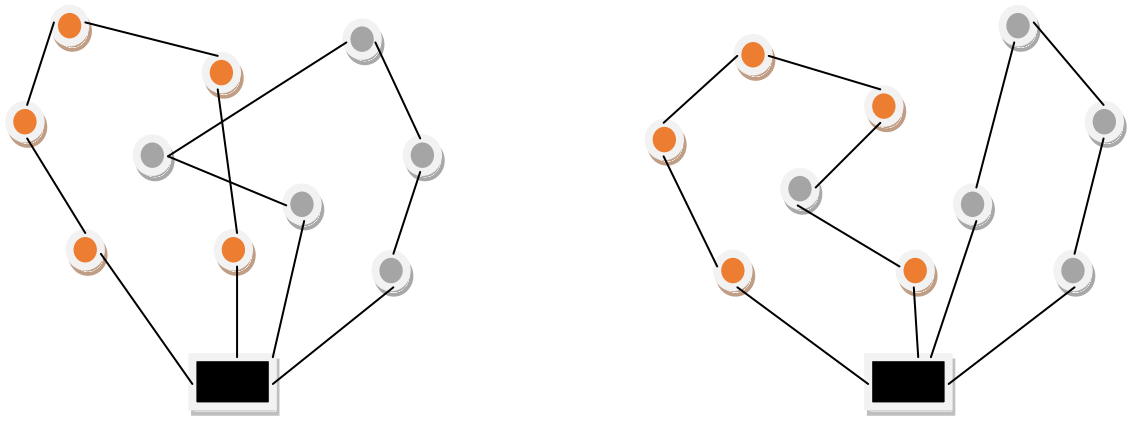
TEMPCOST είναι μικρότερο του κόστους **COST** που βρήκαμε κατά την κατασκευή της αρχικής εφικτής λύσης ή της προηγούμενης επανάληψης, τότε στην μεταβλητή **COST** θα εκχωρηθεί το **TEMPCOST** και στο **ROUTE** θα εκχωρηθεί το διάνυσμα **TEMPROUTE**. Σε αντίθετη περίπτωση, θα διατηρηθούν οι τιμές του **ROUTE** και του **COST** που βρήκαμε προηγουμένως.

4.4.2 Εφαρμογή τοπικής αναζήτησης 1-0 επανατοποθέτηση(1-0 relocate)

Με τη μέθοδο 1-0 relocate μπορούμε να πάρουμε έναν κόμβο από τη θέση του και να τον επανατοποθετήσουμε μεταξύ δύο άλλων. Η μέθοδος Relocate βασίζεται στην απλή ιδέα της διαγραφής ενός πελάτη από μια διαδρομή και επανατοποθέτησή του σε μια άλλη διαδρομή με καλύτερο κόστος.

Παρουσιάζεται η διαδικασία σε βήματα :

Βήμα 1:
Υπολογισμός της απόστασης που διανύει ο κάθε πωλητής – όχημα που θέλουμε να επανατοποθετήσουμε κόμβο που επισκέπτεται καθώς και της διαδρομής που πρόκειται να αποσταλεί ο κόμβος αυτός.
Βήμα 2:
Στη συνέχεια υπολογίζουμε τις νέες αποστάσεις αφού τοποθετήσουμε τον κόμβο στο νέο μονοπάτι και τον αφαιρέσουμε από τον παλιό.
Βήμα 3:
Ελέγχουμε αν οι νέες αποστάσεις είναι μικρότερες από τις προηγούμενες και αν είναι τότε τοποθετούμε τον κόμβο στην διαδρομή του νέου οχήματος.
Βήμα 4:
Επαναλαμβάνουμε την διαδικασία από το Βήμα 1 έως ότου δεν υπάρχει κάποια αλλαγή
Βήμα 5:
Τερματίζουμε τον αλγόριθμο 1-0 Επανατοποθέτηση (1-0 Relocate) και επιστρέφουμε τα νέα μονοπάτια.



Σχήμα 4.2: 1-0 relocate

Τα αρχικά βήματα που πραγματοποιήσαμε για τη συγγραφή του αλγόριθμου τοπικής αναζήτησης 1-0 relocate είναι τα ίδια που πραγματοποιήσαμε και στον παραπάνω αλγόριθμο. Κατασκευάσαμε το διάνυσμα temproute και εκχωρήσαμε στις βοηθητικές μεταβλητές temp1 και temp2 δύο τυχαίες τιμές (randi(n-1)),ώστε να ξεκινήσει η αναζήτηση.

Για παράδειγμα, σε ένα διάνυσμα:

6	5	**8**	2	9	**3**	1

αν το randi(7)=3 και το άλλο randi(7)=6, τότε το κελί 3, δηλαδή ο πελάτης 8, θα τοποθετηθεί μετά τον κόμβο 6, δηλαδή τον πελάτη 3. Όλα τα ενδιάμεσα κελιά θα μετακινηθούν μία θέση οπότε το διάνυσμα θα γίνει :

6	5	2	9	**3**	**8**	1

Μετά την τοποθέτηση του κελιούου στην νέα διαδρομή, ελέγχεται και πάλι ο περιορισμός του χρόνου. Με τη βοήθεια του **TEMPROUTE** και του **TEMPCOST** συγκρίνουμε το κόστος της διαδρομής που μόλις δημιουργήσαμε με το κόστος που είχαμε καταχωρήσει πριν γίνει η επανατοποθέτηση (**COST**). Σε κάθε επανάληψη, καταχωρούμε διαδρομή που έχει το μικρότερο κόστος.

4.4.3 Εφαρμογή τοπικής αναζήτησης 2-opt.

Μια κίνηση 2-opt διαγράφει δύο άκρα, χωρίζοντας τη διαδρομή σε δύο μέρη, και στη συνέχεια επανασυνδέει τα μονοπάτια με άλλον δυνατό τρόπο, ώστε να έχουμε μείωση του κόστους. Αυτό ισοδυναμεί με αντιστροφή της σειράς των πελατών μεταξύ των δύο άκρων.

Η διαδικασία της μεθόδου 2-opt είναι η ακόλουθη:
Βήμα 1:
Έστω T η τρέχουσα διαδρομή.
Βήμα 2:
Για κάθε κόμβο i=1,...,n: Εξετάζουμε όλες τις πιθανές 2-opt κινήσεις που μπορεί να γίνουν από τον i και την επόμενη της μέσα στην διαδρομή. Αν με αυτό τον τρόπο μπορούμε να μειώσουμε το κόστος της διαδρομής, τότε επιλέγουμε την καλύτερη 2-opt κίνηση και εφαρμόζουμε τις αλλαγές στην διαδρομή T.
Βήμα 3:
Αν δεν μπορούμε να βρούμε επιπλέον βελτίωση, σταματάμε.

Στην χειρότερη περίπτωση, μπορεί μονάχα να εγγυηθεί ότι μια κίνηση βελτίωσης μειώνει το μήκος της διαδρομής τουλάχιστον μια μονάδα.

Δουλεύουμε και εδώ όπως και στους προηγούμενους αλγορίθμους χρησιμοποιώντας το temproute, το temp1 και το temp2.

Για παράδειγμα, σε ένα διάνυσμα:

6	5	**8**	2	9	**3**	1

αν το randi(7)=3 και το άλλο randi(7)=6, τότε το κελί 3, δηλαδή ο πελάτης 8, θα τοποθετηθεί στη θέση του κόμβου 6, δηλαδή θα γίνει αντιμετάθεση με τον πελάτη 3. Στη συνέχεια γίνεται αντιστροφή όλων των ενδιάμεσων κελιών/πελατών. Οπότε το συγκεκριμένο διάνυσμα θα γίνει:

6	5	**3**	**9**	**2**	**8**	1

Κατά την ολοκλήρωση του αλγορίθμου τοπικής αναζήτησης 2-opt γίνεται και πάλι έλεγχος του περιορισμού του χρόνου και σύγκριση του κόστους της διαδρομής.

4.5 Τερματισμός αλγορίθμου και ψευδοκώδικας

Η επαναληπτική διαδικασία συνεχίζεται για όσες επαναλήψεις προκαθορίσουμε εμείς. Όσο περισσότερες είναι οι επαναλήψεις τόσο αυξάνονται οι πιθανότητες να βρούμε καλύτερο αποτέλεσμα. Στο τέλος κρατάμε τα δρομολόγια που παρουσιάζουν το χαμηλότερο κόστος, ανεξάρτητα από τη μέθοδο της τοπικής αναζήτησης από την οποία προέκυψε.

Η διαδικασία επίλυσης του προβλήματος παρουσιάζεται στον ακόλουθο ψευδοκώδικα:

Αρχικοποίηση
Επέλεξε παράδειγμα
Υπολόγισε πίνακα κόστους C
Πάραξε την πρώτη διαδρομή με πλησιέστερο γείτονα (ROUTE)
λαμβάνοντας υπόψη τους περιορισμούς
Υπολόγισε το συνολικό κόστος διαδρομών (COST)
Επέλεξε αριθμό λύσεων VNS (e)
Κυρίως αλγόριθμος
Για e επαναλήψεις
Αποθήκευσε το ROUTE στο διάνυσμα TEMPROUTE
Κάνε 1-1 exchange στην TEMPROUTE
Υπολόγισε το κόστος της TEMPROUTE (TEMPCOST)
Αν TEMPCOST≤ COST τότε
ROUTE=TEMPROUTE
COST= TEMPCOST
Τέλος_Αν
Αποθήκευσε το ROUTE στο διάνυσμα TEMPROUTE
Κάνε 1-0 relocate στην TEMPROUTE
Υπολόγισε το κόστος της TEMPROUTE (TEMPCOST)

Αν TEMPCOST≤ COST τότε
ROUTE=TEMPROUTE
COST= TEMPCOST
Τέλος_Αν
Αποθήκευσε το ROUTE στο διάνυσμα TEMPROUTE
Κάνε 2-opt στην TEMPROUTE
Υπολόγισε το κόστος της TEMPROUTE (TEMPCOST)
Αν TEMPCOST≤ COST τότε
ROUTE=TEMPROUTE
COST= TEMPCOST
Τέλος_Αν

Τέλος_για

Επέστρεψε το ROUTE και το COST

ΚΕΦΑΛΑΙΟ 5 : ΠΕΡΙΓΡΑΦΗ ΚΑΙ ΑΝΑΛΥΣΗ ΤΩΝ ΑΠΟΤΕΛΕΣΜΑΤΩΝ

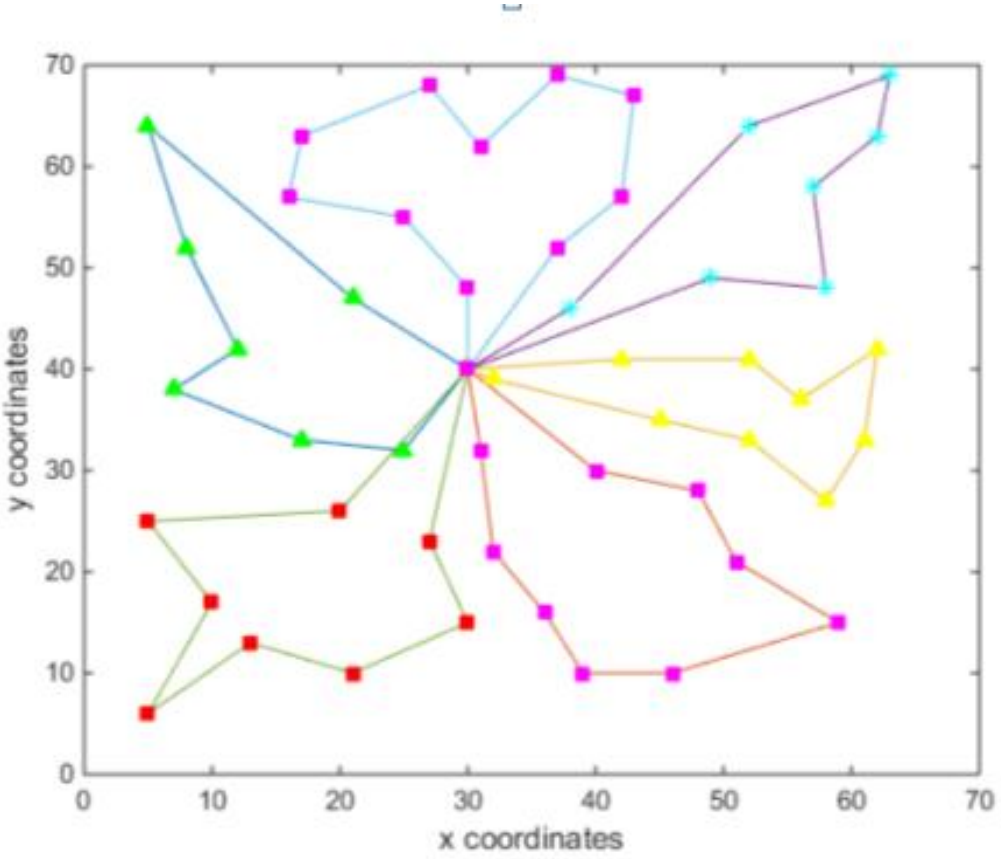
Στην παρούσα διπλωματική εργασία επιλύθηκαν επτά προβλήματα δρομολόγησης οχημάτων. Για κάθε πρόβλημα είχαμε ως δεδομένα το πλήθος των κόμβων, το χρόνο παραμονής σε κάθε πελάτη και το συνολικό χρόνο της διαδρομής. Για κάθε πρόβλημα έγιναν πολλές δοκιμές στον κώδικα, ώστε να επιτευχθούν τα καλύτερα δυνατά αποτελέσματα. Σε κάθε περίπτωση χρησιμοποιήσαμε τρεις κατηγορίες επαναλήψεων. Ο αριθμός των επαναλήψεων κυμαίνεται από 20.000 έως και 2.000.000. Σε κάθε πρόβλημα πραγματοποιήθηκαν 10 εκτελέσεις του αλγορίθμου ώστε να υπάρχει μια ολοκληρωμένη εικόνα της απόδοσής του.

5.1 Παράδειγμα par6 με 51 κόμβους

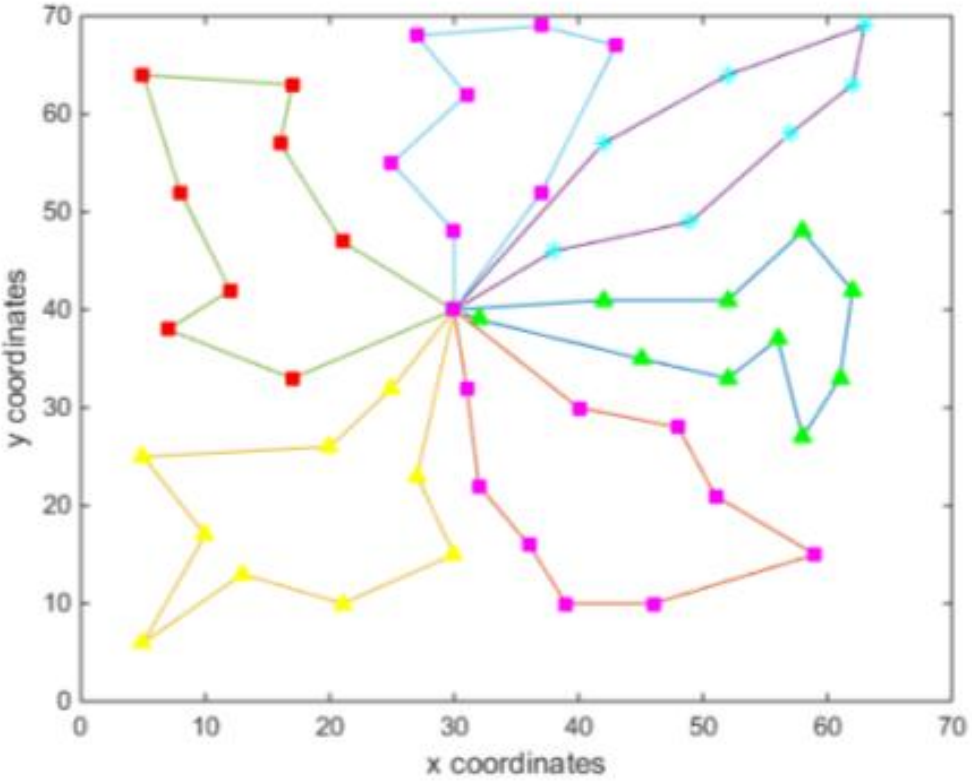
Παρουσιάζεται ο πίνακας αποτελεσμάτων των 10 εκτελέσεων του αλγορίθμου για τις 3 κατηγορίες του αριθμού επαναλήψεων (20.000,200.000 και 2.000.000), καθώς και τα γραφήματα των καλύτερων λύσεων που βρέθηκαν σε κάθε κατηγορία.

par6	Επαναλήψεις αλγορίθμου		
No τρεξίματος	e=20.000	e=200.000	e=2.000.000
1	572,8599	562,5885	589,8081
2	638,5825	639,0149	639,2832
3	736,7912	635,5918	658,7654
4	653,447	652,7011	581,5834
5	663,179	666,106	666,5121
6	651,9192	686,339	654,6352
7	632,2891	651,0558	671,5467
8	642,0277	657,7515	639,734
9	633,4503	589,6407	663,5525
10	664,8911	569,7084	630,971
Μ.Ο.	***648,9437***	***631,0498***	***639,6391***

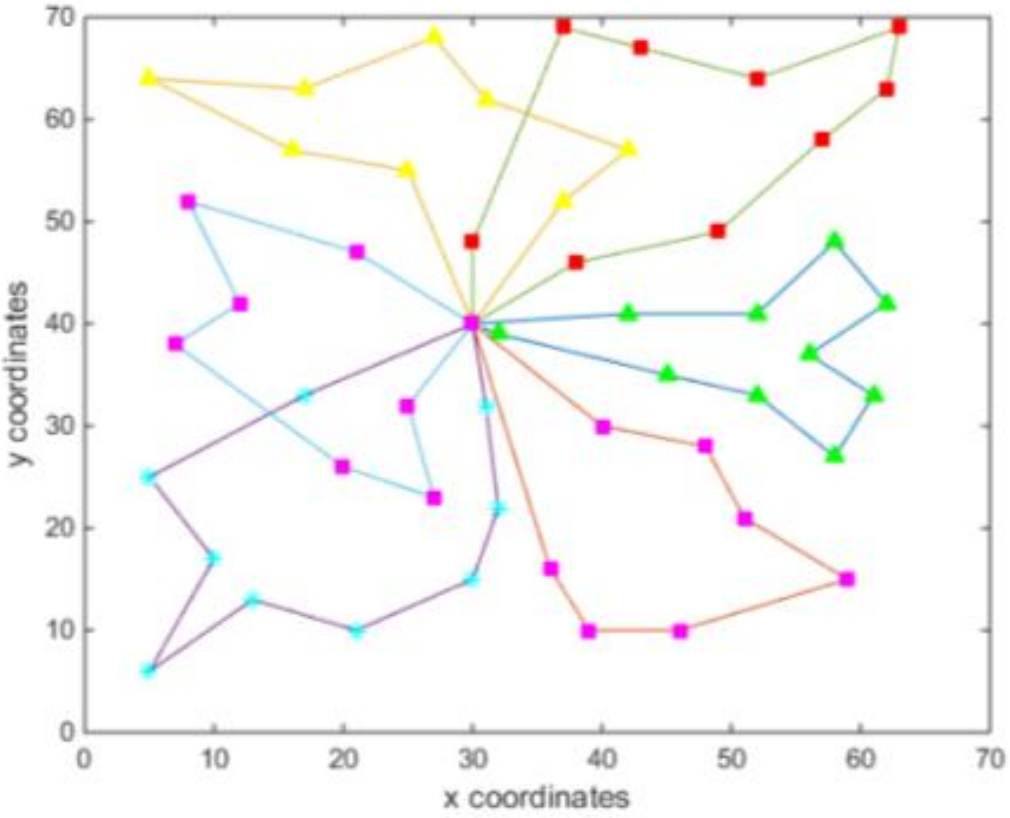
Εικόνα 5.1.1 Βέλτιστη διαδρομή για παράδειγμα par6 με e=20.000 (Βέλτιστη εκτέλεση αλγορίθμου η Νο 1 με COST=572,8599)



Εικόνα 5.1.2 Βέλτιστη διαδρομή για παράδειγμα par6 με e=200.000 (Βέλτιστη εκτέλεση αλγορίθμου η Νο 1 με COST=562,5885)



Εικόνα 5.1.3 Βέλτιστη διαδρομή για παράδειγμα par6 με e=2.000.000 (Βέλτιστη εκτέλεση αλγορίθμου η No 4 με COST=581,5834)



Η καλύτερη λύση για το παράδειγμα par 6 των 51 κόμβων, βρέθηκε για 200.000 επαναλήψεις του VNS και η τιμή της είναι **COST=562,5885**. Επίσης, παρατηρούμε ότι ο καλύτερος μέσος όρος παρουσιάζεται στις 200.000 επαναλήψεις (**M.O.= *631,04980*).**

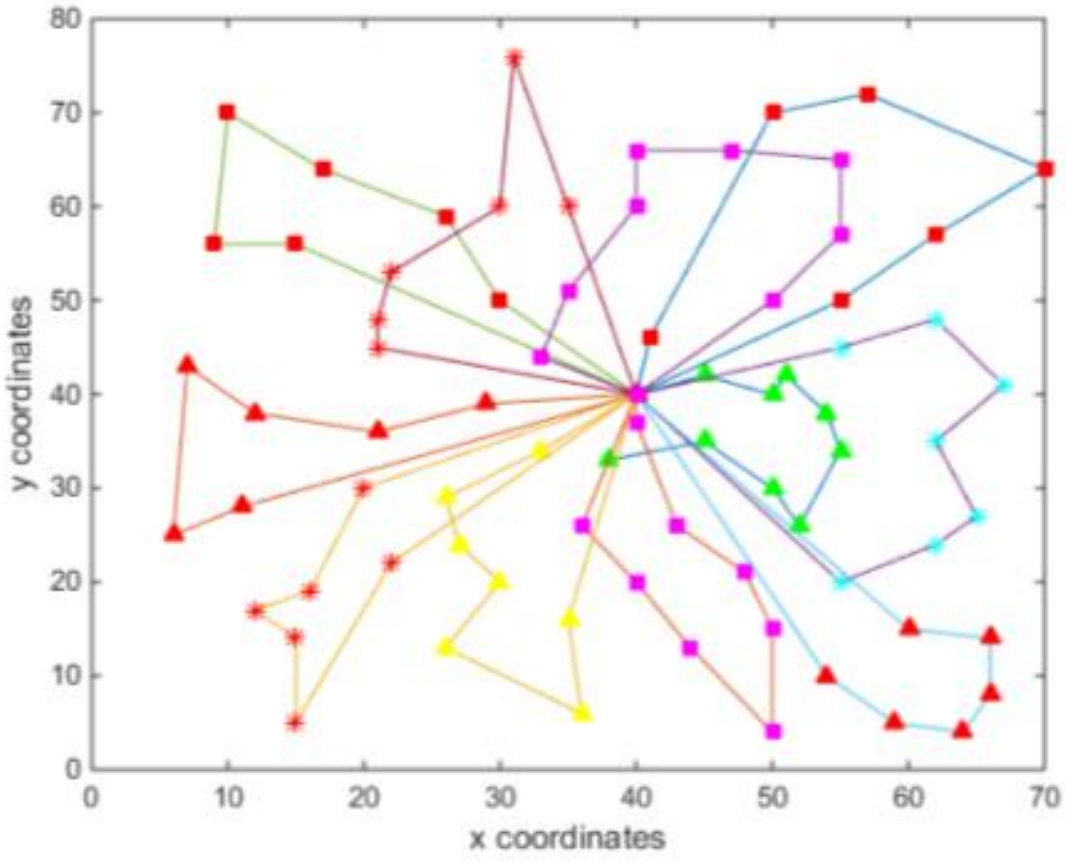
5.2 Παράδειγμα par7 με 76 κόμβους

Ακολουθεί ο πίνακας αποτελεσμάτων για τις 3 κατηγορίες επαναλήψεων. Σε κάθε κατηγορία γίναν και πάλι 10 εκτελέσεις του αλγορίθμου.

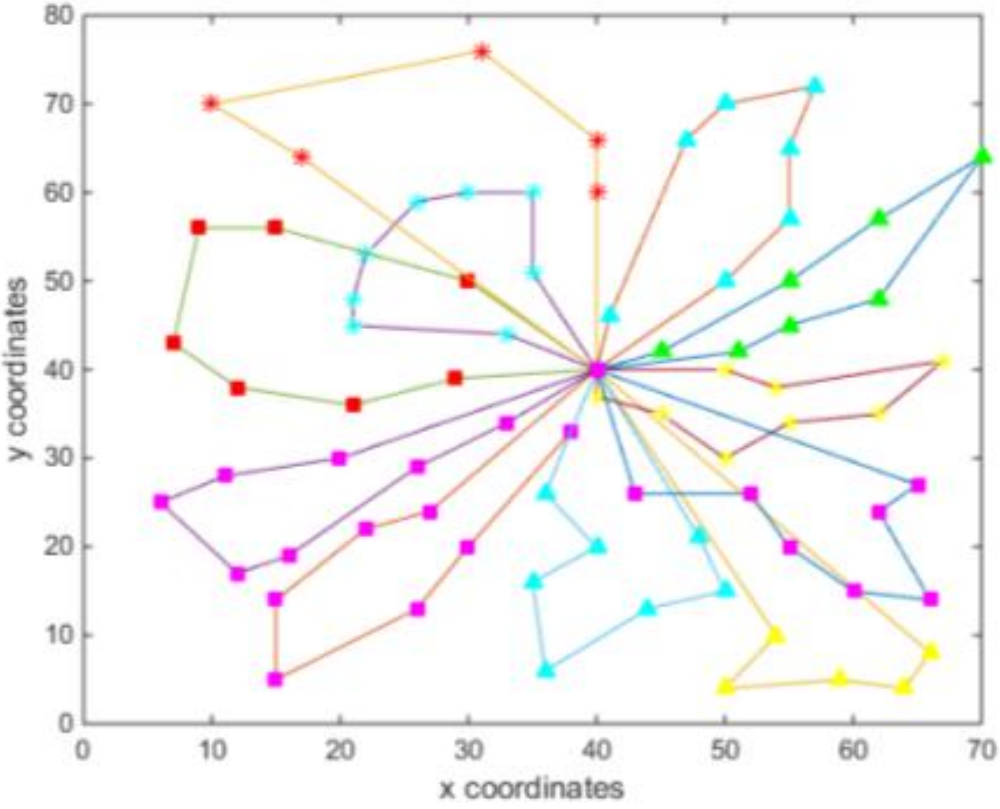
par7	Επαναλήψεις αλγορίθμου		
No τρεξίματος	e=20.000	e=200.000	e=2.000.000
1	1013,288	930,5361	968,229
2	974,6068	955,1903	936,5607
3	986,0217	937,3716	973,1088
4	1013,288	919,2848	953,1883
5	1005,957	961,9971	970,3851
6	974,6068	943,0859	966,175
7	986,0217	955,6344	918,4179
8	1005,957	947,0805	920,4677
9	973,8843	954,7773	944,8191
10	940,2588	944,8191	967,6408
Μ.Ο.	***987,3889***	***944,9777***	***951,8992***

Ακολουθούν τα γραφήματα των καλύτερων λύσεων για κάθε κατηγορία:

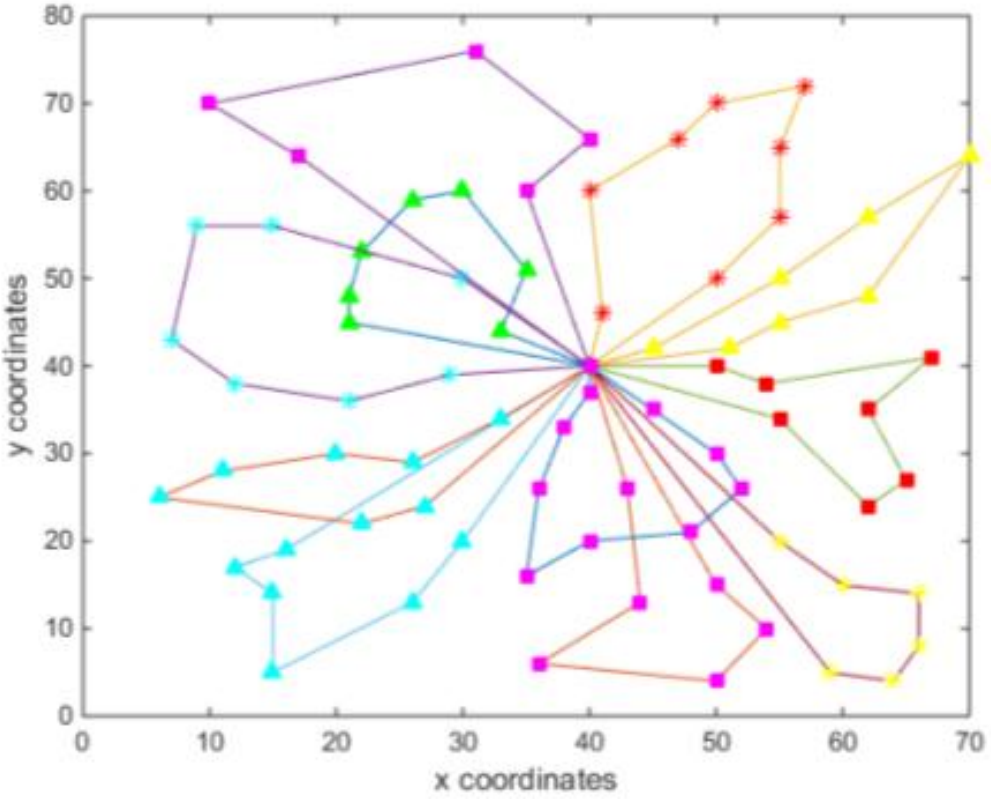
Εικόνα 5.2.1 Βέλτιστη διαδρομή για παράδειγμα par7 με e=20.000 (Βέλτιστη εκτέλεση αλγορίθμου η No 10 με COST=940,2588)



Εικόνα 5.2.2 Βέλτιστη διαδρομή για παράδειγμα par7 με e=200.000 (Βέλτιστη εκτέλεση αλγορίθμου η Νο 4 με COST=919,2848)



Εικόνα 5.2.3 Βέλτιστη διαδρομή για παράδειγμα par7 με e=2.000.000 (Βέλτιστη εκτέλεση αλγορίθμου η No 7 με COST=918,4179)



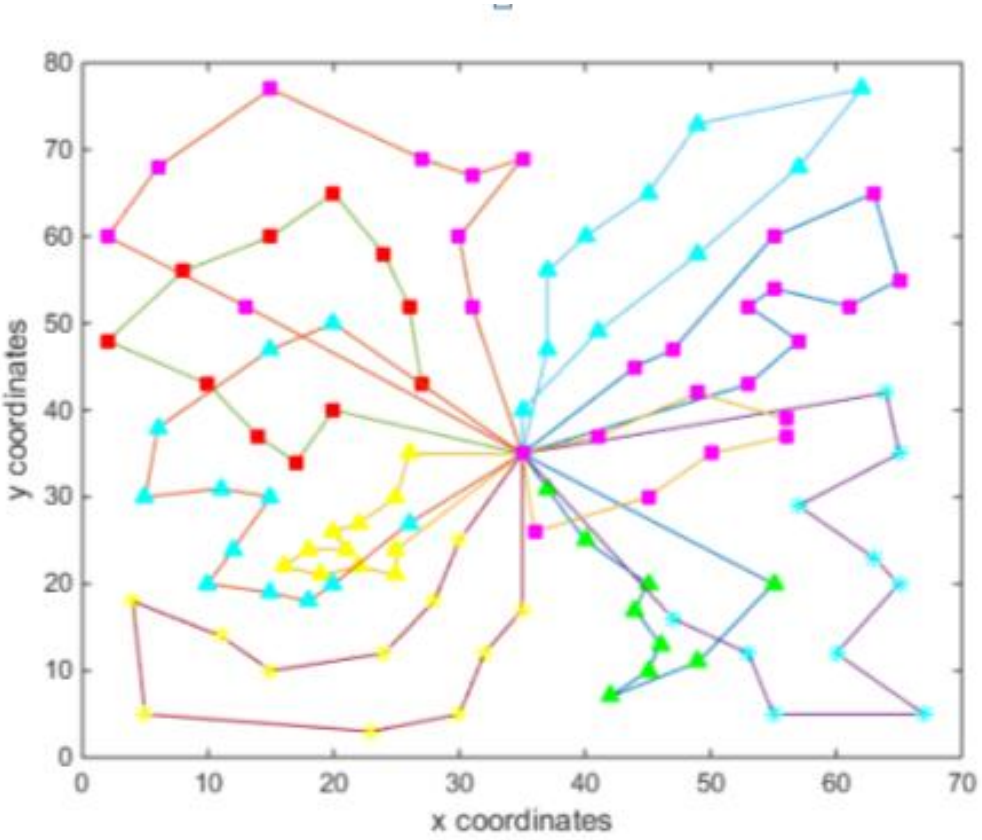
Βέλτιστη λύση για το παράδειγμα par7 των 76 κόμβων, βρέθηκε για 2.000.000 επαναλήψεις του VNS και η τιμή της είναι **COST=918,4179**. Όμως ο καλύτερος μέσος όρος παρουσιάζεται στις 200.000 επαναλήψεις του αλγορίθμου (**Μ.Ο.= *944,9777*).**

5.3 Παράδειγμα par8 με 101 κόμβους

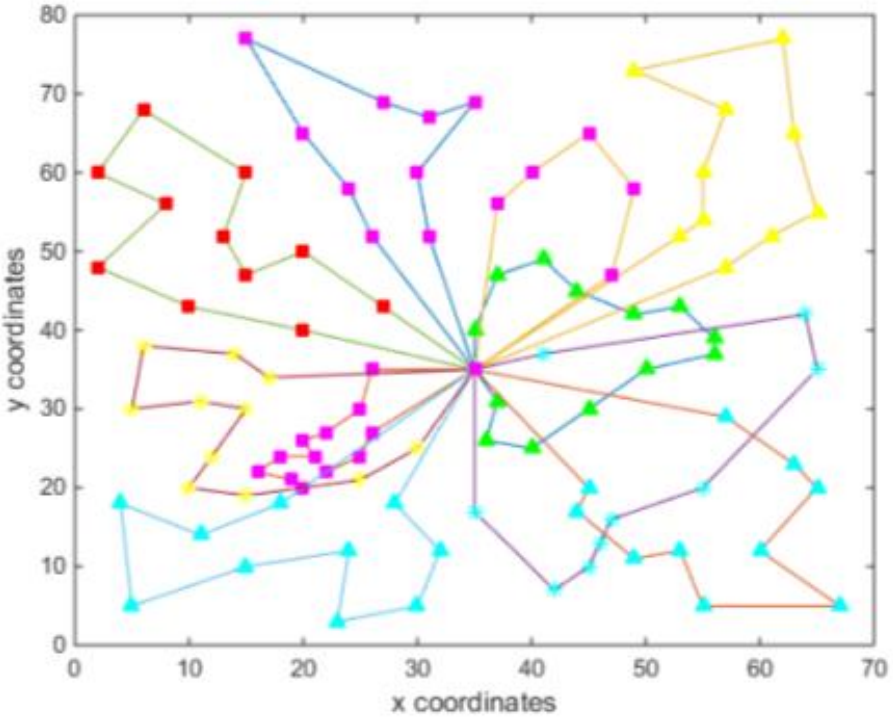
Στη συνέχεια παρουσιάζεται ο πίνακας αποτελεσμάτων για το παράδειγμα par8 με 101 κόμβους, καθώς και τα γραφήματα των καλύτερων αποτελεσμάτων. Εκτελέστηκε ο αλγόριθμος 10 φορές για κάθε είδος επανάληψης.

par8	Επαναλήψεις αλγορίθμου		
No τρεξίματος	e=20.000	e=200.000	e=2.000.000
1	993,7423	970,3141	993,903
2	984,3317	1001,068	1008,425
3	968,023	1007,463	993,7569
4	1017,613	991,306	967,9806
5	985,6018	979,8192	997,1907
6	978,2704	990,3494	971,2778
7	1005,514	967,5788	974,0305
8	999,9264	967,9806	984,8682
9	985,5509	999,7769	1001,068
10	995,0559	993,7569	984,2276
Μ.Ο.	***991,363***	***986,9412***	***987,6727***

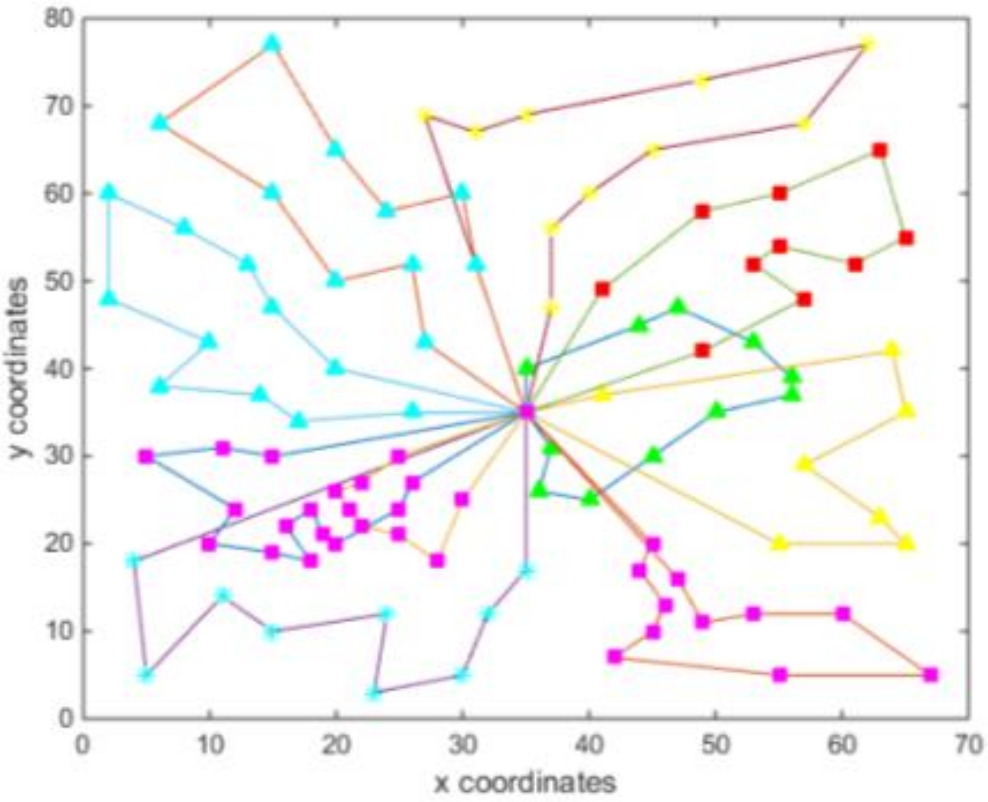
Εικόνα 5.3.1 Βέλτιστη διαδρομή για παράδειγμα par8 με e=20.000 (Βέλτιστη εκτέλεση αλγορίθμου η No 3 με COST=968,023)



Εικόνα 5.3.2 Βέλτιστη διαδρομή για παράδειγμα par8 με e=200.000 (Βέλτιστη εκτέλεση αλγορίθμου η No 7 με COST=967,5788)



Εικόνα 5.3.3 Βέλτιστη διαδρομή για παράδειγμα par8 με e=2.000.000 (Βέλτιστη εκτέλεση αλγορίθμου η No 4 με COST=967,9806)



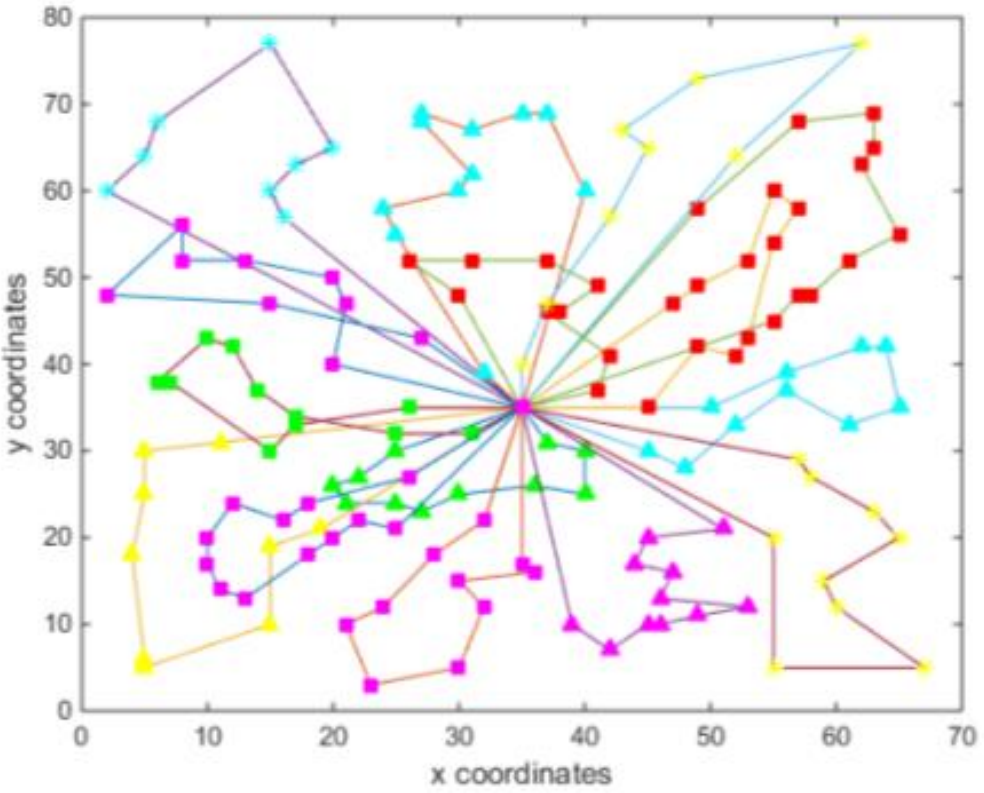
Παρατηρούμε πως η καλύτερη λύση έχει βρεθεί για 200.000 επαναλήψεις του αλγορίθμου (**COST=967,5788**), η διαφορά βέβαια με την καλύτερη λύση που βρήκαμε στις 2.000.000 επαναλήψεις είναι πολύ μικρή (COST=967,9806). Παρόλα αυτά καλύτερος μέσος όρος βρέθηκε και πάλι στις 200.000 επαναλήψεις του αλγορίθμου (**M.O.=986,9412**).

5.4 Παράδειγμα par9 με 151 κόμβους

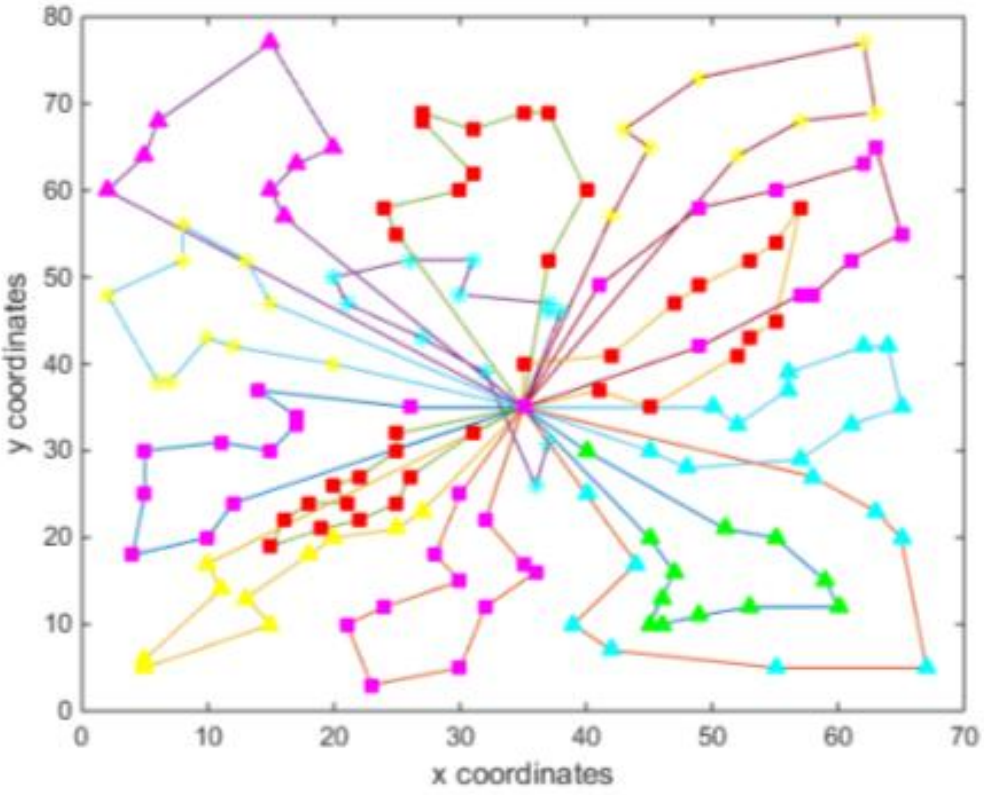
Παρουσιάζεται ο πίνακας αποτελεσμάτων των 10 εκτελέσεων του αλγορίθμου για τις 3 κατηγορίες επαναλήψεων για το par9 των 151 κόμβων.

par9	Επαναλήψεις αλγορίθμου		
No τρεξίματος	e=20.000	e=200.000	e=2.000.000
1	1308,502	1295,042	1295,03
2	1315,028	1292,132	1236,386
3	1340,364	1224,023	1273,591
4	1325,973	1299,178	1269,333
5	1331,325	1294,805	1224,024
6	1322,127	1261,637	1319,549
7	1281,074	1226,213	1296,907
8	1327,28	1295,793	1303,219
9	1322,127	1273,947	1224,023
10	1370,203	1296,535	1,25E+03
M.O.	***1324,4***	***1275,931***	***1269,089***

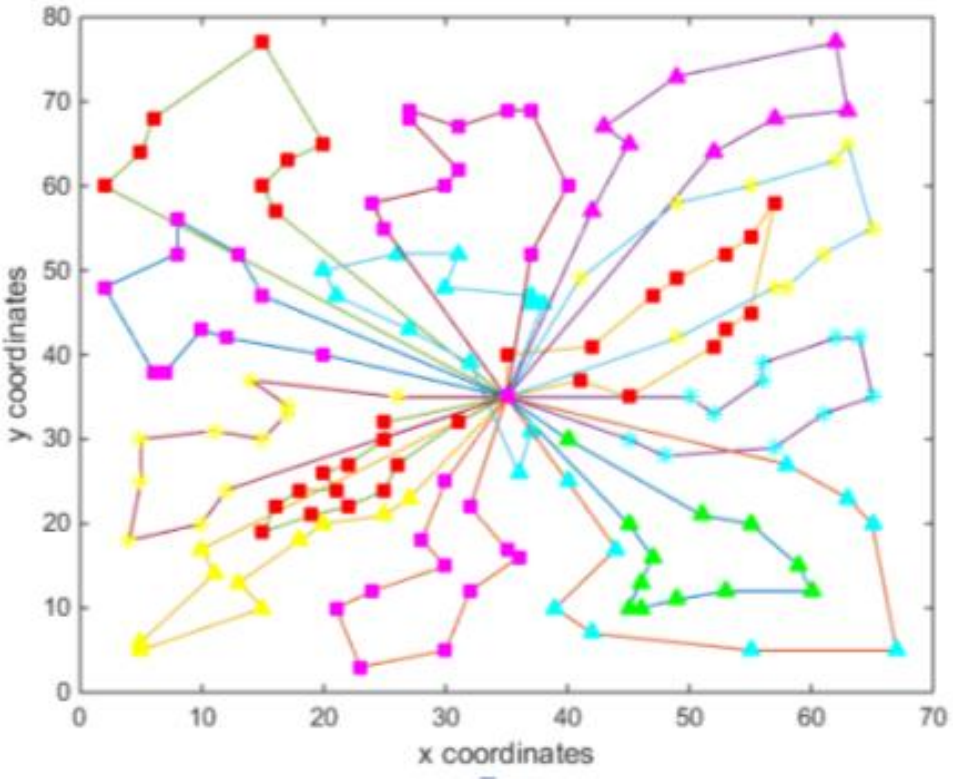
Εικόνα 5.4.1 Βέλτιστη διαδρομή για παράδειγμα par9 με e=20.000 (Βέλτιστη εκτέλεσ η αλγορίθμου η No 7 με COST=1281,074)



Εικόνα 5.4.2 Βέλτιστη διαδρομή για παράδειγμα par9 με e=200.000 (Βέλτιστη εκτέλεση αλγορίθμου η Νο 3 με COST=1224,023)



Εικόνα 5.4.3 Βέλτιστη διαδρομή για παράδειγμα par9 με e=2.000.000 (Βέλτιστη εκτέλεση αλγορίθμου η No 5 με COST=1224,023)



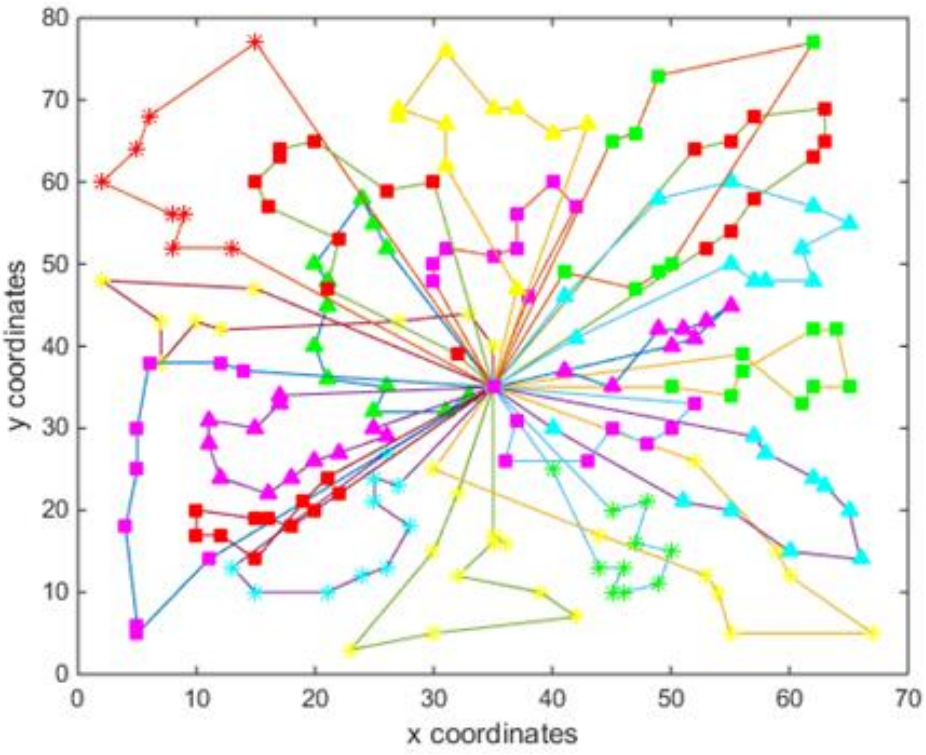
Σε αυτήν την περίπτωση η βέλτιστη λύση είναι σχεδόν κοινή στη δεύτερη και στην τρίτη κατηγορία επαναλήψεων (e=200.000 και e=2.000.000 αντίστοιχα), καθώς η διαφορά τους είναι ελάχιστη. Βέλτιστο κόστος **COST=1224,023**. Επίσης, παρατηρούμε ότι οι λύσεις βελτιώνονται με την αύξηση του αριθμού των επαναλήψεων (e=2.000.000) λόγω της μεγαλύτερης διερεύνησης που πραγματοποιείται στις γειτονίες αναζήτησης.(**M.O=1269,089**)

5.5 Παράδειγμα par10 με 200 κόμβους

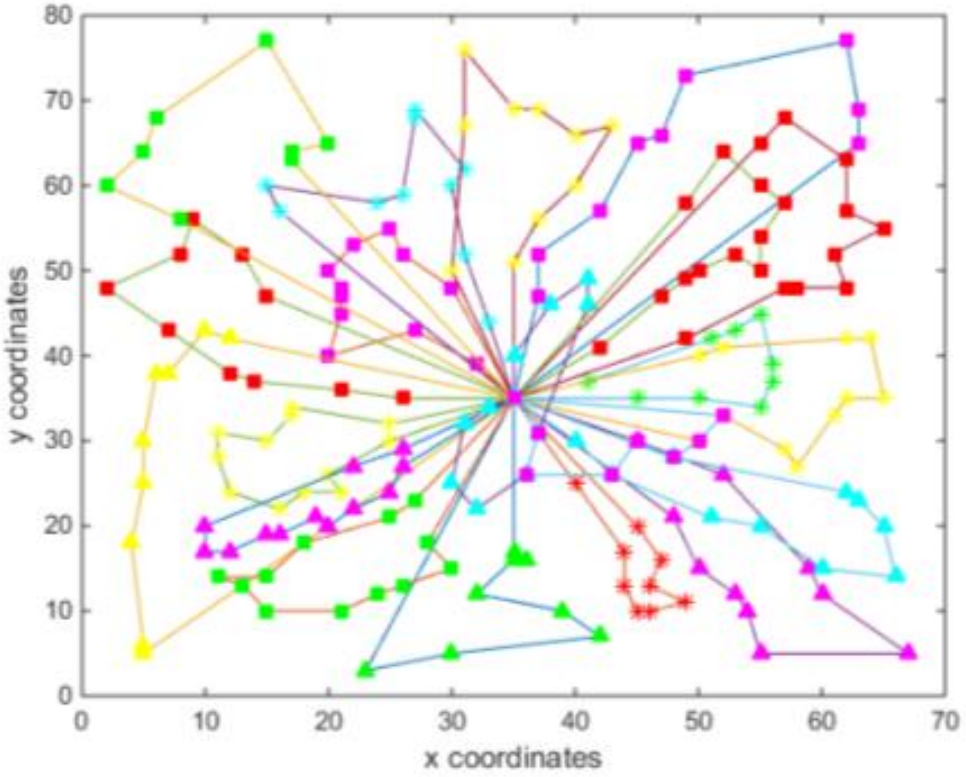
Στο παράδειγμα par10 των 200 κόμβων κάναμε την ίδια διαδικασία με τα προηγούμενα παραδείγματα για τις τρεις κατηγορίες επαναλήψεων. Τα αποτελέσματα που προέκυψαν φαίνονται παρακάτω:

par10	Επαναλήψεις αλγορίθμου		
No τρεξίματος	e=20.000	e=200.000	e=2.000.000
1	1721,521	1570,804	1611,004
2	1719,175	1603,303	1560,386
3	1649,16	1631,303	1551,193
4	1727,476	1594,84	1604,81
5	1727,297	1573,348	1469,787
6	1692,478	1650,994	1579,581
7	1682,892	1566,177	1511,908
8	1719,193	1669,03	1593,964
9	1623,469	1614,462	1555,566
10	1724,896	1555,62	1581,733
Μ.Ο.	***1698,756***	***1602,988***	***1561,993***

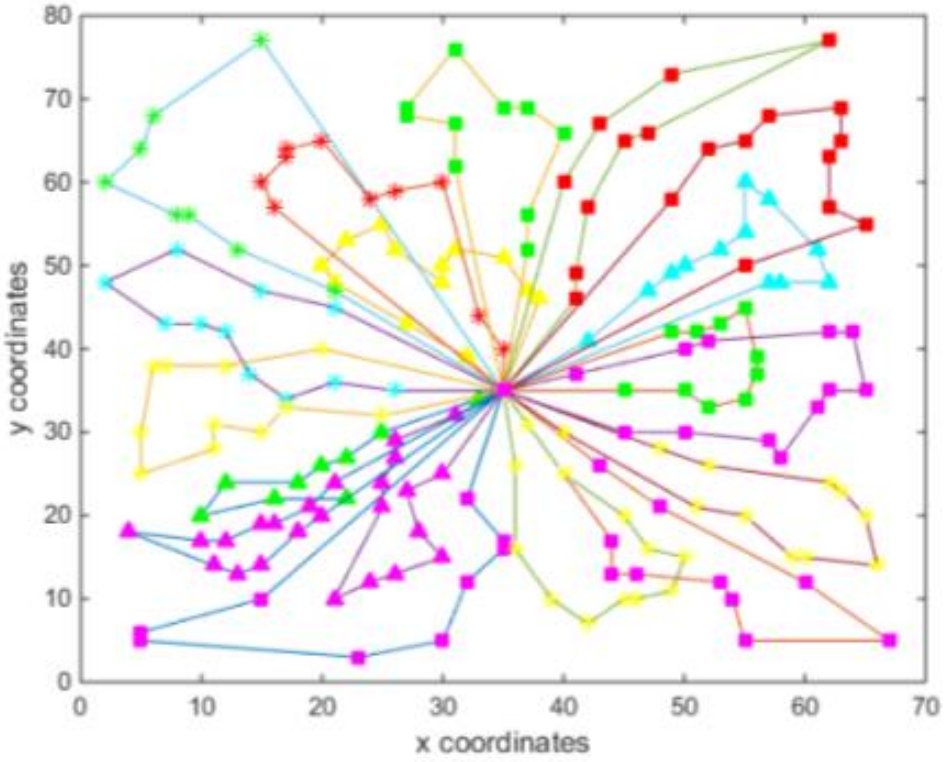
Εικόνα 5.5.1 Βέλτιστη διαδρομή για παράδειγμα par10 με e=20.000 (Βέλτιστη εκτέλεση αλγορίθμου η No 9 με COST=1623,469)



Εικόνα 5.5.2 Βέλτιστη διαδρομή για παράδειγμα par10 με e=200.000 (Βέλτιστη εκτέλεση αλγορίθμου η Νο 10 με COST=1555,62)



Εικόνα 5.5.3 Βέλτιστη διαδρομή για παράδειγμα par10 με e=2.000.000 (Βέλτιστη εκτέλεση αλγορίθμου η No 5 με COST=1469,787)



Στην συγκεκριμένη περίπτωση επιβεβαιώνεται πως όταν αυξάνεται ο αριθμός των επαναλήψεων βελτιώνεται η ποιότητα των αποτελεσμάτων με το καλύτερο αποτέλεσμα να εμφανίζεται στον μέγιστο αριθμό επαναλήψεων (**COST=1469,787**), όπως και ο βέλτιστος μέσος όρος(**M.O=1561,993**).

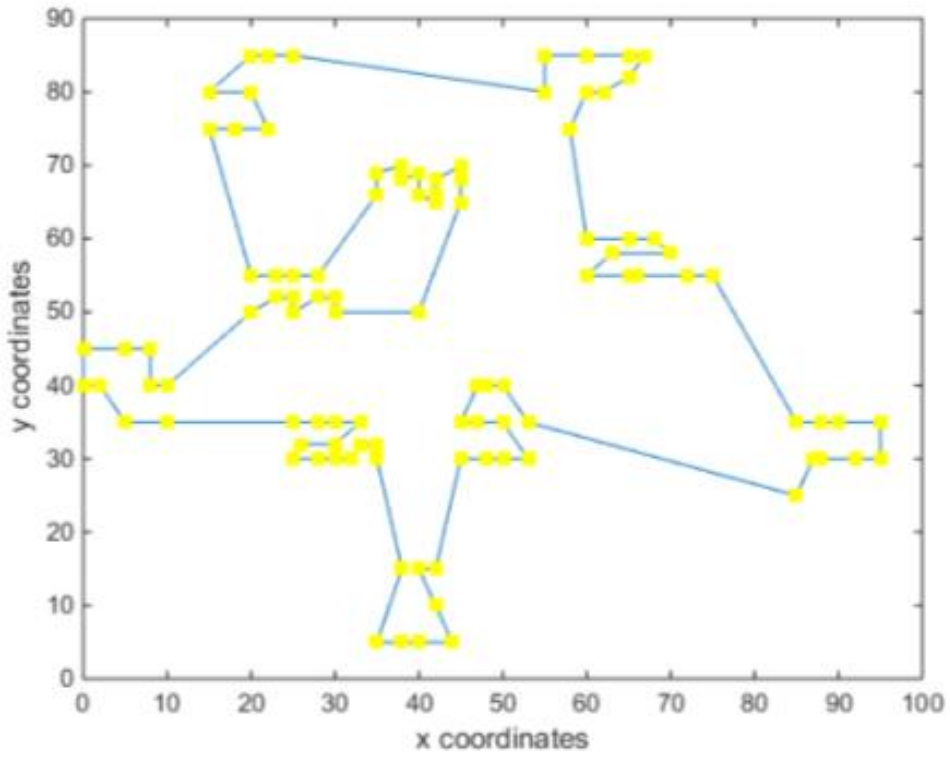
5.6 Παράδειγμα par12 με 101 κόμβους

Στον ακόλουθο πίνακα παρουσιάζονται τα αποτελέσματα για τα 10 διαφορετικά τρεξίματα του αλγορίθμου, για τα 3 είδη επαναλήψεων. Παράδειγμα par12 με 101 κόμβους.

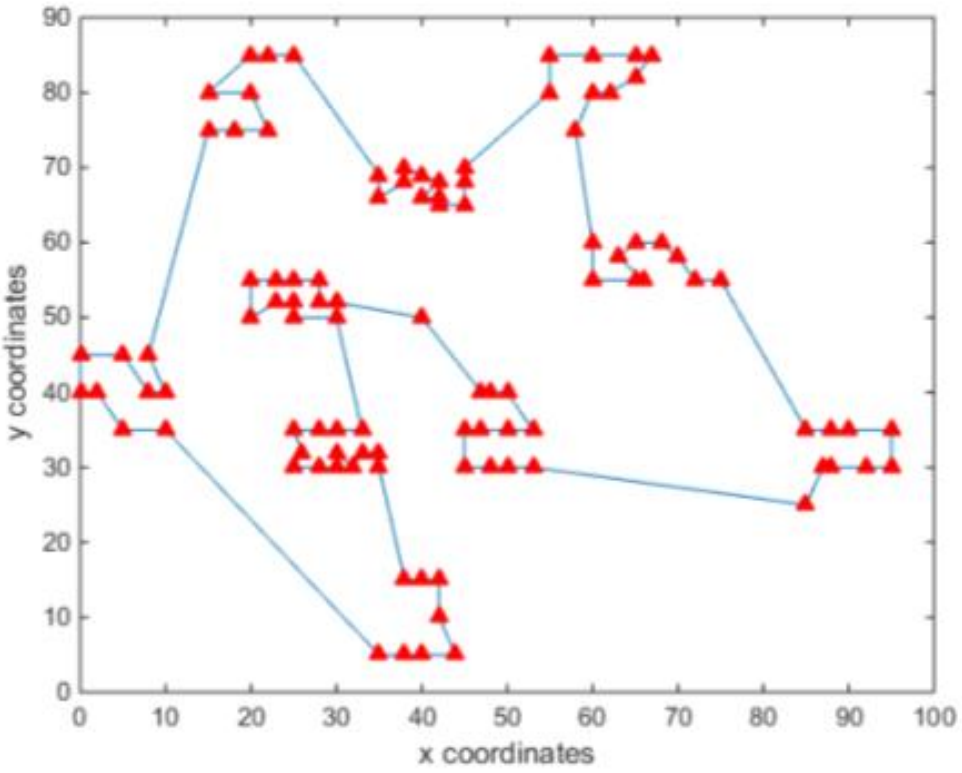
par12	Επαναλήψεις αλγορίθμου		
No τρεξίματος	e=20.000	e=200.000	e=2.000.000
1	553,5368	531,4427	542,3747
2	536,4976	532,1573	537,4262
3	527,3982	527,3758	527,3758
4	535,6751	522,3889	544,6041
5	548,5016	524,5593	521,8113
6	541,6549	542,777	538,9637
7	550,8332	521,928	552,5686
8	538,5894	534,9418	532,7766
9	546,3861	536,6256	544,2056
10	537,1932	551,8378	511,3236
M.O.	***541,6266***	***532,6034***	***535,343***

Ακολουθούν τα γραφήματα των καλύτερων αποτελέσματων ανά κατηγορία επαναλήψεων:

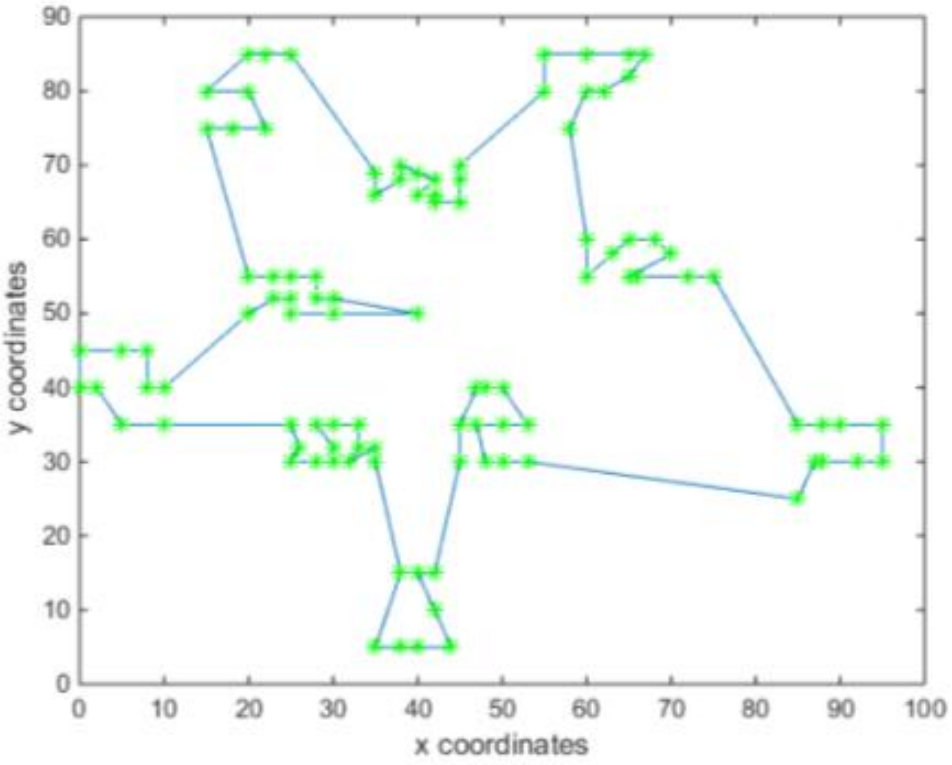
Εικόνα 5.6.1 Βέλτιστη διαδρομή για παράδειγμα par12 με e=20.000 (Βέλτιστη εκτέλεση αλγορίθμου η No 3 με COST=527,3982)



Εικόνα 5.6.2 Βέλτιστη διαδρομή για παράδειγμα par12 με e=200.000 (Βέλτιστη εκτέλεση αλγορίθμου η Νο 7 με COST=521,928)



Εικόνα 5.6.3 Βέλτιστη διαδρομή για παράδειγμα par12 με e=2.000.000 (Βέλτιστη εκτέλεση αλγορίθμου η Νο 10 με COST=511,3236)



Εδώ μπορούμε να παρατηρήσημε ότι το όχημα δεν γυρνάει στην αποθήκη. Αυτό συμβαίνει στο συγκεκριμένο παράδειγμα, διότι ο περιορισμός της απόστασης/χρόνου που δίνεται από τα δεδομένα είναι πολύ μεγάλος.

Σχολιάζοντας τα αποτελέσματα, το τοπικό βέλτιστο έχει εντοπιστεί στον μεγαλύτερο αριθμό επαναλήψεων(2.000.000) και αυτό είναι: **COST=511,3236.** Ο καλύτερος όμως μέσος όρος έχει βρεθεί για τις 200.000 επαναλήψεις (**Μ.Ο.=532,6034**).

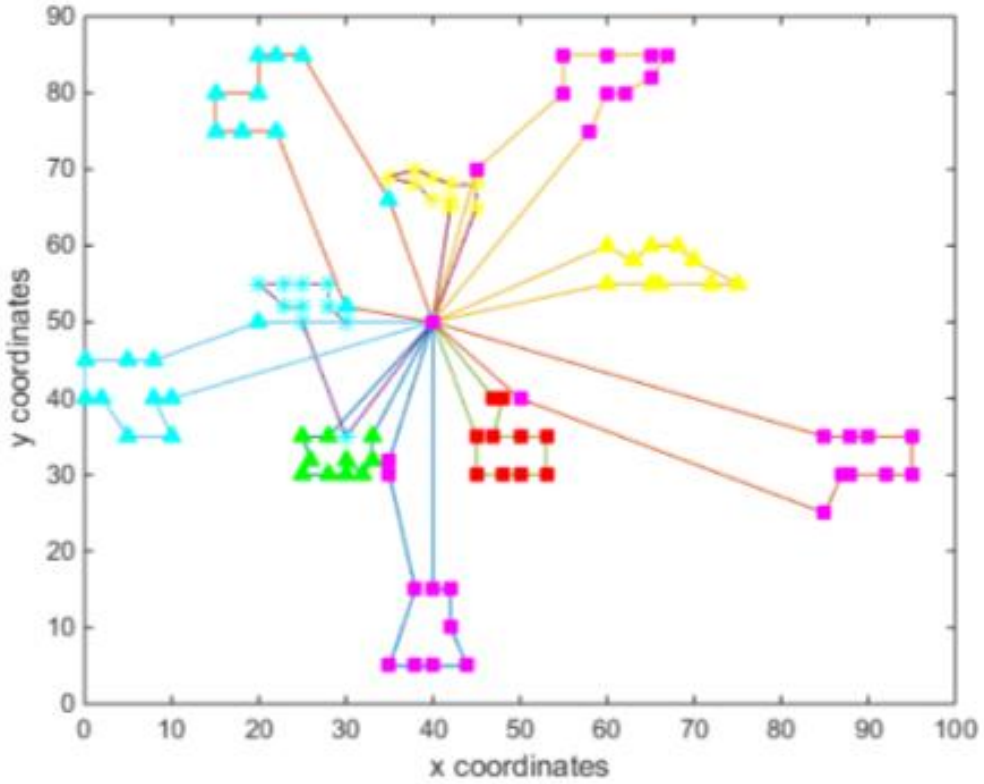
5.7 Παράδειγμα par14 με 101 κόμβους

Για το παράδειγμα par14 με τους 101 κόμβους βρήκαμε τα ακόλουθα αποτελέσματα μετά τη χρήση του αλγορίθμου :

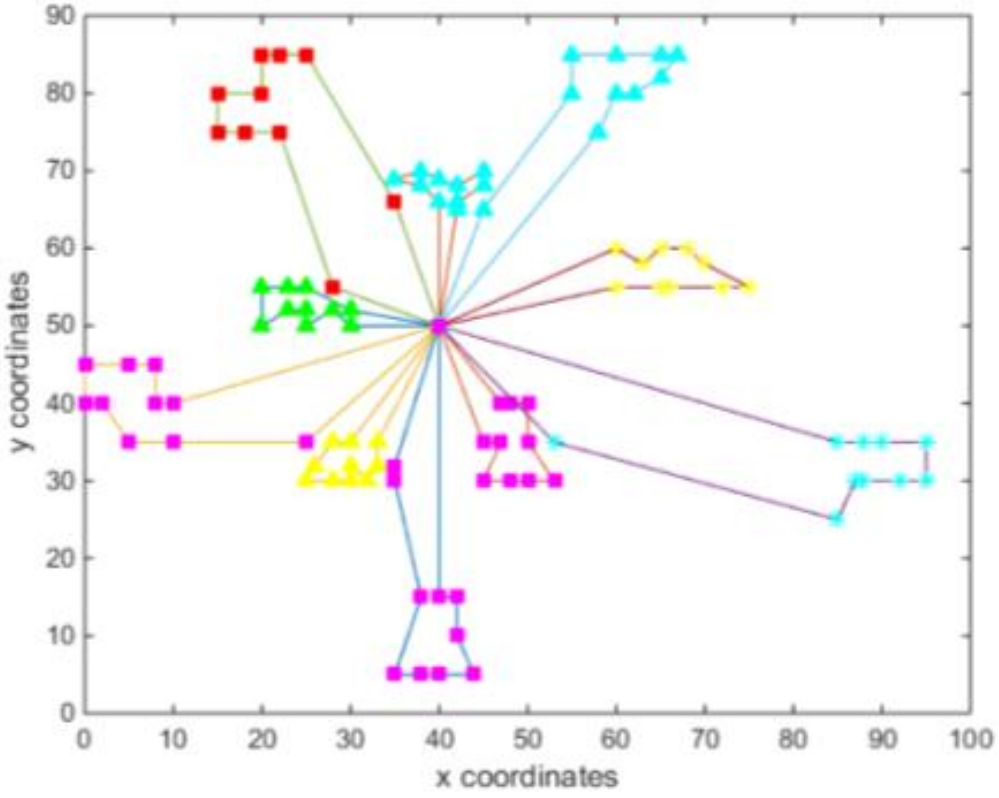
par14	Επαναλήψεις αλγορίθμου		
No τρεξίματος	e=20.000	e=200.000	e=2.000.000
1	931,9133	941,1292	845,1111
2	858,9751	878,6469	997,7942
3	838,5178	863,0747	824,6901
4	857,8878	829,4802	841,2021
5	956,9069	823,8496	825,9334
6	850,2422	854,1501	930,8861
7	902,0121	828,1855	857,6361
8	858,3132	839,3532	825,2904
9	892,3873	856,2943	824,706
10	911,4895	825,7283	926,3784
Μ.Ο.	***885,8645***	***853,9892***	***869,9628***

Ακολουθούν τα γραφήματα των καλύτερων αποτελεσμάτων:

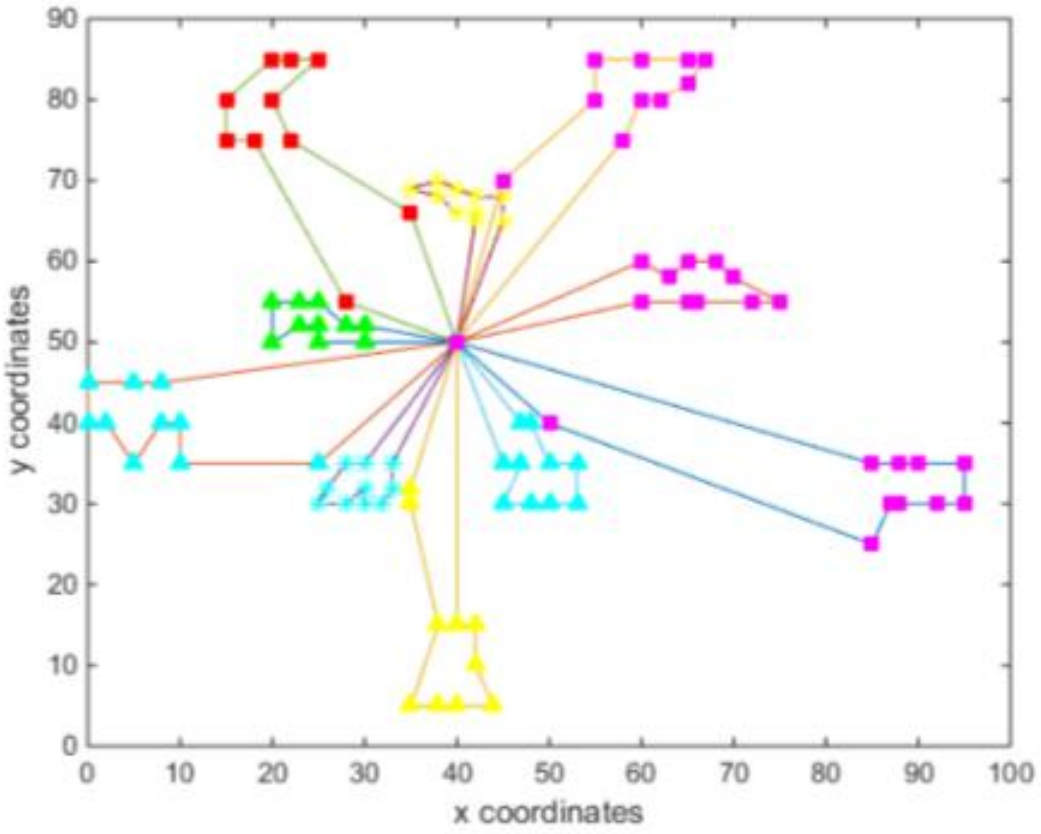
Εικόνα 5.7.1 Βέλτιστη διαδρομή για παράδειγμα par14 με e=20.000 (Βέλτιστη εκτέλεση αλγορίθμου η No 3 με COST=838,5178)



Εικόνα 5.7.2 Βέλτιστη διαδρομή για παράδειγμα par14 με e=200.000 (Βέλτιστη εκτέλεση αλγορίθμου η Νο 5 με COST=823,8496)



Εικόνα 5.7.3 Βέλτιστη διαδρομή για παράδειγμα par14 με e=2.000.000 (Βέλτιστη εκτέλεση αλγορίθμου η Νο 3 με COST=824,6901)



Και σε αυτήν την περίπτωση το τοπικό βέλτιστο βρέθηκε στις 200.000 επαναλήψεις **(COST=823,8496**), όπως επίσης και ο καλύτερος μέσος όρος (**Μ.Ο.=83,9891**).

5.8 Συμπεράσματα

Μετά από τη λύση μιάς ομάδας παραδειγμάτων με τη χρήση του αλγορίθμου της μεταβλητής γειτονιάς αναζήτησης για το πρόβλημα δρομολόγησης οχημάτων σε περιορισμένη απόσταση καταλήξαμε σε κάποια συμπεράσματα.

Ο αλγόριθμος δουλεύει καλύτερα, οπότε εξάγει και καλύτερα αποτελέσματα για τον μέσο αριθμό επαναλήψεων, ο οποίος δεν ξεπερνά τις 200.000. Η τυχαιότητα παίζει μεγάλο ρόλο στην εξαγωγή αποτελεσμάτων. Λόγω της τυχαιότητας, λοιπόν, δε μπορούμε να διασφαλίσουμε ότι οι περισσότερες επαναλήψεις μπορούν να μας δώσουν πάντα το καλύτερο αποτέλεσμα.

Καλύτερο αποτέλεσμα στις 2.000.000 επαναλήψεις βρήκαμε μόνο όταν δουλεύαμε για 200 κόμβους. Φαίνεται, δηλαδή, πως ένας μεγάλος αριθμός επαναλήψεων οδηγεί στο βέλτιστο αποτέλεσμα όταν δουλεύει για ένα μεγάλο πλήθος κόμβων (200 κόμβοι και πάνω). Όμως η εφαρμογή του αλγορίθμου για μεγάλο αριθμό επαναλήψεων σε συνδιασμό με μεγάλο πλήθος κόμβων είναι αρκετά χρονοβόρα.

Για την μελλοντική βελτίωση της ποιότητας του αλγορίθμου Μεταβλητής γειτονιάς αναζήτησης μπορούμε να προτείνουμε κάποιους τρόπους:

- Να χρησιμοποιηθούν επιπλέον αλγόριθμοι τοπικής αναζήτησης σε συνδιασμό με αυτές που χρησιμοποιήσαμε σε αυτήν την εργασία. Όπως 3-opt και 2-0 επανατοποθέτηση (2-0 relocate).
- Μετά από την ολοκλήρωση ενός αριθμού επαναλήψεων να αποθηκεύεται το καλύτερο αποτέλεσμα και στη συνέχεια να χρησιμοποιείται αυτό σαν αρχική λύση για τον αμέσως επόμενο μεγαλύτερο αριθμό επαναλήψεων.

ΒΙΒΛΙΟΓΡΑΦΙΑ

[1] Hansen, P., Mladenovic, N., (2001). Variable Neighborhood Search: Principles and applications*, European Journal of Operational Research*, 130, 3, 449-467

[2] Lambert, D., Cooper, M., (2000). Issues in supply chain Management, *Industrial Marketing Management*, 29, 1, 65-83.

[3] Laptorte, G., Desrochers, M., Nobert, Y., (1984). Two exact algorithms for the distance-constrained vehicle-routing problem*, Networks*, 14, 161-172.

[4] Thomas, D., Griffin, P., (1996). Coordinated supply chain management*, European Journal of Operational Research*, 94, 1-15.

[5] Tlili , T., Faiz, S., Krichen, S., (2014). A Hybrid Metaheuristic for the Distance-constrained Capacitated Vehicle Routing Problem, *Procedia - Social and Behavioral Sciences*, 109, 779–783.

[6] Toth, P., Vigo, D., (2002). Models, relaxations and exact approaches for the capacitated vehicle routing problem, *Discrete Applied Mathematics*, 123, 487 – 512.

[7] Κυριαζόπουλος, Π., (1996) Διοίκηση Logistics, Εκδόσεις "ΣΥΓΧΡΟΝΗ ΕΚΔΟΤΙΚΗ".

[8] Μαρινάκης, Ι., Μαρινάκη, Μ., Ματσατσίνης, Ν., Ζομπουνίδης, Κ., (2011) *Μεθευρετικοί και εξελικτικοί αλγόριθμοι σε προβλήματα διοικητικής επιστήμης*, Εκδόσεις "ΚΛΕΙΔΑΡΙΘΜΟΣ".

[9] Μαρινάκης, Ι., Μυγδαλάς, Α., (2008). *Σχεδιασμός και Βελτιστοποίηση της Εφοδιαστικής Αλυσίδας*, Εκδόσεις "σοφία".

[10] Μαρινάκης, Ι., Μαρινάκη, Μ. (2010) *Εξελικτικοί Αλγόριθμοι και Βελτιστοποίηση Συστημάτων Μεγάλης Κλίμακας*, Σημειώσεις Μαθήματος Εξελικτικοί Αλγόριθμοι και Βελτιστοποίηση Συστημάτων Μεγάλης Κλίμακας.

[11] https://el.wikiversity.org

[12] www.logistics-managment.gr

[13] https://en.wikipedia.org/wiki/NP-hardness