ΠΟΛΥΤΕΧΝΕΙΟ ΚΡΗΤΗΣ

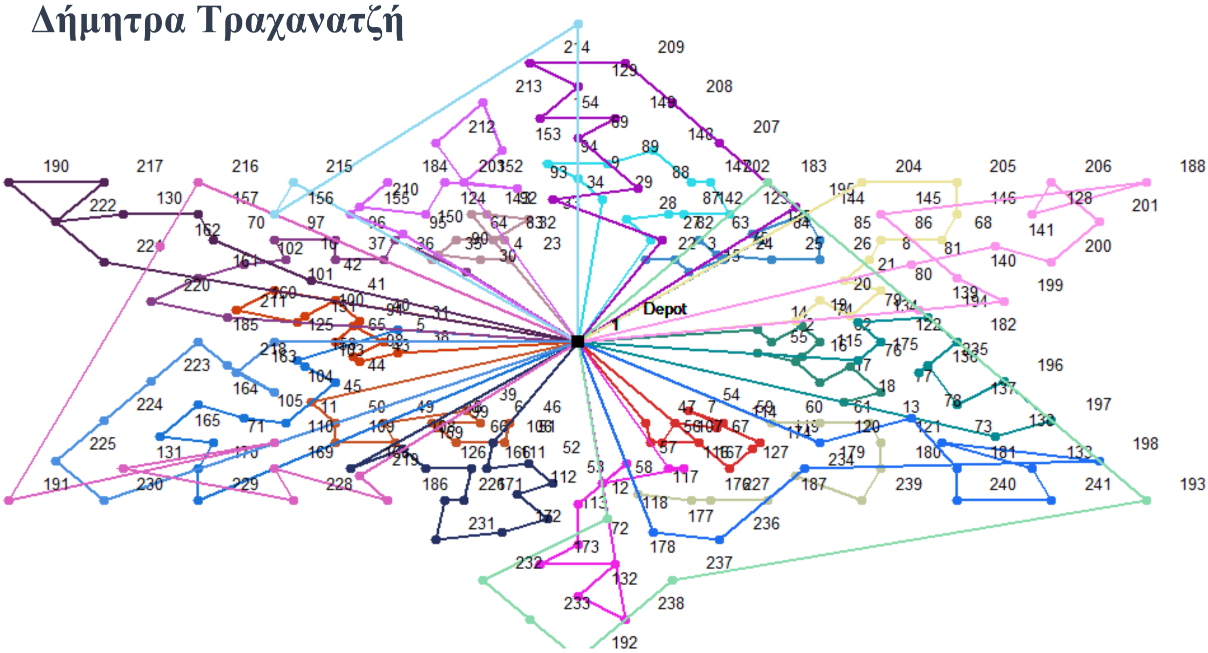
ΣΧΟΛΗ ΜΗΧΑΝΙΚΩΝ ΠΑΡΑΓΩΓΗΣ ΚΑΙ ΔΙΟΙΚΗΣΗΣ

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

ΕΠΙΛΥΣΗ ΠΡΟΒΛΗΜΑΤΩΝ ΔΡΟΜΟΛΟΓΗΣΗΣ ΟΧΗΜΑΤΩΝ ΜΕ ΧΡΗΣΗ ΑΛΓΟΡΙΘΜΟΥ ΑΠΛΗΣΤΗΣ ΤΥΧΑΙΟΠΟΙΗΜΕΝΗΣ ΠΡΟΣΑΡΜΟΣΤΙΚΗΣ ΑΝΑΖΗΤΗΣΗΣ

Solving vehicle routing problems with greedy randomized adaptive search procedure (GRASP)

Δήμητρα Τραχανατζή



Επιβλέπων: Δρ. Μαρινάκης Ιωάννης, Επίκουρος Καθηγητής

Πίνακας περιεχομένων

Χανιά 2015

Ευχαριστίες

Θα ήθελα πρώτα απ' όλα να ευχαριστήσω τον καθηγητή μου κ. Ιωάννη Μαρινάκη, για την καθοδήγηση και τις γνώσεις που μου προσέφερε, όπως επίσης την κα. Μάγδα Μαρινάκη και τον Ηρακλή-Δημήτριο Ψύχα.

Ακόμα νιώθω την ανάγκη να ευχαριστήσω όλους τους φίλους μου, που απετέλεσαν στήριγμά τα τελευταία χρόνια, τόσο σε ακαδημαϊκό όσο και σε προσωπικό επίπεδο. Ιδιαίτερα ευχαριστώ τους Απανωμεριτάκη Χ., Σπανό Ι., Στιβανάκη Χ., Ρηγάκη Μ., Γιακουμάκη Θ.

Τέλος, θέλω να δώσω τις πιο θερμές ευχαριστίες στην οικογένεια μου, τους σημαντικότερους συνοδοιπόρους μου , για την στήριξη τους και τις θυσίες που έχουν κάνει για την μόρφωση του .

Η παρούσα διπλωματική εργασία αφιερώνεται στην οικογένεια μου.

Στον πατέρα μου Μανώλη , στην μητέρα μου Μαρία

και κυρίως στον αδερφό μου Γιώργο,

με ελπίδα να αποτελέσει έμπνευση γι' αυτόν.

Περίληψη

Η μεταφορά και διανομή προϊόντων είναι από τις σημαντικότερες δραστηριότητες της εφοδιαστικής αλυσίδας και συνήθως αντιστοιχούν σε μεγάλο ποσοστό των δαπανών μίας επιχείρησης. Η διαχείριση του στόλου οχημάτων για την μεταφορά και την διανομή αποτελεί κομβικό σημείο. Σήμερα τα καύσιμα είναι ακριβά και οι αστοχίες απαγορεύονται, η ορθή διαχείριση στοχεύει όχι μόνο στην απρόσκοπτη λειτουργία, αλλά και στον περιορισμό του κόστους. Αυτά σε συνδυασμό με την βελτίωση της εξυπηρέτησης των πελατών, εξασφαλίζουν ανταγωνιστικό πλεονέκτημα στις επιχειρήσεις. Για να επιτευχθεί αυτό, απαιτείται βέλτιστη δρομολόγηση οχημάτων, η οποία μπορεί να καθοριστεί από αλγοριθμική επίλυση των αντίστοιχων προβλημάτων δρομολόγησης. Στην παρούσα διπλωματική εργασία επιλύουμε τα εξής προβλήματα δρομολόγησης οχημάτων: το ανοιχτό πρόβλημα δρομολόγησης οχημάτων (Open Vehicle Routing Problem) και το πρόβλημα δρομολόγησης οχημάτων περιορισμένης χωρητικότητας (Capacitated Vehicle Routing Problem). Μέσα από την επίλυση των προβλημάτων εντοπίζουμε τις βέλτιστες διαδρομές οχημάτων που ξεκινάνε από την ίδια αποθήκη, με περιορισμένη χωρητικότητα και εξυπηρετούν πελάτες γεωγραφικά διασκορπισμένους. Κάθε πελάτης έχει ντετερμινιστική ζήτηση και χρόνο εξυπηρέτησης, ενώ δεν μπορεί να εξυπηρετηθεί πάνω από μία φορά. Η συνολική διαδρομή που διανύει το κάθε όχημα δεν επιτρέπεται να ξεπερνάει ένα συγκεκριμένο μήκος, θεωρούμε ότι οι μονάδες μήκους και χρόνου ταυτίζονται. Η διαδρομή ολοκληρώνεται όταν ένα όχημα δεν μπορεί να εξυπηρετήσει άλλο πελάτη. Τέλος στο ανοιχτό πρόβλημα δρομολόγησης τα οχήματα δεν είναι αναγκασμένα να επιστρέψουν στην αποθήκη όταν ολοκληρώσουν την διαδρομή τους, ενώ στο πρόβλημα δρομολόγησης περιορισμένης χωρητικότητας επιστρέφουν στην αποθήκη, από την οποία ξεκίνησαν. Τα προβλήματα επιλύθηκαν στο προγραμματιστικό περιβάλλον Matlab, με την χρήση του μεθευρετικού αλγόριθμου της άπληστης τυχαιοποιημένης προσαρμοστικής αναζήτησης (Greedy Randomized Adaptive Search Procedure/ GRASP). Αυτή η τυχαιοποιημένη τεχνική παρέχει μια εφικτή λύση σε κάθε επανάληψη. Οι επαναλήψεις της διαδικασίας της άπληστης τυχαιοποιημένης προσαρμοστικής αναζήτησης σταματούν όταν κάποιο κριτήριο τερματισμού ικανοποιείται. Το τελικό αποτέλεσμα είναι απλά η καλύτερη λύση που βρέθηκε από όλες τις επαναλήψεις. Κάθε επανάληψη αποτελείται από δύο φάσεις. Στην πρώτη, τη φάση κατασκευής ,μια τυχαιοποιημένη συνάρτηση απληστίας χρησιμοποιείται για να κατασκευαστεί μια αρχική λύση. Αυτή η αρχική λύση στη συνέχεια βελτιώνεται στη δεύτερη φάση, με τη χρήση της διαδικασίας τοπικής αναζήτησης.

Κεφάλαιο 1: Εισαγωγή

1.1 Διαχείριση Εφοδιαστικής Αλυσίδας

Η όξυνση του ανταγωνισμού τα τελευταία χρόνια σε συνδυασμό με τη συνεχώς αυξανόμενη πολυπλοκότητα της οικονομικής και εμπορικής παγκοσμιοποίησης, επέτειναν την ανάγκη των επιχειρήσεων για εφαρμογή μίας πετυχημένης εφοδιαστικής διαχείρισης. Με στόχο αφενός τη μείωση του λειτουργικού κόστους και αφετέρου δε τη βελτίωση του επιπέδου εξυπηρέτησης των πελατών τους, ώστε να αποκτήσουν ανταγωνιστικό πλεονέκτημα.

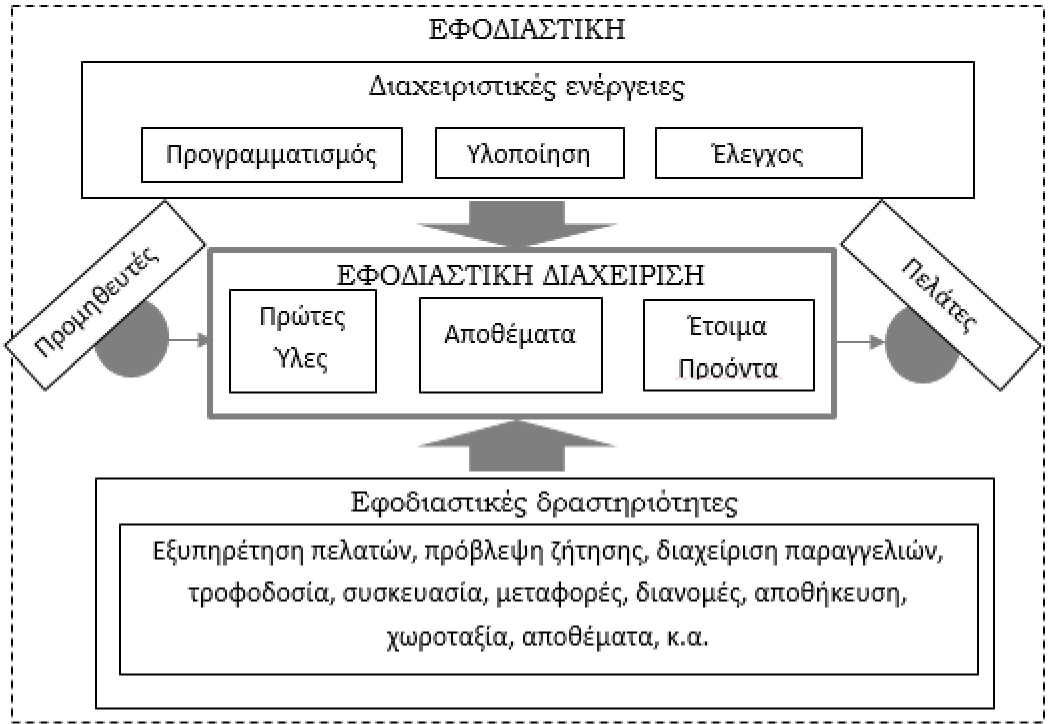
Μια εφοδιαστική αλυσίδα είναι ένα σύνολο από οργανώσεις που συνδέονται άμεσα με μία ή περισσότερες ανοδικές (upstream) και καθοδικές (downstream) ροές προϊόντων, υπηρεσιών, χρηματοοικονομικών υπηρεσιών και πληροφοριών από μια πηγή σε έναν πελάτη. Η διαχείριση της εφοδιαστικής αλυσίδας (Supply Chain Management) περιλαμβάνει το σχεδιασμό και τη διαχείριση όλων των δραστηριοτήτων που εμπλέκονται στην προμήθευση, τη μετατροπή και τον έλεγχο της εφοδιαστικής αλυσίδας, όπως ορίζεται από το Council of Supply Chain Management Professionals [1]. Περιλαμβάνει επίσης τις βασικές συνιστώσες του συντονισμού και της συνεργασίας με εταιρικά κανάλια, τα οποία μπορεί να είναι οι προμηθευτές, οι μεσάζοντες, οι τρίτοι πάροχοι υπηρεσιών και οι πελάτες. Επί της ουσίας, η διαχείριση της εφοδιαστικής αλυσίδας ενσωματώνει την διαχείριση της προσφοράς και της ζήτησης εντός και μεταξύ των εταιρειών. Πρωταρχική ευθύνη της είναι η σύνδεση των σημαντικών επιχειρηματικών λειτουργιών και των επιχειρηματικών διαδικασιών, σε ένα συνεκτικό και υψηλής απόδοσης επιχειρησιακό μοντέλο.

1.2 Εφοδιαστική και Εφοδιαστική Διαχείριση

Η εφοδιαστική (Logistics) είναι η 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 την συμμόρφωση με τις απαιτήσεις των πελάτων. Η εφοδιαστική διαχείριση έχει παίξει σημαντικό ρόλο στην παγκόσμια εξέλιξη, περίπου από το 2700 π.Χ., όταν οι Αιγύπτιοι κατασκεύασαν πυραμίδες και ο σωστός σχεδιασμός για την μεταφορά και την τοποθέτηση των ογκωδών υλικών που χρησιμοποιούσαν [2]. Ιστορικά η εφοδιαστική διαχείριση διαδραμάτιζε σημαντικό ρόλο και σε περιόδους πολέμου.

Η στρατιωτική εφοδιαστική (military logistics) ανάγεται στην εποχή των ρωμαϊκών χρόνων κατά την οποία ήταν απαραίτητη η ταχεία ανάπτυξη λεγεώνων και μεταφορά όπλων και αποθεμάτων, ακόμα στον δεύτερο παγκόσμιο πόλεμο αποτέλεσε σημείο κλειδί για την αποτελεσματική μεταφορά πυρομαχικών και εξοπλισμού.

Η εφοδιαστική διαχείριση (Logistics Management) είναι το κομμάτι της διαχείρισης εφοδιαστικής αλυσίδας (Supply Chain Management) που σχεδιάζει, υλοποιεί και ελέγχει την αποδοτική, αποτελεσματική (ευθεία και αντίστροφη) ροή και αποθήκευση των αγαθών, των υπηρεσιών και των σχετικών πληροφοριών μεταξύ του σημείου προέλευσης και του σημείου κατανάλωσης, προκειμένου να ανταποκριθεί στις απαιτήσεις των πελατών [3]. Οι δραστηριότητες διαχείρισης της εφοδιαστικής συνήθως περιλαμβάνουν την διαχείριση εισερχόμενων και εξερχόμενων μεταφορών, τη διαχείριση του στόλου, την αποθήκευση και διαχείριση υλικών, την εκπλήρωση παραγγελιών, τον σχεδιασμό του δικτύου εφοδιασμού, τον σχεδιασμό προσφοράς/ζήτησης και διαχείριση υπηρεσιών από τρίτους παρόχους. Η εφοδιαστική διαχείριση είναι μία ολοκληρωμένη λειτουργία που συντονίζει και βελτιστοποιεί όλες τις διαδικασίες της εφοδιαστικής και ενσωματώνει επιπρόσθετες λειτουργίες όπως το μάρκετινγκ, τις πωλήσεις, την κατασκευή, την χρηματοδότηση και τις πληροφοριακές τεχνολογίες.



1.3 Μεταφορά και Διανομές

Η δραστηριότητα της μεταφοράς και διανομής αφορά τον τρόπο και τα μέσα για την μεταβίβαση υλικών και αγαθών ή υπηρεσιών μέσω των φυσικών καναλιών. Για να ολοκληρωθεί η μεταφορά πρέπει να γίνει επιλογή του μέσου, του τύπου οχήματος, ο καθορισμός των διαδρομών και των δρομολογήσεων, η επιλογή των μεταφορέων και αλλά. Τα έξοδα των μεταφορών αποτελούν συνήθως το μεγαλύτερο κομμάτι της πίτας των εξόδων της εφοδιαστικής αλυσίδας, άρα απορροφούν και μεγάλο κόστος από την επιχείρηση. Η εφοδιαστική διανομή (Distribution logistics) είναι εκείνος ο τομέας της διαχείρισης της εφοδιαστικής αλυσίδας ο οποίος είναι υπεύθυνος για την παράδοση των προϊόντων στους πελάτες στον σωστό τόπο, στην σωστή χρονική στιγμή, σε σωστή κατάσταση και για το σωστό κόστος.

Οι μεταφορές χωρίζονται σε δύο μέρη:

i. Σε εσωτερικές, που περιλαμβάνουν τόσο μεταφορά πρώτων υλών από τις πηγές προς τα εργοστάσια όσο και μερών των τελικών προϊόντων ανάμεσα σε διάφορα της εταιρίας ή από το εργοστάσιο στην αποθήκη και στα σημεία πωλήσεων (Inbound logistics)
ii. Σε εξωτερικές μεταφορές που περιλαμβάνουν μεταφορά των τελικών προϊόντων από τις αποθήκες στους πελάτες άμεσα ή διαμέσου κέντρων διανομής (Outbound logistics).

Προβλήματα στον σχεδιασμό των μεταφορών περιλαμβάνουν την επιλογή του στόλου μεταφοράς (μέγεθος του στόλου και χρήση διαφορετικών τύπων οχημάτων), τη δρομολόγηση οχημάτων (επιλογή των βέλτιστων διαδρομών, λαμβάνοντας υπόψη τη δομή του δικτύου, τις αποστάσεις και τη χωρητικότητα των διαδρομών), το σχεδιασμό του δικτύου διανομής (βελτίωση δρομολογίων, του χρονοπρογραμματισμού των δρομολογίων και επιλογή διαφόρων ενδιάμεσων αποθηκών) και την επιλογή του προσωπικού που θα πραγματοποιήσει τις διανομές (καθορισμός απαιτήσεων προσωπικού). Μερικά από τα λειτουργικά προβλήματα που περιλαμβάνονται στον όρο μεταφορά είναι το πρόβλημα του καθορισμού και του ελέγχου της διαδικασίας αποστολής των προϊόντων, όπως επίσης και ο χρονοπρογραμματισμός των πληρωμάτων και των οχημάτων που θα πραγματοποιήσουν τις μεταφορές.

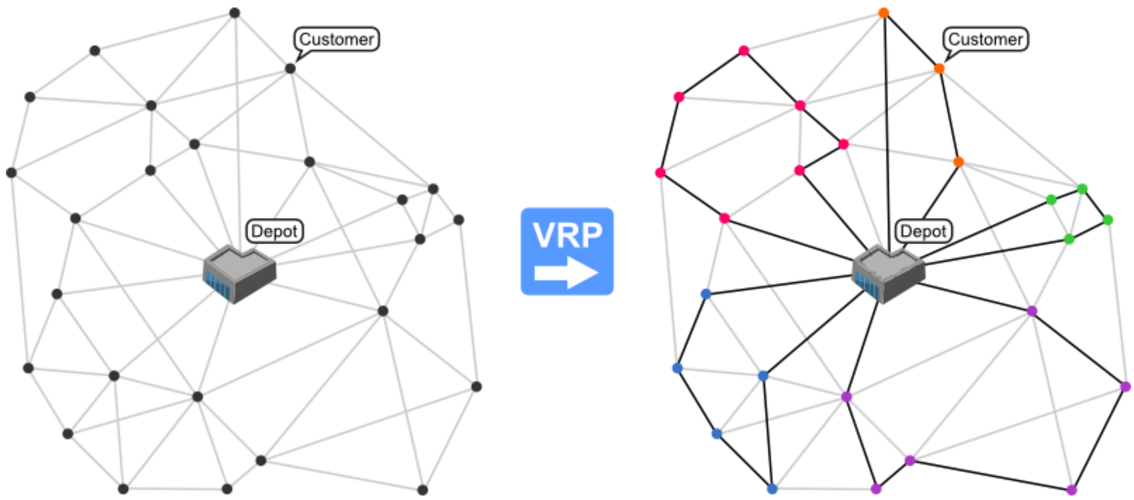
Ένα σύστημα μεταφοράς μπορεί να αναπαρασταθεί με την μορφή δικτύου κόμβων και τόξων, όπου οι κόμβοι τυπικά αντιπροσωπεύουν πόλις, αεροδρόμια, στάσεις και αποθήκες και τα τόξα αντιπροσωπεύουν τους συνδέσμους ή τις διαδρομές μεταξύ των κόμβων. Οι κόμβοι και τα τόξα μπορεί να έχουν περιορισμούς χωρητικότητας. Διαφορετικοί τρόποι είναι συνήθως διαθέσιμοι για την μεταφορά των φορτίων μεταξύ δύο σημείων. Υπάρχουν πέντε βασικοί τρόποι μεταφοράς, οι σιδηροδρομικοί, οδικοί, θαλάσσιοι, αεροπορικοί και αγωγοί μεταφορών.

Κεφάλαιο 2: Δρομολόγηση οχημάτων

2.1 Πρόβλημα Δρομολόγησης Οχημάτων (VRP)

Το Πρόβλημα Δρομολόγησης Οχημάτων (Vehicle Routing Problem) είναι ένα από τα πιο μελετημένα υπολογιστικά προβλήματα βελτιστοποίησης, το πρόβλημα εξετάζει την βέλτιστη δρομολόγηση οχημάτων για την παράδοση ή/και παραλαβή προϊόντων στους/από τους πελάτες, σε δεδομένη χρονική περίοδο, στα πλαίσια επιχειρησιακών δραστηριοτήτων.
Η θεωρητική έρευνα και πρακτική εφαρμογή ξεκίνησε το 1959 από τους Dantzig και Ramser [4] με το πρόβλημα «αποστολής φορτηγών», χρησιμοποιώντας μέθοδο βασισμένη σε γραμμικό προγραμματισμό και υπολογισμούς με το χέρι κατέληξαν σε μία σχεδόν βέλτιστη λύση.

Η σωστή επίλυση ενός προβλήματος δρομολόγησης οχημάτων έχει σαν αποτέλεσμα τον καθορισμό ενός συνόλου από διαδρομές, με ελάχιστο ολικό κόστος, με σκοπό την εξυπηρέτηση ενός πλήθους πελατών γεωγραφικά διασκορπισμένων, χωρίς να παραβιαστεί κάποιος περιορισμός. Τα οχήματα ξεκινάνε από μία ή περισσότερες αποθήκες και έχουν την δυνατότητα να επιστρέψουν σε αυτές. Υπάρχουν πολλές μεταβλητές, οι οποίες δημιουργούν παραλλαγές στο αρχικό πρόβλημα. Όπως για παράδειγμα ο χρόνος εξυπηρέτησης ή η προτεραιότητα των πελατών [5].



Σχήμα 2.1: Πρόβλημα δρομολόγησης οχημάτων (Πηγή: neo.lcc.uma.es)

Όπως φαίνεται και στο παραπάνω σχήμα, το οδικό δίκτυο που χρησιμοποιούν τα οχήματα περιγράφεται γενικά από ένα γράφημα, του οποίου τα τόξα (arcs) αντιπροσωπεύουν τα τμήματα δρόμου του δικτύου και οι κόμβοι vertices) τις αποθήκες και τις θέσεις των πελατών. Τα τόξα μπορεί να είναι προσανατολισμένα ή όχι ανάλογα αν ο δρόμος είναι διπλής κατεύθυνσης ή όχι (directed or undirected). Κάθε τόξο συσχετίζεται με ένα κόστος, που αντιστοιχεί συνήθως στο μήκος του και στο χρόνο που απαιτείται για να το διασχίσει κάποιο όχημα. Προκειμένου να προσδιοριστεί ένα πρόβλημα δρομολόγησης απαιτούνται πληροφορίες σχετικά με τους πελάτες και τα οχήματα που διατίθενται [6]. Πιο αναλυτικά:

Χαρακτηριστικά των πελατών:

- Τη γεωγραφική θέση του πελάτη, άρα το σημείο του γραφήματος στο οποίο βρίσκεται (road graph).
- Την ποσότητα των αγαθών (demand) τα οποία πρέπει είτε να παραδοθούν, είτε να συλλεχθούν από τον πελάτη.
- Τα χρονικά διαστήματα (time windows) κατά τη διάρκεια της ημέρας, μέσα στα οποία ο πελάτης μπορεί να εξυπηρετηθεί.
- Χρόνος που απαιτείται για την εξυπηρέτηση κάθε πελάτη (service time).Η παράμετρος μπορεί να αφορά χρόνο φορτοεκφόρτωσης (unloading or loading time).

Χαρακτηριστικά οχημάτων:

- Από ποια αποθήκη προέρχονται και αν υπάρχει η δυνατότητα να τερματίσουν σε άλλη αποθήκη από αυτή που ξεκίνησαν ή να μην επιστρέψουν κάπου.
- Η χωρητικότητα του οχήματος.
- Αν τα οχήματα είναι χωρισμένα σε τμήματα και πώς αυτά θα φορτωθούν.
- Αν τα οχήματα διαφέρουν μεταξύ τους, ανάλογα με την χωρητικότητα και το είδος των αγαθών που μεταφέρουν.
- Το σύνολο των δρόμων που είναι προσπελάσιμοι από το όχημα.
- Το κόστος που συσχετίζεται με την λειτουργία του κάθε οχήματος.

Πρέπει να αναφερθεί ότι οι οδηγοί των οχημάτων πρέπει να ακολουθούν κανόνες που θεσπίζονται και δημιουργούν αντίστοιχους περιορισμούς στο πρόβλημα. Ενδεικτικά: οκτώ ώρες ύπνου υποχρεωτικά, όχι περισσότερες από δέκα ώρες συνεχόμενης οδήγησης, έξι ημέρες εβδομαδιαίως και δεκαπέντε ώρες ημερησίως.

Ανάλογα με το πλήθος και το είδος των περιορισμών, τροποποιείται και το πρόβλημα δρομολόγησης, οι ποιο βασικοί που συναντάμε στην βιβλιογραφία είναι:

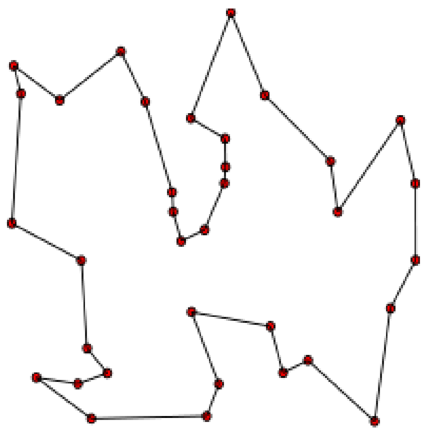
- Σε κάθε ξεχωριστή διαδρομή η ποσότητα που μεταφέρει το όχημα να μην ξεπερνάει την συνολική του χωρητικότητα.
- Η εξυπηρέτηση να πραγματοποιείται σε συγκεκριμένη χρονική περίοδο.

- Οι πελάτες να διαφέρουν ως προς το είδος, κάποιοι να ζητούν μόνο παραλαβή ή μόνο διανομή.
- Οι πελάτες να πρέπει να εξυπηρετηθούν με συγκεκριμένη σειρά.

Οι στόχοι της επίλυσης προβλημάτων δρομολόγησης οχημάτων είναι οι εξής :

1. Η ελαχιστοποίηση του συνολικού κόστος μεταφοράς των προϊόντων. Το συνολικό κόστος εξαρτάται από τη συνολική απόσταση ή τον συνολικό χρόνο που απαιτείται και από το πάγιο κόστος ανά όχημα ή/και οδηγό.
2. Η ελαχιστοποίηση του αριθμού των οχημάτων .
3. Η ισορροπία μεταξύ των διαδρομών σε σχέση με τις απαιτούμενες ώρες που χρειάζονται.
4. Η ελαχιστοποίηση των ποινών που πηγάζουν από την μερική εξυπηρέτηση πελατών.

2.1.1 Το Πρόβλημα του Πλανόδιου Πωλητή (TSP)

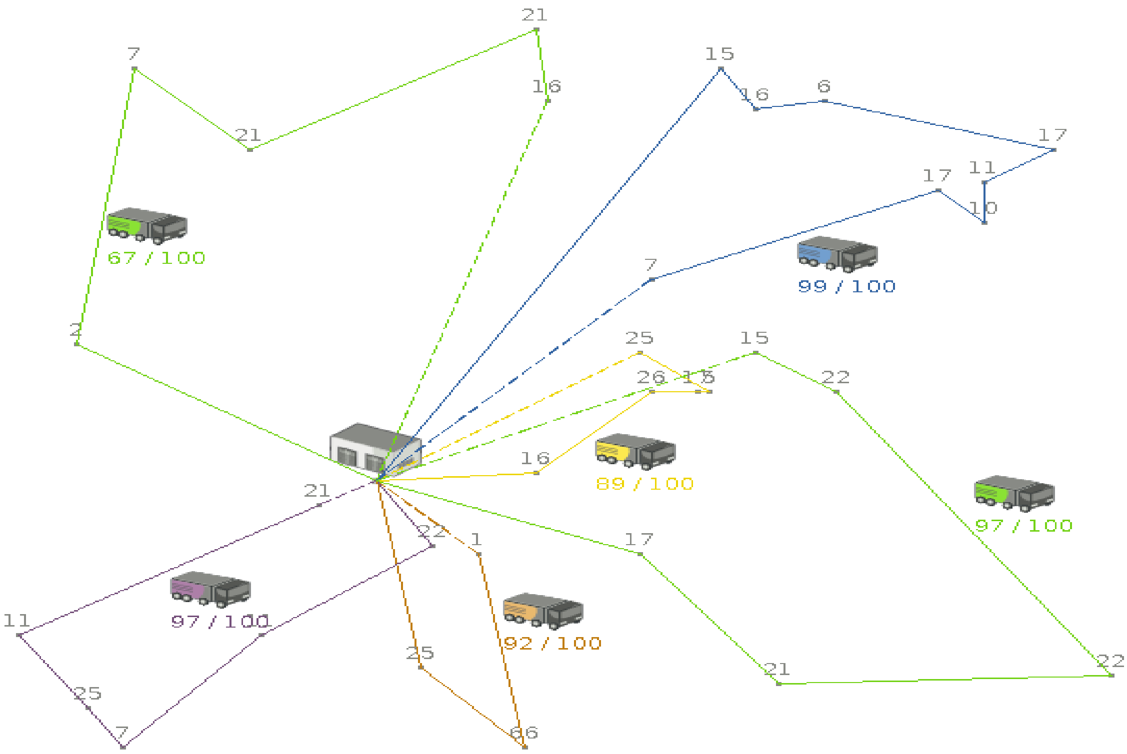


Το πρόβλημα του πλανόδιου πωλητή (traveling salesman problem-TSP) αφορά την εύρεση της συντομότερης (σε χρόνο ή απόσταση) διαδρομής για ένα όχημα με αφετηρία ένα σημείο και επιστροφή στο ίδιο, αφού πρώτα έχει επισκεφτεί έναν σταθερό αριθμό πελατών ακριβώς μία φορά τον καθένα [7].

Σχήμα2.2: Λύση TSP (Πηγή: en.wikipedia.org)

2.2 Το Περιορισμένης Χωρητικότητας Πρόβλημα Δρομολόγησης Οχημάτων (CVRP)

Το πρόβλημα της δρομολόγησης οχημάτων περιορισμένης χωρητικότητας (Capacitated Vehicle Routing Problem) είναι η πιο βασική εκδοχή του προβλήματος δρομολόγησης οχημάτων (Vehicle Routing Problem). Στο CVRP όλοι οι πελάτες δέχονται την παράδοση των προϊόντων, η ζήτηση τους είναι ντετερμινιστική και γνωστή εκ των προτέρων, ενώ δεν μπορεί να διασπαστεί. Τα οχήματα είναι ίδια (και με ίδιο επίπεδο χωρητικότητας) και ξεκινάνε από μία κεντρική αποθήκη και επιστρέφουν σε αυτή. Περιορισμός του προβλήματος αποτελεί το επίπεδο της ζήτησης σε κάθε όχημα. Στόχος είναι η ελαχιστοποίηση του κόστους για την εξυπηρέτηση όλων των πελατών [6]. Ουσιαστικά το πρόβλημα εκφράζεται ως επέκταση του προβλήματος του πλανόδιου πωλητή (TSP) στην περίπτωση που ένα όχημα δεν δύναται να εξυπηρετήσει όλους τους πελάτες. Έτσι δημιουργούνται κυκλικές διαδρομές με ίδια αρχή και τέλος (την αποθήκη), όμως ο κάθε πελάτης θα καλύπτεται από μια και μόνο διαδρομή.



Σχήμα 2.3: Παράδειγμα CVRP με 32 πελάτες και 6 οχήματα, χωρητικότητας 100 μονάδες. (Πηγή: docs.jboss.org)

Στη συνέχεια περιγράφουμε το πρόβλημα περιορισμένης χωρητικότητας δρομολόγησης οχημάτων με όρους θεωρίας γραφημάτων[6]. Έστω ένα πλήρης γράφημα $G=(V,A)$, όπου $V= \{0,\ldots\ldots, n\}$ είναι το σύνολο των κόμβων και A το σύνολο των τόξων. Η αποθήκη αντιστοιχείται στον κόμβο 0 ή σε πολλές περιπτώσεις στον n+1. Ένα μη αρνητικό κόστος c_{ij} συσχετίζεται με κάθε τόξο (i, j) και αντιπροσωπεύει το κόστος ταξιδιού από τον κόμβο i στον κόμβο j. Γενικά τόξα επανάληψης (i,i) δεν επιτρέπονται και αυτό διασφαλίζεται θέτοντας το αντίστοιχο κόστος ίσο με άπειρο. Όταν το γράφημα είναι προσανατολισμένο, ο πίνακας με τα κόστη είναι μη συμμετρικός και το πρόβλημα ονομάζεται: Μη συμμετρικό CVRP (Asymmetric CVRP / ACVRP). Στην αντίθετη περίπτωση ισχύει ότι: $c_{ij}= c_{ji}$ για κάθε i, j που ανήκουν στο Α και το πρόβλημα ονομάζεται : συμμετρικό CVRP (Symmetric CVRP / SCVRP). Στις περισσότερες πρακτικές εφαρμογές οι κόμβοι συσχετίζονται με σημεία στο επίπεδο, δηλαδή τα δεδομένα είναι συντεταγμένες και κάθε κόστος c_{ij} υπολογίζεται από την Ευκλείδεια απόσταση ανάμεσα στα δυο σημεία που αντιστοιχούν στους κόμβους i, j. Ακολουθεί μία βασική μορφοποίηση για την επίλυση του προβλήματος δρομολόγησης οχημάτων περιορισμένης χωρητικότητας, που βασίζεται στην ροή των οχημάτων, έτσι όπως έχει παρουσιαστεί από τους Fisher και Jaikumar [8]. Έστω:

$$\mathrm{x}_{\mathrm{ijk}} = \begin{cases} 1, & \text{εάν το όχημα } k \text{ επισκέπτεται τον πελάτη } j \\ & \text{αμέσως μετα τον πελάτη } i \\ 0, & \text{αλλιώς} \end{cases}$$

$$y_{ik} = \begin{cases} 1, & \text{εάν ο πελάτης } i \text{ επισκέπτεται από το όχημα } k \\ 0, & \text{αλλιώς} \end{cases}$$

Ελαχιστοποιείται η αντικειμενική συνάρτηση κόστους:

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

Υπό:

$$\sum_{k} y_{ik} = \begin{cases} 1, & i = 2, \dots, n, \\ m, & i = 1 \end{cases} \qquad (2.1)$$

$$\sum_{i} q_i \, y_{ij} \ \le \ Q_k \qquad k = 1, \dots, m \qquad (2.2)$$

$$\sum_{j} x_{ijk} = \sum_{j} x_{jik} = y_{ik} \qquad i = 1, \dots, n \ \& \ k = 1, \dots, m \qquad (2.3)$$

$$\sum_{i,j \in S} x_{ijk} \ \le |S| - 1 \quad \text{για όλα τα } S \subseteq \{2, \dots, n\} \ \& \ k = 1, \dots, m \quad (2.4)$$

$$y_{ik} \ \in \{0,1\} \qquad i = 1, \dots, n \ \& \ k = 1, \dots, m \qquad (2.5)$$

$$x_{ijk} \ \in \{0,1\} \qquad i,j = 1, \dots, n \ \& \ k = 1, \dots, m \qquad (2.6)$$

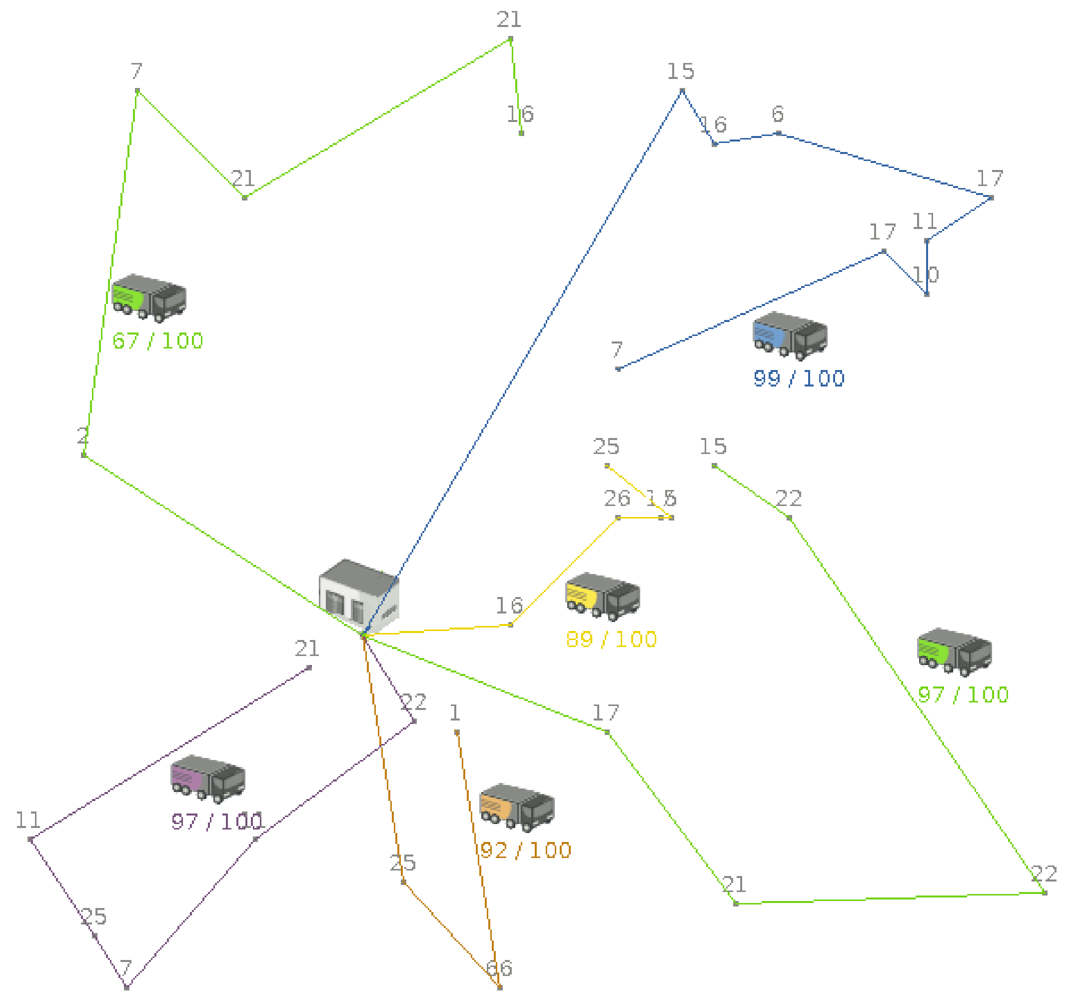
Ο περιορισμός (2.1) δείχνει ότι κάθε πελάτης εκχωρείται σε ένα μόνο όχημα, εκτός από την αποθήκη που την επισκέπτονται όλα τα οχήματα. Ο περιορισμός (2.2) αφορά την χωρητικότητα των οχημάτων. Ο (2.3) ότι ένα όχημα που επισκέπτεται έναν πελάτη φεύγει από αυτόν.

2.2.1 Το Πρόβλημα Δρομολόγησης Οχημάτων Περιορισμένης Χωρητικότητας και Απόστασης.

Τροποποιώντας το βασικό πρόβλημα της περιορισμένης χωρητικότητας δρομολόγησης οχημάτων καταλήγουμε στο λεγόμενο πρόβλημα δρομολόγησης οχημάτων περιορισμένης χωρητικότητας και απόστασης (Distance-Constrained Capacitated Vehicle Routing Problem / DCVRP).Η νέα μορφή προβλήματος προκύπτει όταν προσθέσουμε περιορισμό στην συνολική απόσταση(ή χρόνο) που διανύει κάθε όχημα (maximum tour length). Συνήθως σε αυτό το πρόβλημα το κόστος μίας διαδρομής ισούται με τον χρόνο που απαιτείται για να ολοκληρωθεί. Στόχος του DCVRP είναι να ελαχιστοποιήσει το συνολικό μήκος κάθε διαδρομής (ή την διάρκεια της), όταν συμπεριλαμβάνει και τους χρόνους εξυπηρέτησης των πελατών που καλύπτει.

2.3 Το Ανοιχτό Πρόβλημα Δρομολόγησης Οχημάτων (OVRP)

Στο βασικό πρόβλημα δρομολόγησης οχημάτων (VRP) δημιουργούμε μία σειρά από διαδρομές, με τις οποίες ένας ομογενής στόλος οχημάτων, χρησιμοποιώντας μία και μόνο αποθήκη εξυπηρετεί όλους τους πελάτες και στόχος μας είναι η μείωση της συνολικής δανεισθείσας απόστασης. Κάθε όχημα πρέπει να ξεκινάει και να γυρίζει στην αποθήκη και έχει μια δεδομένη χωρητικότητα και δεδομένο μέγιστο μήκος διαδρομής που μπορεί να ταξιδέψει. Στο ανοιχτό πρόβλημα δρομολόγησης οχημάτων (Open Vehicle Routing Problem/ OVRP) τα οχήματα δεν χρειάζεται να επιστρέψουν στην αποθήκη έπειτα από την εξυπηρέτηση και του τελευταίου πελάτη στη διαδρομή τους . Επιπρόσθετα πρέπει να υπολογίσουμε τον ελάχιστο αριθμό οχημάτων που χρειαζόμαστε για καλύψουν όλο το σύνολο των πελατών. Αυτό σημαίνει ότι το κόστος από την προσθήκη ενός επιπλέον οχήματος είναι μεγαλύτερο από την εξοικονόμηση που πιθανόν έχει γίνει λόγω μείωση του μήκους διαδρομής Η βασική διαφορά με το VRP είναι ότι κάθε διαδρομή αποτελεί ένα μονοπάτι (και όχι κύκλος)το οποίο περνάει από κάθε κόμβο που περιέχει μόνο μια φορά (Hamiltonian path). [9].



Σχήμα 2.3: Παράδειγμα OVRP με 32 πελάτες και 6 οχήματα, χωρητικότητας 100 μονάδες. (Πηγή: docs.jboss.org)

Το ανοιχτό πρόβλημα δρομολόγησης οχημάτων χρησιμοποιείται συνήθως από μία εταιρία που δεν διαθέτει τον δικό της στόλο οχημάτων ή δεν διαθέτει τους κατάλληλους πόρους για να ικανοποιήσει τους πελάτες της. Έτσι εξαναγκάζεται στην μίσθωση στόλου ή την παραχώρηση μέρος της διανομής σε εξωτερικούς παράγοντες, οι οποίοι αναλαμβάνουν και το κόστος ανά όχημα. Παραδείγματα εφαρμογής είναι και η δρομολόγηση τρένων τα οποία ξεκινάνε ή καταλήγουν σε ένα σταθμό [10]. Παρουσιάζεται στη συνέχεια η μαθηματική διατύπωση του προβλήματος. Έστω n το πλήθος των πελατών και $V = \{0,\ldots,n\}$ το σύνολο των πελατών και της αποθήκης. Η αποθήκη αντιστοιχεί στο μηδέν. Η ζήτηση κάθε πελάτη i συμβολίζεται με q_i , Q η χωρητικότητα του οχήματος, για K πλήθος οχημάτων. Ακόμα συμβολίζουμε cij κόστος ταξιδιού από τον πελάτη i στον πελάτη j. Επιπλέον λαμβάνουμε υπόψιν ένα κόστος w_k σχετικό με την χρήση του οχήματος k.

Χρησιμοποιούμε δύο είδη δυαδικών μεταβλητών:

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

$$z_k = \begin{cases} 1, & \text{εάν το όχημα } k \text{ χρησιμοποιείται} \\ 0, & \text{αλλιώς} \end{cases}$$

Η αντικειμενική συνάρτηση ελαχιστοποίησης του κόστους, αντικατοπτρίζει το κόστος της συνολικής διαδρομής και το κόστους των χρησιμοποιημένων οχημάτων:

$$min \quad \sum_{k=1}^{K}\sum_{i=0}^{n}\sum_{j=0}^{n} c_{ij}x_{ij}^k + \sum_{k=1}^{K} w_k\, z_k$$

υπό:

$$\sum_{k=1}^{K}\sum_{i=0}^{n} x_{ij}^k = 1 \qquad \forall j = 1, \dots, n \qquad (2.7)$$

$$\sum_{k=1}^{K}\sum_{j=1}^{n} x_{ij}^k = 1 \qquad \forall i = 1, \dots, n \qquad (2.8)$$

$$x_{ij}^k \le z_k \qquad \forall k = 1, \dots, K, \forall i = 0, \dots, n\, , \forall j = 1, \dots, n \qquad (2.9)$$

,

$$\sum_{i=0}^{n} x_{iu}^k - \sum_{j=1}^{n} x_{uj}^k = 0 \qquad \forall k = 1, \dots, K, \forall u = 0, \dots, n \qquad (2.10)$$

,

$$\sum_{(i,j)\in S\times S}^{n} x_{ij}^k \le |S| - 1 \qquad \forall S \subseteq V: 1 \le |S| \le n\, , \forall k \qquad (2.11)$$

$$\sum_{j=1}^{n} q_j \left(\sum_{i=0}^{n} x_{ij}^k\right) \le Q \qquad \forall k = 1, \dots, K \qquad (2.12)$$

$$\sum_{j=1}^{n} x_{0j}^k \le 1 \qquad \forall k = 1, \dots, K \qquad (2.13)$$

$$\sum_{i=1}^{n} x_{i0}^{k} = 0 \qquad \forall k = 1, \dots, K \qquad (2.14)$$

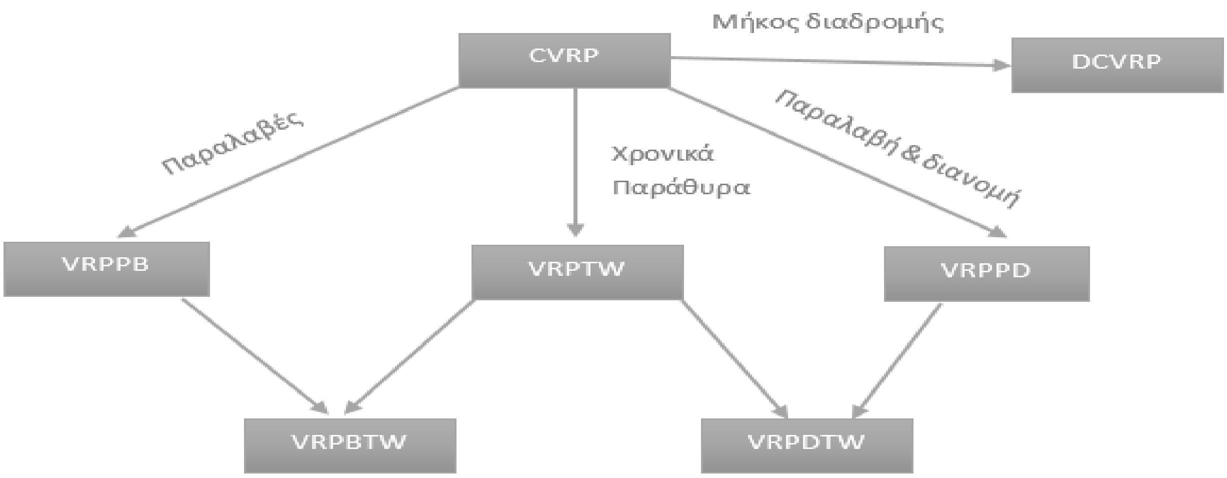
$$x_{ij}^{k} \in \{0,1\} \qquad \forall k = 1, \dots, K, \forall i = 0, \dots, n, \forall j = 1, \dots, n \qquad (2.15)$$

$$z_k \in \{0,1\} \qquad \forall k = 1, \dots, K \qquad (2.16)$$

Ο πρώτοι δύο περιορισμοί (2.7), (2.8) διασφαλίζουν ότι ακριβώς ένα όχημα εισέρχεται και εξέρχεται από κάθε πελάτη κόμβο και την αποθήκη. Ο επόμενος (2.9) σχετίζεται με τις μεταβλητές x, z και ελέγχει ότι κάθε πελάτης θα εξυπηρετηθεί από ενεργό όχημα. Ο πέμπτος περιορισμός (2.10) είναι η τυπική εξίσωση διατήρησης της ροής και εξασφαλίζει τη συνέχεια της διαδρομής, ο επόμενος (2.11) αποτρέπει τις υπό-διαδρομές και ο (2.12) θέτει άνω φράγμα στην χωρητικότητα των οχημάτων. Οι περιορισμοί (2.13), (2.14) διασφαλίζουν ότι μόνο ένα όχημα θα αναχωρήσει από την αποθήκη για να εξυπηρετήσει μία σειρά πελατών και ότι κανένα δεν θα γυρίσει πίσω.

2.4 Περισσότερα Προβλήματα Δρομολόγησης Οχημάτων.

Ξεκινώντας με το πρόβλημα δρομολόγησης οχημάτων περιορισμένης χωρητικότητας καθώς προσθέτουμε και αλλάζουμε περιορισμούς και μεταβλητές, δημιουργούμε τα βασικά είδη προβλημάτων δρομολόγησης και βλέπουμε την συσχέτιση τους.



Ακολουθεί μια συνοπτική αναφορά στα ποιο δημοφιλή είδη προβλημάτων δρομολόγησης οχημάτων.

2.4.1 Πρόβλημα Δρομολόγησης Οχημάτων με Χρονικά Παράθυρα (VRPTW)

Στο συγκεκριμένο πρόβλημα οι πελάτες-κόμβοι έχουν περισσότερα χαρακτηριστικά και η λύση θα πρέπει να ικανοποιήσει περισσότερους περιορισμούς. Σε αυτή την κατηγορία προβλημάτων έχουμε ένα σύνολο πελατών γεωγραφικά διασκορπισμένων σε συγκεκριμένη περιοχή. Ο κάθε ένας από αυτούς έχει συγκεκριμένη ποσότητα φορτίου που πρέπει να παραλάβει και η ιδιαιτερότητα του προβλήματος είναι ότι ο πελάτης πρέπει να εξυπηρετηθεί μέσα σε μια χρονική περίοδο, το χρονικό παράθυρο. Πριν και μετά το χρονικό παράθυρο, ο πελάτης δεν μπορεί να εξυπηρετηθεί. Τα οχήματα περιορισμένης χωρητικότητας βρίσκονται την στιγμή t=0 στην αποθήκη. Έτσι το πρόβλημα δρομολόγησης οχημάτων με χρονικά παράθυρα (Vehicle Routing Problem with Time Windows / VRPTW) προκύπτει σαν επέκταση του CVRP με ίδια αντικειμενική συνάρτηση ελαχιστοποίησης του κόστους, αλλά με επιπλέον περιορισμό την εξυπηρέτηση των πελατών μέσα σε συγκεκριμένα χρονικά παράθυρα (time windows) [14]

2.4.2 Πρόβλημα Δρομολόγησης Οχημάτων με Διανομή και Παραλαβή κατά τη Διάρκεια της Διαδρομής (VRPPD)

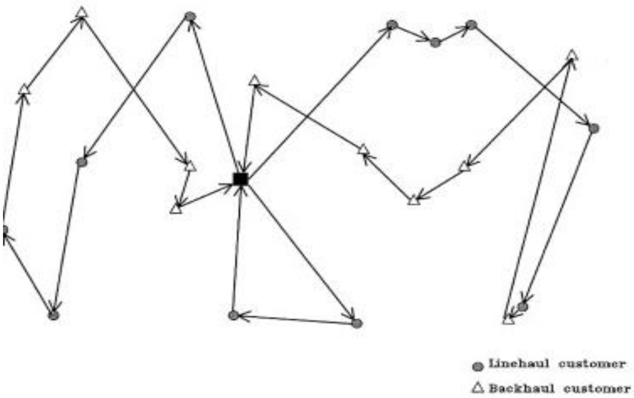
Όταν ο κάθε πελάτης για να εξυπηρετηθεί απαιτεί να του διανεμηθούν προϊόντα από το όχημα αλλά και το όχημα να παραλάβει προϊόντα από αυτόν, προκύπτει το πρόβλημα δρομολόγησης οχημάτων με διανομή και παραλαβή κατά τη διάρκεια της διαδρομής (Vehicle Routing Problem with pick-up and delivery /VRPPD). Στη βασική μορφή του προβλήματος, σε κάθε πελάτη αντιστοιχούνται δύο ποσότητες, η ποσότητα που θα διανεμηθεί d_i και αυτή που θα παραληφθεί p_i. Ακόμα για κάθε πελάτη καθορίζονται δύο κόμβοι: ο O_i και ο D_i, που είναι οι κόμβοι από τους οποίους ξεκινάνε τα προϊόντα που πρέπει να διανεμηθούν στον πελάτη και αυτοί που καταλήγουν τα προϊόντα που συλλέγονται από τον πελάτη [15]. Βασικός περιορισμός του προβλήματος είναι ότι σε κάθε πελάτη η διανομή γίνεται πριν από την παραλαβή και λόγου αυτού δεν πρέπει να παραβιάζονται τα παρακάτω:

- Για κάθε πελάτη i ο κόμβος O_i , αν δεν είναι η αποθήκη, πρέπει να εξυπηρετηθεί στην ίδια διαδρομή και πριν από τον πελάτη i.

- Για κάθε πελάτη i ο κόμβος D_i , αν δεν είναι η αποθήκη, πρέπει να εξυπηρετηθεί στην ίδια διαδρομή και μετά από τον πελάτη i.

2.4.3 Πρόβλημα Δρομολόγησης Οχημάτων με Δύο Είδη Πελατών (VRPB)

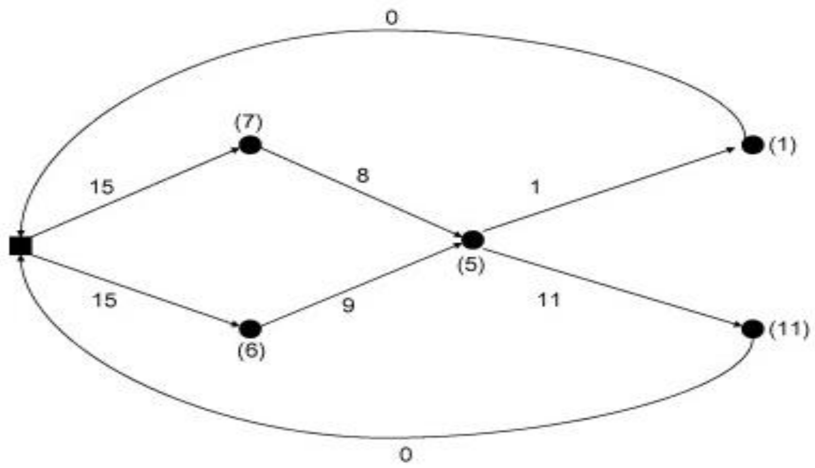
Ακόμα ένα πρόβλημα που αποτελεί επέκταση του κλασσικού CVRP είναι το πρόβλημα δρομολόγησης οχημάτων με δύο είδη πελατών κατά την διάρκεια της διαδρομής (Vehicle Routing Problem with Backhauls / VRPB) [16], [17]. Το πρώτο είδος είναι οι πελάτες που απαιτούν την διανομή κάποιας ποσότητας προϊόντων (linehauls customers) και το δεύτερο απαιτεί μια ποσότητα του προϊόντος να περισυλλεχθεί από αυτόν (backhauls customers). Εισάγουμε επιπλέον τους περιορισμούς ότι : σε κάθε διαδρομή οι πελάτες του δεύτερου τύπου επισκέπτονται έπειτα από τους πελάτες του πρώτου τύπου και ότι οι διαδρομές που περιλαμβάνουν πελάτες μόνο του δεύτερου τύπου δεν επιτρέπονται.



Σχήμα 2.4 Παράδειγμα λύσης VRPB (Πηγή: [11])

2.4.4 Το Πρόβλημα Δρομολόγησης Οχημάτων με Διαχωρισμένη Παράδοση(SDVRP)

Στο πρόβλημα δρομολόγησης οχημάτων με διαχωρισμένη παράδοση (Split Delivery Vehicle Routing Problem / SDVRP) ένας στόλος οχημάτων, ίδιας χωρητικότητας εξυπηρετεί ένα σύνολο πελατών, όμως σε αντίθεση με το κλασσικό VRP, ένα όχημα μπορεί να επισκεφθεί έναν πελάτη-κόμβο πάνω από μια φορά, όταν η ζήτηση του ξεπερνά την χωρητικότητα του οχήματος. Κάθε όχημα πρέπει να ξεκινάει και να καταλήγει στην ίδια αποθήκη. Στόχος του προβλήματος είναι η εύρεση ενός συνόλου διαδρομών που να εξυπηρετούν όλους τους πελάτες, χωρίς να υπερβαίνει η ποσότητα που παραδίδεται σε κάθε διαδρομή την χωρητικότητα του οχήματος και η ελαχιστοποίηση της συνολικής απόστασης που διανύεται. Το SDVRP έγινε γνωστό από τους Dror και Trudeau που ανέδειξαν την εξοικονόμηση που μπορεί να επιτύχει [11], [12].



Σχήμα 2.5: Παράδειγμα λύσης SDVRP (Πηγή: [13])

2.4.5 Το Πρόβλημα Δρομολόγησης Οχημάτων με ύπαρξη Πολλαπλών Αποθηκών (MDVRP)

Μια παραλλαγή του προβλήματος δρομολόγησης οχημάτων προκύπτει με την χρήση παραπάνω από μία αποθηκών (Vehicle Routing Problem with Multiple Depots /MDVRP). Το πρόβλημα μπορεί να αντιμετωπιστεί με δύο τρόπους:

i. Να λυθούν πολλά απλά VRP, αν καθεμία από τις αποθήκες έχει τον δικό της αριθμό οχημάτων και τους δικούς της πελάτες να εξυπηρετήσει.
ii. Να ξεκινάει ένα όχημα από μία αποθήκη και είτε να τερματίζει σε άλλη, είτε να σταματάει ενδιάμεσα σε κάποια άλλη και να συνεχίσει έπειτα την διαδρομή του.

2.4.6 Το Περιοδικό Πρόβλημα Δρομολόγησης Οχημάτων (PVRP)

Το περιοδικό πρόβλημα δρομολόγησης οχημάτων (Periodic Vehicle Routing Problem/ PVRP) είναι ισοδύναμο με το MDVRP αν αντικαταστήσουμε της αποθήκες με ημέρες (έστω ημέρες, η μετρική μονάδα του χρόνου). Στο περιοδικό VRP οι πελάτες εξυπηρετούνται μία ή περισσότερες φορές μέσα στο χρονικό ορίζοντα, με διαδρομές που πραγματοποιούνται σε κάθε μία χρονική περίοδο.

2.4.7 Στοχαστικά Πρόβλημα Δρομολόγησης Οχημάτων (SVRP)

Σε πρακτικές εφαρμογές μία ή περισσότερες παράμετροι των προβλημάτων δρομολόγησης τείνουν σε τυχαίο ή στοχαστικό χαρακτήρα, έτσι έγιναν δημοφιλή τα στοχαστικά προβλήματα δρομολόγησης οχημάτων (Stochastic Vehicle Routing Problem/ SVRP). Σε αυτά τα προβλήματα συναντάμε πολλές μεταβλητές με στοχαστικό χαρακτήρα όπως: η ζήτηση, οι πελάτες και ο χρόνο διαδρομής (ή/και εξυπηρέτησης) ,έχουν όμως γνωστές κατανομές πιθανοτήτων [18].

2.4.8 Το δυναμικό πρόβλημα δρομολόγησης οχημάτων (DVRP)

Στα δυναμικά προβλήματα δρομολόγησης οχημάτων (Dynamic Vehicle Routing Problem) κάποια από τα δεδομένα του προβλήματος δεν είναι γνωστά εκ των προτέρων. Καινούργιες πληροφορίες γνωστοποιούνται και μορφοποιούν το πρόβλημα καθώς οι διαδρομές βρίσκονται σε εξέλιξη και στις περισσότερες περιπτώσεις απαιτείται άμεση ή γρήγορη απόκριση. Οι νέες αυτές πληροφορίες συχνά αφορούν καινούργιους πελάτες που πρέπει να εισαχθούν στην διαδρομή ή άλλους υπάρχοντες που τελικά ακυρώνονται. Ακόμα μεταβαλλόμενες παράμετροι μπορεί να είναι ο χρόνος ταξιδιού ή η ζήτηση του κάθε πελάτη [19].

Κεφάλαιο 3:
Αλγόριθμοι βελτιστοποίησης προβλημάτων εφοδιαστικής αλυσίδας

3.1 Απλοί Ευρετικοί Αλγόριθμοι (Heuristics)

Η επίλυση ενός προβλήματος βελτιστοποίησης και ιδιαίτερα συνδυαστικής βελτιστοποίησης γίνεται ολοένα και δυσκολότερη όσο αυξάνει το μέγεθος του προβλήματος και πολλές φορές το να προσπαθούμε να βρούμε την ολικά βέλτιστη λύση σε λογικό χρόνο είναι πρακτικά αδύνατο. Για να επιλυθούν προβλήματα αυτής της μορφής συχνά καταφεύγουμε σε διαφορετικές τεχνικές που μας οδηγούν σε μια σχεδόν βέλτιστη, αλλά ικανοποιητική λύση. Μια λύση ενός ευρετικού αλγορίθμου γίνεται αποδεκτή αν ικανοποιεί κάποια κριτήρια όπως η ποιότητα της λύσης και η ευκολία απόκτησης της.

Για κάθε πρόβλημα βελτιστοποίησης δεν υπάρχει μονάχα ένας ευρετικός αλγόριθμος που να δίνει τη βέλτιστη λύση, αλλά έχουν αναπτυχθεί αρκετοί αλγόριθμοι οι οποίοι συγκρινόμενοι μεταξύ τους, οδηγούν ολοένα και σε καλύτερες λύσεις. Σε ότι αφορά την ποιότητα της λύσης, σε μερικά προβλήματα είναι αδύνατο να βρεθεί η βέλτιστη λύση για κάποιο πρόβλημα σε ικανοποιητικό χρόνο.

Διάφορες κατηγορίες των ευρετικών αλγορίθμων είναι οι ακόλουθες :

- Αλγόριθμοι απληστίας (greedy algorithms).
  Οι αλγόριθμοι απληστίας προσπαθούν να οδηγήσουν σε μια εφικτή λύση του προβλήματος, αλλά πολλές φορές χρειάζονται πάρα πολύ μεγάλο χρόνο γιατί είναι μυωπικοί αλγόριθμοι, δηλαδή βλέπουν μόνο μπροστά.

- Προσεγγιστικοί αλγόριθμοι (approximation algorithms).
  Οι προσεγγιστικοί αλγόριθμοι προσπαθούν να λύσουν αυτό το πρόβλημα χρησιμοποιώντας επιπλέον πληροφορία.

- Αλγόριθμοι τοπικής αναζήτησης (local search algorithms).
  Οι αλγόριθμοι τοπικής αναζήτησης προσπαθούν ξεκινώντας από μια αρχική εφικτή λύση να βελτιώσουν τη λύση με κάποια μέθοδο αναζήτησης στη γειτονιά της λύσης.

Συνοπτικά μπορούμε να πούμε ότι οι δύο πρώτες κατηγορίες χρησιμοποιούνται για να παράγουμε μία αρχική λύση ενώ η τρίτη προσπαθεί να βελτιώσει μία ήδη υπάρχουσα λύση.

3.1.1 Κλασσικοί Ευρετικοί Αλγόριθμοι (Classical Heuristics)

Αρκετοί ευρετικοί αλγόριθμοι έχουν εφευρεθεί για την επίλυση του προβλήματος δρομολόγησης οχημάτων (VRP).Σύμφωνα με την προτεινόμενη ταξινόμηση κατά Laporte & Semet (2002) [20] περιγράφουμε τους κλασσικούς ευρετικούς για προβλήματα δρομολόγησης στα πλαίσια των παρακάτω ενοτήτων: κατασκευαστικοί, μέθοδος δύο φάσεων και μέθοδος βελτίωσης της διαδρομής.

- Κατασκευαστικοί ευρετικοί αλγόριθμοι.

Η μέθοδος κατασκευής της διαδρομής ήταν η πρώτη που εφαρμόστηκε σε προβλήματα δρομολόγησης οχημάτων περιορισμένης χωρητικότητας (CVRP). Οι αλγόριθμοι αυτοί ξεκινούν από μία άδεια λύση και επαναληπτικά κτίζουν διαδρομές εισάγοντας έναν πελάτη σε κάθε επανάληψη, μέχρι να έχουν εξυπηρετηθεί όλοι οι πελάτες. Η διαδικασία έχει δύο εκδοχές την παράλληλη και την ακολουθητική, ανάλογα με τον αριθμό των επιλέξιμων διαδρομών για την εισαγωγή ενός πελάτη. Ο πρώτος κατασκευαστικός αλγόριθμος έχει προταθεί για την επίλυση του προβλήματος δρομολόγησης οχημάτων είναι ο αλγόριθμος των εξοικονομήσεων Clarke & Whright [21]. Σε αυτή την μέθοδο αρχικά υπολογίζονται εξοικονομήσεις όλων των πελατών και στην συνέχεια δημιουργούνται διαδρομές βάση των καλύτερων εξοικονομήσεων .

- Ευρετικοί αλγόριθμοι δύο φάσεων.

Η μέθοδος δύο φάσεων βασίζεται στην αποσύνθεση της διαδικασίας της λύσης ενός προβλήματος δρομολόγησης οχημάτων σε δύο χωριστά υπό-προβλήματα:

1. Ομαδοποίηση (clustering) πελατών σε υποσύνολα και αντιστοίχιση μίας διαδρομής σε κάθε ένα από αυτά.
2. Δρομολόγηση (routing) των οχημάτων, καθορισμός δηλαδή τις σειράς με την οποία θα περάσουν από τους πελάτες.

Γνωστοί αλγόριθμοι είναι, ο αλγόριθμος σαρώματος των Gillet & Miller [22] και η μέθοδος ομαδοποίηση πρώτα-δρομολόγηση έπειτα των Fisher & Jaikumar [23] .

- Ευρετικοί αλγόριθμοι βελτίωσης της διαδρομής.

Ουσιαστικά αναφερόμαστε στις μεθόδους τοπικής αναζήτησης που χρησιμοποιούνται στην βελτίωση λύσεων που έχουν προκύψει από άλλους ευρετικούς αλγορίθμους.

3.1.2 Αλγόριθμοι Απληστίας (Greedy algorithm)

Σχεδόν όλα τα προβλήματα βελτιστοποίησης έχουν n εισόδους (έστω κόμβοι, για το πρόβλημα δρομολόγησης οχημάτων), ένας αλγόριθμος απληστίας (Greedy algorithm) δουλεύει σταδιακά, χρησιμοποιώντας ένα από τα δεδομένα εισόδου κάθε φορά, με σκοπό να καθορίσει ένα υποσύνολο που να ικανοποιεί τους περιορισμούς ,δηλαδή μια εφικτή λύση. Σε κάθε στάδιο μια απόφαση παίρνεται σύμφωνα με το αν ο συγκεκριμένος για παράδειγμα κόμβος μπορεί να οδηγήσει σε βέλτιστη λύση κατατάσσοντας τα δεδομένα εισόδου σε κάποια σειρά, βάση ενός κριτηρίου βελτιστοποίησης. Εάν ο συνυπολογισμός του επόμενου κόμβου οδηγήσει σε μη-εφικτή λύση, τότε αυτός ο κόμβος τελικά δεν θα συμπεριληφθεί στην λύση.

Έτσι μπορούμε να πούμε ότι ξεκινώντας από μία μερική λύση σχηματίζουμε μία πλήρη λύση και σε κάθε βήμα επιλέγουμε μεταξύ των υποψήφιων δεδομένων εισόδου, εκείνο που διατηρεί την δυνατότητα ολοκλήρωσης της λύσης σε εφικτή και βελτιστοποιεί κάποιο κριτήριο και στο τέλος δημιουργείται μία τελική εφικτή λύση με σχετικά καλό αποτέλεσμα. Γενικά η αρχή της απληστίας είναι η αρχή ανάπτυξης μυωπικών κατασκευαστικών αλγορίθμων, αλγορίθμων που κατασκευάζουν δηλαδή μία εφικτή λύση για δεδομένο πρόβλημα σε χρόνο πολυωνυμικό ως προς το μέγεθος των δεδομένων.

Επειδή όμως είναι μυωπικοί και η λύση κατασκευάζεται δηλαδή τμηματικά, βρίσκοντας την καλύτερη λύση από το σημείο που βρισκόμαστε μία δεδομένη στιγμή και δεν ελέγχονται άλλες λύσεις που πιθανά δίνουν καλύτερο αποτέλεσμα, οι αλγόριθμοι αυτοί δίνουν βέλτιστη λύση μόνο υπό προϋποθέσεις. Στις περισσότερες περιπτώσεις δίνουν ικανοποιητικές αρχικές εφικτές λύσεις, οι οποίες τροφοδοτούνται σε άλλους αλγόριθμους (όπως τους μεθευρετικούς), που την βελτιστοποιούν.

3.1.3 Αλγόριθμοι Τοπικής Αναζήτησης

Η τοπική αναζήτηση βασίζεται [24] στην αρχαιότερη μέθοδο βελτιστοποίησης, στη μέθοδο δοκιμής και σφάλματος. Η τοπική αναζήτηση έχει αποδειχθεί πολύ επιτυχημένη στην πράξη σε ένα πολύ μεγάλο αριθμό από προβλήματα συνδυαστικής βελτιστοποίησης. Στη συνέχεια περιγράφεται ένας γενικός αλγόριθμος.

Δοθέντος ενός προβλήματος βελτιστοποίησης, επιλέγουμε τη γειτονιά και αρχίζουμε από μια αρχική εφικτή λύση και χρησιμοποιούμε μία υπό-ρουτίνα για να ψάξουμε για μια καλύτερη λύση στη γειτονιά της αρχικής λύσης. Όσο βρίσκεται μια βελτιωμένη λύση, την εφαρμόζουμε και επαναλαμβάνουμε τη διαδικασία αναζήτησης από τη νέα λύση, όταν φθάσουμε σε τοπικό ελάχιστο σταματάμε γιατί η λύση δεν θα βελτιώνεται παραπάνω.

Ένα πολύ σημαντικό χαρακτηριστικό της τοπικής αναζήτησης είναι το γεγονός ότι μπορεί να εκτελείται από πολλά διαφορετικά αρχικά σημεία και να επιλέγεται το καλύτερο ως βέλτιστη λύση. Σε αυτές τις περιπτώσεις μια άλλη απόφαση που πρέπει να παρθεί είναι πόσα θα είναι τα αρχικά σημεία που θα πρέπει να επιλέξουμε. Το επόμενο πρόβλημα που πρέπει να επιλυθεί για να είμαστε σε θέση να πούμε ότι η τοπική αναζήτηση που εφαρμόζουμε οδηγεί σε ικανοποιητικά αποτελέσματα είναι το ότι θα πρέπει να γίνει σωστή επιλογή της γειτονιάς που θα πραγματοποιηθεί η αναζήτηση.

Αναλυτικά αυτό που μπορούμε να πούμε για την τοπική αναζήτηση είναι ότι μπορεί να θεωρηθεί ως ο παλιότερος και απλούστερος μεθευρετικός αλγόριθμος [25]. Όπως έχει ήδη αναφερθεί, αρχίζουμε με κάποια δεδομένη λύση, όπως όλοι οι μεθευρετικοί αλγόριθμοι που θα περιγράψουμε στη συνέχεια. Η λύση παράγεται είτε με τυχαίο τρόπο είτε με κάποιο αλγόριθμο απληστίας. Σε κάθε επανάληψη ο αλγόριθμος αντικαθιστά την τρέχουσα λύση με μία γειτονική λύση που βελτιώνει την αντικειμενική συνάρτηση. Η αναζήτηση σταματά όταν όλες οι υποψήφιες γειτονιές έχουν εξερευνηθεί και καμία καλύτερη λύση δεν έχει βρεθεί, πράγμα που σημαίνει ότι βρέθηκε ένα τοπικό ελάχιστο. Αν έχουμε μεγάλες γειτονιές αναζήτησης τότε οι υποψήφιες λύσεις μπορεί να είναι ένα υποσύνολο της συνολικής γειτονιάς. Η μη εξερεύνηση όλης της γειτονιάς γίνεται για να επιταχύνουμε τη διαδικασία.

3.2 Μεθευρετικοί Αλγόριθμοι (Metaheuristics)

3.2.1 Εισαγωγή στους Μεθευρετικούς Αλγορίθμους

Οι μεθευρετικοί αλγόριθμοι (Metaheuristics) είναι μέθοδοι επίλυσης που συνδυάζουν διαδικασίες τοπικής αναζήτησης και υψηλότερου επιπέδου στρατηγικές για να δημιουργήσουν μια διαδικασία που είναι ικανή να ξεφύγει από κάποιο τοπικό ελάχιστο. Συνήθως χρησιμοποιούν κλασσικούς ευρετικούς αλγορίθμους σαν υπό-διαδικασίες τους. Οι μεθευρετικοί μπορούν να χωριστούν σε κατηγορίες ανάλογα με το πόσες λύσεις χρησιμοποιούν. Υπάρχουν οι αλγόριθμοι που χρησιμοποιούν μία λύση και κάνουν αναζήτηση στη γειτονιά αυτής της λύσης και υπάρχουν και οι αλγόριθμοι που έχουν ένα πληθυσμό από λύσεις και προσπαθούν να κάνουν αναζήτηση σε όλο το χώρο λύσεων. Φυσικά υπάρχουν και υβριδικές μορφές αυτών των δύο κατηγοριών (hybrid or memetic algorithms).Οι αλγόριθμοι που χρησιμοποιούν μία λύση και κάνουν εξερεύνηση στη γειτονιά αναζήτησης γύρω από τη λύση έχουν πάρα πολύ ισχυρές δυνατότητες εκμετάλλευσης η εντατικοποίησης (exploitation ή intensification) της περιοχής γύρω από τη λύση. Από την άλλη μεριά οι αλγόριθμοι που έχουν πληθυσμό λύσεων, έχουν πολύ ισχυρές δυνατότητες εξερεύνησης ή διάχυσης (exploration ή diversification) σε όλο το χώρο λύσεων.

Στην παρούσα διπλωματική εργασία θα ασχοληθούμε με τους μεθευρετικούς αλγόριθμους που επικεντρώνονται στην όσο το δυνατό πιο ενδελεχή αναζήτηση γύρω από κάποιο σημείο που έχει βρεθεί το τοπικό ελάχιστο με τη μέθοδο τοπικής αναζήτησης και θα παρουσιάσουμε εφαρμογή για έναν από αυτούς.
Οι αλγόριθμοι αυτοί χωρίζονται σε τέσσερις κύριες κατηγορίες ανάλογα με τον τρόπο που χρησιμοποιούν για να αποφύγουν το τοπικό ελάχιστο :

- Επαναληπτικές διαδικασίες που αρχίζουν από διαφορετικές αρχικές λύσεις.
  Χαρακτηριστικότεροι εκπρόσωποι αυτής της κατηγορίας είναι οι αλγόριθμοι Πολυεναρκτήριας Τοπικής Αναζήτησης (Multistart Local Search),της Επαναληπτικής Τοπικής Αναζήτησης (Iterated Local Search) και η μέθοδος της διαδικασίας Άπληστης Τυχαιοποιημένης Προσαρμοστικής Αναζήτησης (Greedy Randomized Adaptive Search Procedure /GRASP).

- Αλγόριθμοι που δέχονται γειτονικές κινήσεις που δεν βελτιώνουν τη λύση.
  Σε αυτές τις μεθόδους μία κίνηση που δεν βελτιώνει τη λύση μπορεί να γίνει υπό κάποιες συνθήκες αποδεκτή. Με αυτό τον τρόπο μπορεί σε μία από τις επόμενες κινήσεις να ξεφύγουμε από το τοπικό ελάχιστο και να οδηγηθούμε σε κάποιο επόμενο τοπικό ελάχιστο το οποίο να είναι καλύτερο από το τρέχων τοπικό ελάχιστο. Οι δύο πιο χαρακτηριστικοί αλγόριθμοι που εκφράζουν αυτή την κατηγορία και στην ουσία δημιούργησαν το χώρο των μεθευρετικών αλγορίθμων είναι η Προσομοιωμένη Ανόπτηση (Simulated Annealing) και η Περιορισμένη Αναζήτηση (Tabu Search).

- Αλγόριθμοι που αλλάζουν τη γειτονιά αναζήτησης.
  Αυτή η κατηγορία των αλγορίθμων αποτελείται από αλγόριθμους οι οποίοι όταν κολλήσουν σε κάποιο τοπικό ελάχιστο αλλάζουν τον αλγόριθμο που χρησιμοποιούν για την αναζήτηση σε γειτονικά σημεία του χώρου λύσεων. Οι πιο χαρακτηριστικές μέθοδοι αυτής της κατηγορίας είναι ο αλγόριθμος Μεταβλητής Γειτονιάς Αναζήτησης (Variable Neighborhood Search /VNS) και ο αλγόριθμος επέκτασης της γειτονιάς αναζήτησης (Expanding Neighborhood Search /ENS)

- Αλγόριθμοι που αλλάζουν την αντικειμενική συνάρτηση ή κάποια από τα δεδομένα του προβλήματος.
  Σε αυτή την κατηγορία των αλγορίθμων το πρόβλημα μετατρέπεται είτε αλλάζοντας την αντικειμενική συνάρτηση είτε αλλάζοντας τους περιορισμούς του προβλήματος. Η πιο χαρακτηριστική μέθοδος αυτής της κατηγορίας είναι ο αλγόριθμος καθοδηγούμενης τοπικής αναζήτησης (Guided Local Search).

3.2.2 Η Διαδικασία της Άπληστης Τυχαιοποιημένης Προσαρμοστικής Αναζήτησης (GRASP)

Η διαδικασία άπληστης τυχαιοποιημένης προσαρμοστικής αναζήτησης (Greedy Randomized Adaptive Search Procedure/ GRASP)είναι μια επαναληπτική διαδικασία για την εύρεση προσεγγιστικών λύσεων προβλήματα συνδυαστικής βελτιστοποίησης [26], [27], [28]. Αυτή η τυχαιοποιημένη τεχνική παρέχει μια εφικτή λύση σε κάθε επανάληψη. Οι επαναλήψεις της διαδικασίας της άπληστης τυχαιοποιημένης προσαρμοστικής αναζήτησης σταματούν όταν κάποιο κριτήριο τερματισμού ικανοποιείται. Το τελικό αποτέλεσμα είναι απλά η καλύτερη λύση που βρέθηκε από όλες τις επαναλήψεις. Κάθε επανάληψη αποτελείται από δύο φάσεις, μια φάση κατασκευής μιας αρχικής λύσης (Construction Phase) και μια διαδικασία τοπικής αναζήτηση (Local Search Phase) για βελτιστοποίηση αυτής της 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. Συμπεραίνουμε λοιπόν ότι ο αλγόριθμος συνδυάζει άπληστο ευρετικό αλγόριθμο, τυχαιότητα και τοπική αναζήτηση και στο επόμενο κεφάλαιο θα παρουσιάσουμε την εφαρμογή του σε πρόβλημα δρομολόγησης οχημάτων.

1. Φάση κατασκευής

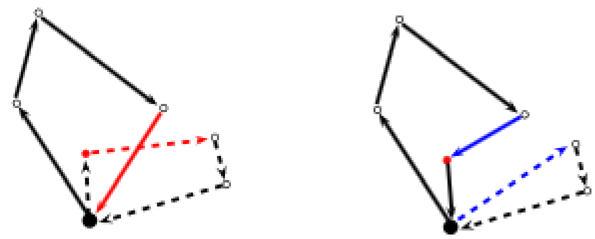
   Η φάση κατασκευής επιτυγχάνεται μέσω τις μεθόδου Τυχαιοποιημένης Απληστίας (Randomized Greedy).Η μέθοδος μπορεί να περιγράφει ως επαναληπτική πρόσθεση ενός στοιχείου στην μη ολοκληρωμένη λύση σε κάθε επανάληψη. Η στρατηγική επιλογής του επόμενου στοιχείου βασίζεται στην τυχαία επιλογή από μια λίστα υποψηφίων, που ονομάζεται λίστα περιορισμού των υποψηφίων (Restricted Candidate List) για εισαγωγή στη λύση. Η λίστα περιορισμού δημιουργείται, εισάγοντας έναν περιορισμένο αριθμό στοιχείων, τα οποία κατατάσσονται βάσει μιας συνάρτησης απληστίας. Η τυχαία επιλογή του στοιχείου σημαίνει ότι δεν είναι ανάγκη να επιλεγεί το πρώτο στοιχείο στη λίστα.. Η κατασκευή της λίστας περιορισμού των υποψηφίων είναι ίσως το πιο σημαντικό κομμάτι της μεθόδου, αφού από αυτό ελέγχεται η διασπορά των λύσεων που θα παραχθούν. Αν, για παράδειγμα, η λίστα είναι πολύ μικρή τότε οι λύσεις που θα παράγονται θα είναι σχεδόν όμοιες μεταξύ τους. Από την άλλη μεριά αν η λίστα είναι πολύ μεγάλη τότε θα πρόκειται πλέον για αλγόριθμο που κατασκευάζεται με τυχαίο τρόπο. Ο ευρετικό αλγόριθμος είναι προσαρμοστικός με την έννοια ότι η συνάρτηση επιλογής προσαρμόζεται για να προσμετρήσει τα στοιχεία που έχουν ήδη επιλεγεί.
   Εφαρμόζοντας την μέθοδο σε προβλήματα δρομολόγησης οχημάτων, θεωρούμε την λίστα περιορισμού υποψηφίων, σαν ένα υποσύνολο πελατών (δεν απαριθμούνται όλοι οι πελάτες), οι οποίοι έχουν ταξινομηθεί βάση του κόστους απόστασης που έχουν από την αποθήκη ή από τον τελευταίο πελάτη που έχει εξυπηρετηθεί, από αυτούς διαλέγουμε τυχαία έναν.
   Έπειτα σύμφωνα με την μέθοδο της Τυχαιοποιημένης Απληστίας, εξετάζουμε αν ο πελάτης που επιλέχτηκε τυχαία από τους k-κοντινότερους στον προηγούμενο (από την λίστα περιορισμού), ικανοποιεί τους περιορισμούς του αντίστοιχου προβλήματος που επιλύουμε (όπως η ζήτηση και ο χρόνος εξυπηρέτησης) και μπορεί να αποτελέσει συνέχεια της διαδρομής. Η

διαδικασία συνεχίζεται μέχρι κάποιος πελάτης να μην ικανοποιεί τους περιορισμούς, οπότε μπορεί να επιστρέψει στην αποθήκη.Η διαδικασία επαναλαμβάνεται μέχρι να εξυπηρετηθούν όλοι οι πελάτες και να σχηματιστούν οι διαδρομές, που αποτελούν εφικτή λύση, συνήθως όμως δεν αντιστοιχεί σε τοπικό (ή ολικό βέλτιστο) για αυτό και ακολουθεί η φάση της τοπικής αναζήτησης.

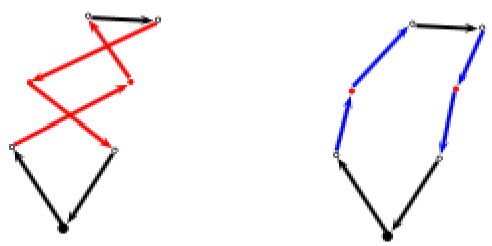
2. Φάση τοπικής αναζήτησης

Η τοπική αναζήτηση εφαρμόζεται ώστε από μια εφικτή λύση, να οδηγηθούμε σε άλλη καλύτερη που θα περιέχει τοπικό ελάχιστο. Για να εφαρμοστεί η φάση τοπικής αναζήτησης, καθορίζεται μια συνάρτηση που να κάνει αναζήτηση στη γειτονιά της αρχικής λύσης μέχρι να φτάσει σε τοπικό ελάχιστο.
Ο Waters [29] έχει προτείνει τέσσερις διαδικασίες ανταλλαγής που μπορούν να εφαρμοστούν μεταξύ δύο ή περισσότερων διαδρομών, σε προβλήματα δρομολόγησης οχημάτων:

i. <u>1-0 Επανατοποθέτηση (1-0 relocate)</u>
Μια απλή διαγραφή ενός πελάτη από μια διαδρομή και επανατοποθέτηση του σε μια άλλη διαδρομή με καλύτερο κόστος.



ii. <u>1-1 Ανταλλαγή (1-1 exchange)</u>
Μία ταυτόχρονη ανταλλαγή δύο πελατών που βρίσκονται σε διαφορετικές διαδρομές.



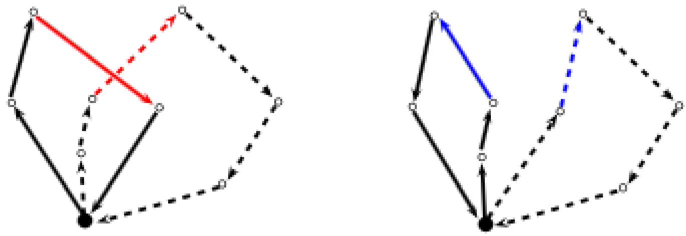
iii. <u>1-2 Ανταλλαγή (1-2 exchange)</u>
Μία συνδυασμένη ανταλλαγή όπου ένας πελάτης μιας διαδρομής ανταλλάσσεται με δυο πελάτες που βρίσκονται σε διαφορετική διαδρομή.

iv. <u>2-2 Ανταλλαγή (2-2 exchange)</u>
Μία συνδυασμένη ανταλλαγή όπου δυο πελάτες μιας διαδρομής ανταλλάσσονται με δυο πελάτες που βρίσκονται σε διαφορετική διαδρομή.

Άλλες αποτελεσματικές μέθοδοι τοπικής αναζήτησης:

- <u>Μέθοδος 2-opt</u>

Η μέθοδος αυτή σε γενικές γραμμές διαγράφει δυο ακμές και επανασυνδέει τα δύο μονοπάτια με διαφορετικό τρόπο, έτσι ώστε να καθοριστεί μια καινούργια διαδρομή, η οποία θα αποφέρει καλύτερο κόστος [30].



Η διαδικασία της μεθόδου 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 την i και την επόμενή της μέσα στη διαδρομή. Αν με αυτό τον τρόπο μπορούμε να μειώσουμε το κόστος της διαδρομής, τότε επιλέγουμε την καλύτερη 2-opt κίνηση και εφαρμόζουμε τις αλλαγές στη διαδρομή T.

Βήμα 3. Αν δεν μπορούμε να βρούμε επιπλέον βελτίωση, σταματάμε.

Ένα πολύ σημαντικό σημείο του αλγόριθμου είναι η επιλογή των τόξων που θα διαγραφούν από τη λύση και των τόξων που θα πρέπει να εισέλθουν στη λύση. Υπάρχουν αρκετές προσεγγίσεις που έχουν αναφερθεί στη βιβλιογραφία. Μία προσέγγιση είναι να διαγράψουμε τα δύο χειρότερα τόξα και να αντικαταστήσουμε με δύο άλλα τόξα. Στο συγκεκριμένο αλγόριθμο όταν θα διαγραφούν δύο τόξα τότε υπάρχει μόνο ένας τρόπος για να επανασυνδεθούν, αλλιώς δεν διατηρείται ο κύκλος και δημιουργούνται δύο διαφορετικοί κύκλοι. Ο δεύτερος τρόπος είναι να διαγραφεί το χειρότερο και να δοκιμάσουμε διάφορες παραλλαγές με τα υπόλοιπα τόξα για να δούμε πότε βελτιώνεται μία λύση. Ο τρίτος τρόπος είναι να διαγράφονται δύο τόξα κάθε φορά τυχαία. Και οι τρεις τρόποι έχουν τα πλεονεκτήματα τους και τα μειονεκτήματα τους. Ένα πλεονέκτημα για παράδειγμα του τρίτου τρόπου είναι ότι μπορεί να γίνει πιο γρήγορος λόγω του ότι δεν χρειάζεται να υπολογίζει σε κάθε επανάληψη τα χειρότερα τόξα και ταυτόχρονα, ακριβώς ο ίδιος λόγος μπορεί να κάνει τον αλγόριθμο πιο αργό αν δεν μπορεί να βρει εύκολα βελτίωση σε κάποια λύση.

- Μέθοδος 3-opt

  Αντίστοιχη με την προηγούμενη μέθοδο, μόνο που μπορούμε να διαχωρίσουμε τον αλγόριθμο σε τρία μέρη αντί για δύο και να συνδυάσουμε τα μονοπάτια με τον καλύτερο δυνατό τρόπο.

Στη συνέχεια παρουσιάζεται ο ψευδοκώδικας της μεθόδου Άπληστης Τυχαιοποιημένης Προσαρμοστικής Αναζήτησης (GRASP):

ΓΙΑ όλες τις επαναλήψεις

ΓΙΑ *n* βήματα

Εφαρμογή Τυχαιοποιημένης Συνάρτησης Απληστίας
(για δημιουργία νέας λύσης)
Αποθήκευση λύσης και του αντίστοιχου κόστους

ΤΕΛΟΣ ΓΙΑ

ΓΙΑ (όλες τις λύσεις)

Εφαρμογή Τοπικής Αναζήτησης
Υπολογισμός της αντικειμενικής συνάρτησης (κόστους) της λύσης

ΑΝ (κόστος νέας λύσης < κόστος παλαιότερης)
Αντικατάσταση παλαιάς λύσης και του αντίστοιχου κόστους
ΤΕΛΟΣ ΑΝ

ΤΕΛΟΣ ΓΙΑ

ΤΕΛΟΣ ΓΙΑ

Κεφάλαιο 4:
Περιγραφή και επίλυση των προβλημάτων δρομολόγησης οχημάτων

4.1 Συνοπτική περιγραφή των προβλημάτων

Στην παρούσα διπλωματική εργασία ασχολούμαστε με την επίλυση των εξής προβλημάτων: το πρόβλημα δρομολόγησης οχημάτων περιορισμένης χωρητικότητας (Capacitated Vehicle Routing Problem/ CVRP) και το ανοιχτό πρόβλημα δρομολόγησης οχημάτων (Open Vehicle Routing Problem/ OVRP).
Σκοπός των δύο προβλημάτων, όπως αναφέρθηκε στα προηγούμενα κεφάλαια, είναι ο καθορισμός των βέλτιστων διαδρομών στόλου οχημάτων που ξεκινάνε από μια αποθήκη και εξυπηρετούν όλους τους πελάτες, ώστε να ελαχιστοποιηθεί το συνολικό κόστος. Τα οχήματα του στόλου είναι ομογενή, έχουν δηλαδή ίδια πεπερασμένη χωρητικότητα και ξεκινούν από την ίδια αποθήκη. Κάθε πελάτης έχει πεπερασμένη ζήτηση που πρέπει να καλυφθεί και χρόνο εξυπηρέτησης. Θεωρούμε ότι το κόστος της διαδρομής προκύπτει αθροιστικά από τις επιμέρους αποστάσεις μεταξύ των πελατών που ανήκουν στην διαδρομή.
Οι περιορισμοί που διέπουν τα προβλήματα είναι οι εξής :

1. Κάθε πελάτης πρέπει να εξυπηρετείται μια μόνο φορά, από ένα και μόνο ένα φορτηγό.
2. Το σύνολο της ζήτησης που πρέπει να καλυφθεί σε κάθε διαδρομή, δεν θα πρέπει να ξεπερνάει την χωρητικότητα του οχήματος.
3. Η συνολική απόσταση που διανεύει το όχημα της διαδρομής δεν θα πρέπει να ξεπερνάει ένα μέγιστο μήκος διαδρομής, το όποιο είναι προκαθορισμένο, αυτό συνυπολογίζουμε κάθε φορά και τον χρόνο εξυπηρέτησης κάθε πελάτη.

Οι παραπάνω περιορισμοί καλύπτουν πλήρως το ανοιχτό πρόβλημα δρομολόγησης οχημάτων (OVRP). Το πρόβλημα δρομολόγησης οχημάτων περιορισμένης χωρητικότητας (CVRP) όμως επιβάλει και έναν επιπλέον περιορισμό.

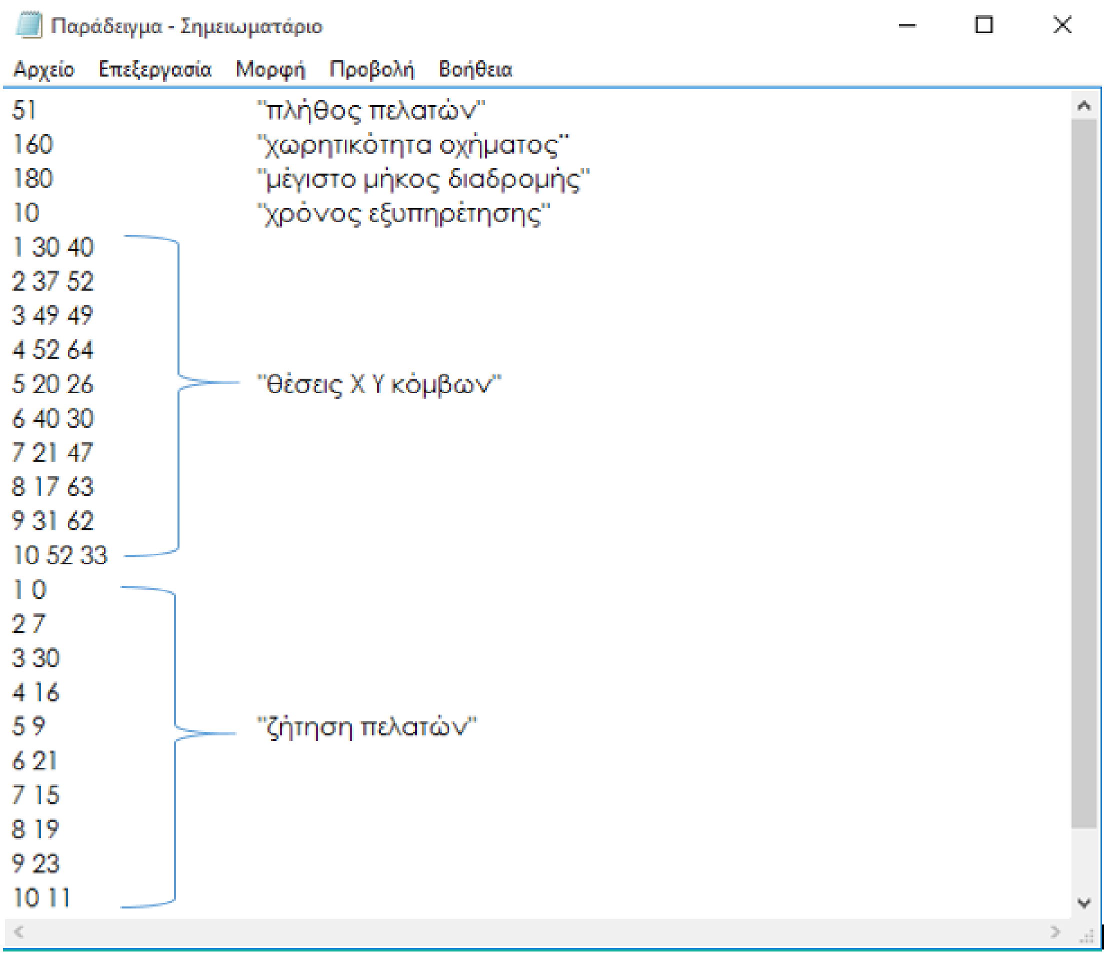
4. Όλα τα οχήματα πρέπει να ξεκινάνε και να τερματίζουν στην ίδια αποθήκη.

Άρα όταν κάποια συνθήκη περιορισμού δεν ικανοποιείται ή έχουν εξυπηρετηθεί όλοι οι πελάτες, τα οχήματα είναι αναγκασμένα να επιστρέψουν στην αποθήκη από την οποία ξεκίνησαν.

Τα παραπάνω προβλήματα θα επιλυθούν με την εφαρμογή της μεθόδου, άπληστης τυχαιοποιημένης προσαρμοστικής αναζήτησης (Greedy Randomized Adaptive Search Procedure/ GRASP), η οποία αναλύθηκε στο προηγούμενο κεφάλαιο. Όπως έχει αναφερθεί, η μέθοδος χωρίζεται σε δύο φάσεις: την εύρεση μίας αρχικής εφικτής λύσης με τον αλγόριθμο απληστίας (Greedy Algorithm) και έπειτα βελτίωση της με εφαρμογή τοπικής αναζήτησης (Local Search).

4.2 Μοντελοποίηση των προβλημάτων

Βάση της παραπάνω περιγραφής τα δεδομένα που χρειαζόμαστε για την επίλυση των προβλημάτων είναι το πλήθος των πελατών , η ζήτηση του κάθε ένα, ο χρόνο εξυπηρέτησης του και η γεωγραφική τους θέση, η χωρητικότητα των οχημάτων καθώς και η τιμή του μέγιστου επιτρεπόμενου μήκους μίας διαδρομής. Τα δεδομένα αυτά θα τα παραλάβουμε από ένα αρχείο κειμένου .txt και θα τα εισάγουμε στο προγραμματιστικό περιβάλλον Matlab. Κάθε γραμμή του αρχείου αντιστοιχεί σε μία παράμετρο του προβλήματος. Θεωρούμε ότι όπως απαριθμούμε τους πελάτες, ο πελάτης 1 είναι η αποθήκη, οπότε ο πρώτος πελάτης αντιστοιχεί στον αριθμό 2 και ούτε κάθε εξής. Αναλυτικό παράδειγμα παρουσιάζεται παρακάτω.



Δημιουργήσαμε μία συνάρτηση (readtxt.m) και σειριακά διαβάζουμε και αποθηκεύουμε τα δεδομένα σε αντίστοιχες μεταβλητές.

N: ο αριθμός πελατών

Q: η χωρητικότητα του οχήματος

MegistoMikos: μέγιστο μήκος διαδρομής

Service Time: χρόνος εξυπηρέτησης πελάτη

Demands: διάνυσμα με την ζήτηση κάθε πελάτη

coordinates: ο πίνακας (*N*, 2) με τις συντεταγμένες X, Y.

Στη συνέχεια δημιουργήσαμε πίνακα που περιέχει τις αποστάσεις μεταξύ όλων τον πελατών, με την βοήθεια του πίνακα *coordinates*. Ο πίνακας συμβολίζεται ως *Costs*(*i*,*j*), όπου *i* ο προηγούμενος πελάτης και *j* ο επόμενος. Η απόσταση που υπολογίζεται είναι ευκλείδεια και ο πίνακας είναι συμμετρικός, που σημαίνει ότι η απόσταση που χρειαζόμαστε για να πάμε από έναν πελάτη *i* στον επόμενο *j*, ισούται με την αντίστροφη από τον *j* στον *i*. Σημαντικό είναι να μην ληφθεί υπόψιν η απόσταση του πελάτη από τον εαυτό του και τα σημεία στον πίνακα *Costs*(*i*,*i*), η διαγώνιος, απειρίζονται. Ο πίνακας γεμίζει με την παρακάτω εξίσωση και στην συνέχεια γίνεται συμμετρικός.

$$Costs(i,j) = \sqrt{\left(coordinates(i,1) - coordinates(j,1)\right)^2 + (coordinates(i,2) - coordinates(j,2))^2}$$

Για λόγους διευκόλυνσης θεωρούμε ότι υπάρχει ισοδυναμία στις μονάδες χρόνου και απόστασης και ο χρόνος εξυπηρέτησης πελατών (ServiceTime) εκφράζεται σε μονάδες απόστασης, ώστε να συνυπολογιστεί με το μέγιστο μήκος διαδρομής (το οποίο θα μπορούσε να εκφραστεί σε μονάδες χρόνου).

Επιπλέον θεωρούμε ότι το μέγεθος του στόλου των οχημάτων που μπορούμε να χρησιμοποιήσουμε είναι άπειρο.

4.3 Μέθοδος Τυχαιοποιημένης Απληστίας για εύρεση αρχικής εφικτής λύσης

Αφού έχουμε αποθηκεύσει όλα τα δεδομένα του προβλήματος, με την μορφή που θα διευκολύνει την επίλυση του, συνέχεια έχει η αρχικοποίηση των μεταβλητών και η εύρεση αρχικής λύσης με τον αλγόριθμο απληστίας. Δηλώνουμε δηλαδή τις απαραίτητες μεταβλητές και την κατάσταση τους πριν ξεκινήσει η επίλυση του προβλήματος. Γνωρίζουμε ότι στην αρχή όλα τα φορτηγά βρίσκονται στην αποθήκη, δεν γνωρίζουμε εξ αρχής πόσα οχήματα θα χρησιμοποιήσουμε, αλλά μέσα από τον αλγόριθμο σχεδιάζουμε τις διαδρομές και αντιστοιχούμε ένα όχημα σε κάθε μία. Βήμα-βήμα, λοιπόν θα κατασκευάσουμε έναν πίνακα, τον *diadromi*, με γραμμές όσες και οι διαδρομές που θα σχεδιάσουμε, κάθε γραμμή θα έχει πρώτο στοιχείο το 1, που συμβολίζει την αποθήκη. Αρχικοποιούμε ως εξής:

diadromi=1: το πρώτο σημείο της διαδρομής είναι η αποθήκη.

diadromiN=1: μετρητής διαδρομών, βοηθητική μεταβλητή για να δώσουμε διάσταση στον πίνακα *diadromi*.

node=[2:*N*]: διάνυσμα που απαριθμούμε όλους τους πελάτες *N*, εκτός της αποθήκης.

trexwnQ=0: τρέχουσα χωρητικότητα, κάνουμε την παραδοχή ότι το φορτηγό ξεκινάει άδειο και γεμίζει σταδιακά ανάλογα με την ζήτηση των πελατών.

routeQ=0: συνολική χωρητικότητα διαδρομής.

trexwnCost=0: τρέχων κόστος, είναι η απόσταση που έχει διανυθεί μέχρι την δεδομένη στιγμή.

totalCost=0: συνολικό κόστος διαδρομής (συνολική απόσταση που έχει διανυθεί).

i=1: ο πελάτης στον οποίον βρισκόμαστε, αρχικά είναι η αποθήκη.

p=2: η θέση που θα πάρει ο επόμενος πελάτης στο διάνυσμα της διαδρομής, τώρα θα μπεί αμέσως μετά την αποθήκη.

k: μεταβλητή που ορίζει το μέγεθος της λίστας περιορισμού των υποψηφίων (RCL).

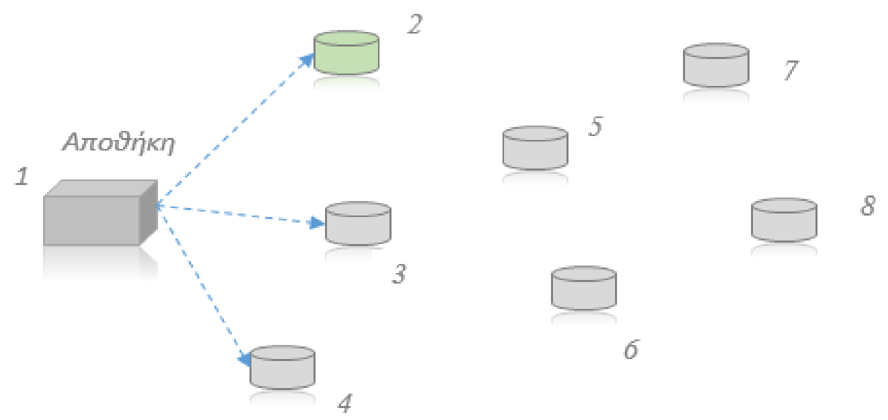
tmpCosts=*Costs*: βοηθητικός πίνακας που περιέχει τις αποστάσεις, ίδιος με τον πίνακα *Costs*.

4.3.3 Αρχική λύση για το Ανοιχτό Πρόβλημα Δρομολόγησης Οχημάτων(OVRP)

Ο αλγόριθμος απληστίας δουλεύει βηματικά, σε κάθε βήμα αναζητεί την καλύτερη τοπική λύση. Σε κάθε βήμα ο αλγόριθμος εξετάζει και έναν πελάτη, τον ελέγχει, τον συνυπολογίζει στην διαδρομή και αν ικανοποιούνται οι περιορισμοί τον κρατάει το διάνυσμα λύσης. Ο αλγόριθμος επαναλαμβάνεται μέχρι να έχουν εξυπηρετηθεί όλοι οι πελάτες, κάθε φορά που προστίθεται ένας πελάτης στην λύση τον αφαιρούμε από το διάνυσμα *node*. Χρησιμοποιούμε μία συνθήκη while, που εξασφαλίζει ότι ο αλγόριθμος θα επαναλαμβάνεται μέχρι το διάνυσμα των πελατών (*node*) να είναι κενό.

Πρώτα εξετάζουμε την αποθήκη και κατασκευάζουμε μια λίστα περιορισμού των υποψήφιων πελατών. Αρχικά ταξινομούμε όλες τους πελάτες ως προς την απόσταση τους από την αποθήκη. Έπειτα δημιουργούμε την λίστα περιορισμού διαλέγοντας και αποθηκεύοντας σε ξεχωριστό διάνυσμα, που ονομάσαμε *kontinoi*, τους *k* πιο κοντινούς πελάτες σε αυτή, όπου *k* μεταβλητή που δίνεται αυθαίρετα από τον χρήση.

Σειρά έχει να καθορίσουμε ποιος είναι ο πελάτης που θα εξεταστεί, λόγω τις τυχαιότητας του αλγορίθμου, διαλέγουμε έναν τυχαίο πελάτη από την λίστα περιορισμού, δηλαδή από το διάνυσμα *kontinoi*. Ο πελάτης που επιλέγεται τυχαία ονομάζεται *geitonas*. Με αυτόν τον τρόπο μπορεί να μην επιλεγεί ο πελάτης με που είναι πιο κοντά στην αποθήκη και φαινομενικά μπορεί να αποτελεί την βέλτιστη λύση με μικρότερο κόστος, αλλά κατά την εξέλιξη του αλγορίθμου μπορεί να οδηγηθούμε εν τέλει σε καλύτερη λύση. Σχεδιάζουμε ένα μικρό παράδειγμα με οκτώ πελάτες, ταξινομώντας τους πελάτες, αρχικά το διάνυσμα της διαδρομής περιέχει μόνο την αποθήκη **[1]**. Δημιουργούμε την λίστα περιορισμού υποψηφίων και κρατάμε τους *k* κοντινότερους, έστω το διάνυσμα της λίστας **[3 4 2]**, διαλέγουμε τυχαία έναν από αυτούς για γείτονα, έστω ο πελάτης 2.



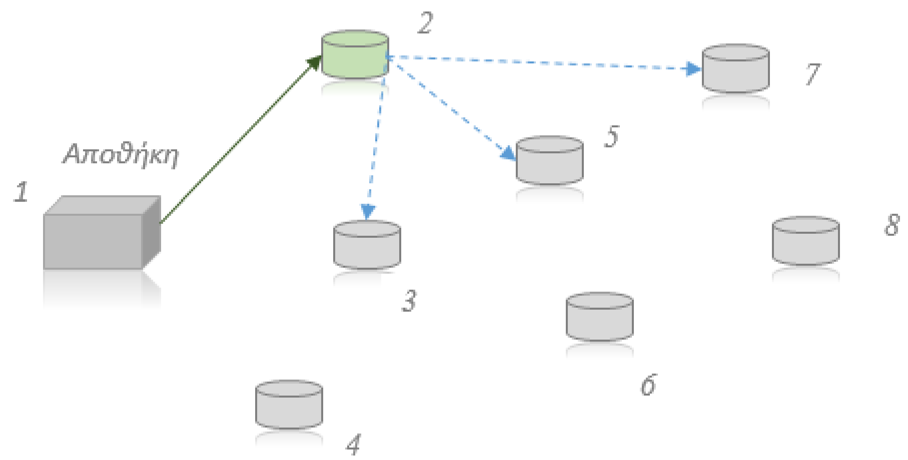
Στο σημείο αυτό πρέπει να εξετάσουμε αν ο πελάτης-γείτονας ικανοποιεί τους περιορισμούς του προβλήματος. Πρέπει να τηρούνται ταυτόχρονα οι παρακάτω συνθήκες, για κάθε γείτονα που δεν είναι η αποθήκη:

1. $trexwnQ + Demands\,(geitonas) \leq Q$,

όταν προστεθεί η ζήτηση του γείτονα στην μέχρι στιγμής συνολική ζήτηση, να μην ξεπερνάνε την χωρητικότητα του οχήματος.

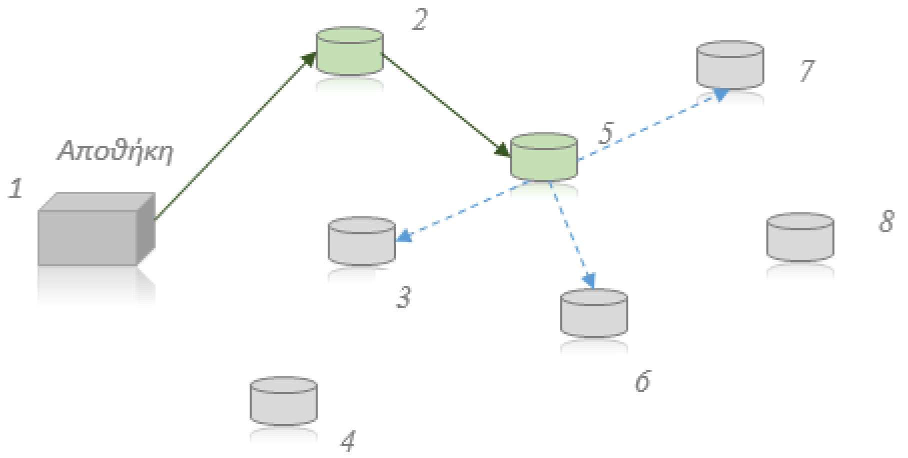
2. $trexwnCost + tmpCosts\,(i, geitonas) + ServiceTime \leq MegistoMikos$

όταν προστεθεί η απόσταση από τον πελάτη *i* (στον οποίο βρισκόμαστε ή την αποθήκη) μέχρι τον πελάτη geitonas και ο χρόνος εξυπηρέτησης, στην απόσταση που έχει διανυθεί μέχρι στιγμής, να μην υπερβαίνουν το μέγιστο μήκος διαδρομής.
Αν οι συνθήκες αυτές τηρούνται τότε ο geitonas μπαίνει στο διάνυσμα της διαδρομής στην θέση *p*, ενημερώνουμε τις μεταβλητές *trexwnQ* και *trexwnCost*, αυξάνουμε την θέση *p* κατά ένα και θέτουμε άπειρο το κόστος μεταβίβασης (*i*,*geitonas*), ώστε να μην υπολογιστεί πάλι ο πελάτης-γείτονας στην λίστα περιορισμού. Το διάνυσμα λύσης γίνεται **[1 2]** και ορίζουμε *i*=*geitonas*, ξεκινάμε δηλαδή την εύρεση του επόμενο από το τελευταίο πελάτη στην διαδρομή.

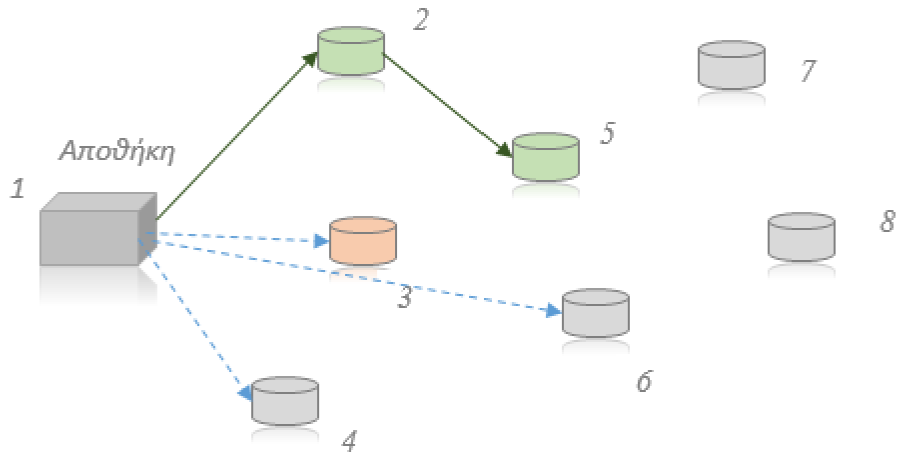


Τώρα η λίστα περιορισμού από τελείται από τους πελάτες **[5 3 7]** , ένα επιλεχτεί τυχαία ο 5, θέτουμε *geitonas*=5 και ελέγχουμε πάλι τους περιορισμούς, εάν ικανοποιούνται τον δεχόμαστε και το διάνυσμα της διαδρομής γίνεται **[1 2 5]** .

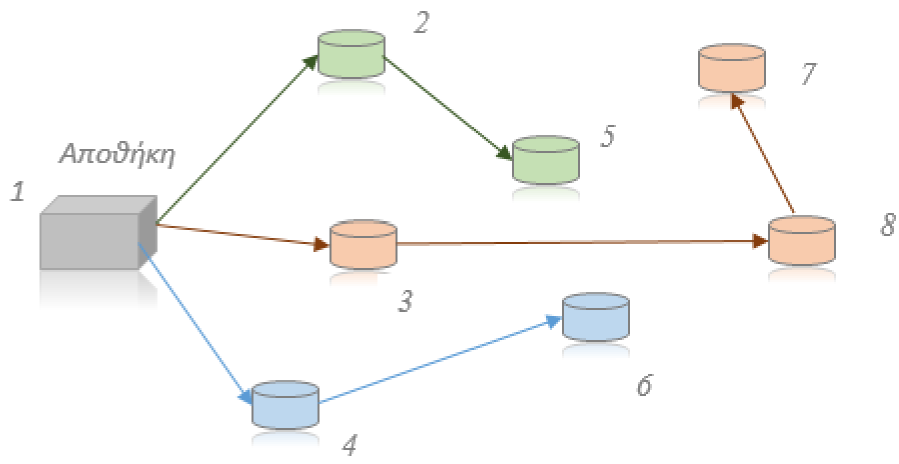
Επαναλαμβάνουμε την διαδικασία από τον πελάτη 5, με λίστα περιορισμού **[3 6 7]**.



Έστω ότι επιλέγεται για γείτονας ο 6, όμως παραβιάζεται ένας περιορισμός. Στον βοηθητικό πίνακα *tmpCosts* απειρίζουμε την γραμμή και την στήλη του πελάτη 5 ώστε να μην εξεταστεί ξανά. Οι μεταβλητές μας γυρίζουν στις αρχικές τους τιμές και διαδρομή μας ολοκληρώνεται. Ξεκινάει νέα διαδρομή από την αποθήκη, ακολουθώντας την ίδια διαδικασία, η λύση μας τώρα θα έχει την μορφή πίνακα $\begin{matrix} 1 & 2 & 5 \\ 1 & 0 & 0 \end{matrix}$.



Η αρχική λύση ολοκληρώνεται όταν έχουν εξεταστεί όλοι οι πελάτες και έχουν τοποθετηθεί στον πίνακα *diadromi*. Στο παράδειγμα μας η λύση μας διαμορφώνεται ως εξής: $\begin{matrix} 1 & 2 & 5 & 0 \\ 1 & 4 & 6 & 0 \\ 1 & 3 & 8 & 7 \end{matrix}$.



4.3.4 Αρχική λύση για το Πρόβλημα Δρομολόγησης Οχημάτων Περιορισμένης Χωρητικότητας (CVRP)

Γνωρίζουμε ότι το πρόβλημα δρομολόγησης οχημάτων περιορισμένης χωρητικότητας επιβάλει ότι το όχημα όταν ολοκληρώσει την διαδρομή του σύμφωνα με τον αλγόριθμο απληστίας, επιστρέφει στην αποθήκη. Η αρχική λύση διαμορφώνεται όπως στην προηγούμενη παράγραφο τροποποιώντας τον περιορισμό της μέγιστης διαδρομής ως εξής:

$$trexwnCost + tmpCosts\,(i, geitonas) + ServiceTime + tmpCosts(geitonas,1) \leq MegistoMikos$$

Προσθέτουμε δηλαδή την απόσταση του πελάτη-γείτονα από την αποθήκη, διασφαλίζουμε έτσι ότι αν ο *geitonas* γίνει ο τελευταίος πελάτης της διαδρομής θα μπορεί να επιστρέψει στην αποθήκη.

4.4 Μέθοδος Τοπικής Αναζήτησης για βελτίωση αρχικής λύσης

Από τις μεθόδους τοπικής αναζήτησης που αναφέρθηκαν στο προηγούμενο κεφάλαιο θα χρησιμοποιήσουμε την μέθοδο 2-opt. Η διαδικασία εφαρμόζεται σε κάθε γραμμή του πίνακα λύσης, άρα σε κάθε διαδρομή. Αρχικά δημιουργούμε βοηθητικό πίνακα ίδιων διαστάσεων με τον πίνακα λύσης, που όμως περιέχει τις αποστάσεις μετάβασης από έναν πελάτη στον αμέσως επόμενο του, πίνακας *Apostaseis*. Αθροίζοντας τα στοιχεία σε κάθε γραμμή αυτού του πίνακα, υπολογίζουμε το συνολικό κόστος κάθε διαδρομής, *totalCost*. Σε κάθε γραμμή υπολογίζουμε από τον πίνακα με τις αποστάσεις, τον ακριβότερο πελάτη, αυτόν δηλαδή που εμφανίζει την μεγαλύτερη απόσταση από τον προηγούμενο του και διαλέγουμε και έναν ακόμα τυχαίο (ο οποίος δεν είναι η αποθήκη). Κόβουμε το διάνυσμα της γραμμής σε αυτά τα δύο σημεία, κρατάμε το τμήμα που δημιουργείτε, το αντιστρέφουμε και το επανασυνδέουμε στο σημείο που είναι κενό το διάνυσμα. Ακολουθεί σχηματικό παράδειγμα για τυχαία διαδρομή με ακριβότερο πελάτη τον 3 και τυχαία επιλογή τον 7.

diadromi(*i*):

1	*2*	*3*	*4*	*5*	*6*	*7*	*8*

Apostaseis (*i*):

0	**10**	**35**	**12**	**22**	**16**	**28**	**14**

Νέα *diadromi*(*i*):

1	**2**	**7**	**6**	**5**	**4**	**3**	**8**

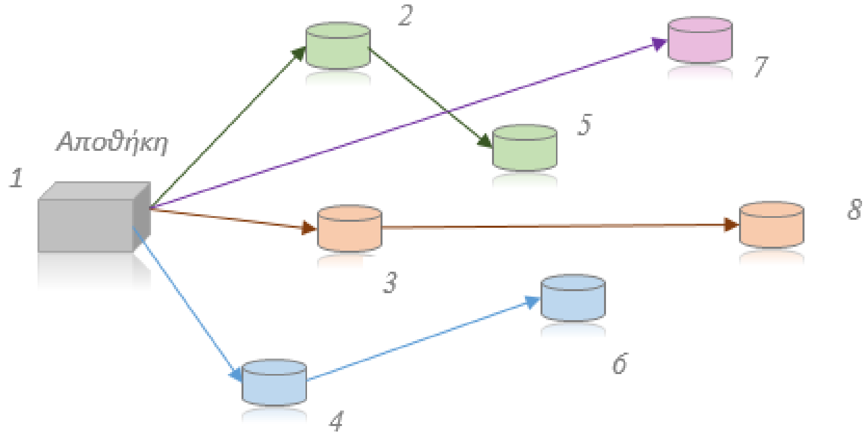
Η νέα διαδρομή που παράγεται είναι σίγουρά κομμάτι εφικτής λύσης, αφού οι πελάτες παραμένουν ίδιοι, άρα και η συνολική τους ζήτηση. Ακόμα, αφού θα αντικαταστήσουμε την πιο ακριβή μετάβαση με μία άλλη τυχαία, είναι αρκετά πιθανό να οδηγηθούμε σε μετάβαση με μικρότερο κόστος, οπότε δεν θα παραβιάζεται και η συνθήκη του μέγιστου μήκους διαδρομής. Υπολογίζουμε το κόστος της νέας διαδρομής και αν είναι καλύτερο από το κόστος της αρχικής, αντικαθιστούμε το νέο διάνυσμα στην λύση. Εάν το κόστος δεν βελτιώνεται παραμένουμε στην αρχική λύση.

Η διαδικασία αυτή επαναλαμβάνεται για κάποιο αριθμό επαναλήψεων, που ορίζουμε εμείς. Οι περισσότερες επαναλήψεις αυξάνουν τις πιθανότητες για καλύτερο αποτέλεσμα. Τα αποτελέσματα της κάθε επανάληψης του 2-opt αλγόριθμου πάνω στην αρχική εφικτή λύση, αποθηκεύονται σε έναν πίνακα, υπολογίζεται το συνολικό κόστος της κάθε λύσης, αθροίζοντας τα *totalCost*, συγκρίνονται μεταξύ τους και κρατάμε την λύση με το ελάχιστο κόστος. Έχουμε πάρει δηλαδή από κάθε επανάληψη έναν πίνακα diadromi που περιέχει τις βελτιωμένες διαδρομές της αρχικής λύσης και από αυτούς κρατάμε τον πίνακα που αντιστοιχεί στη βέλτιστη λύση.

4.5 Μετα-βελτιστοποίηση της λύσης

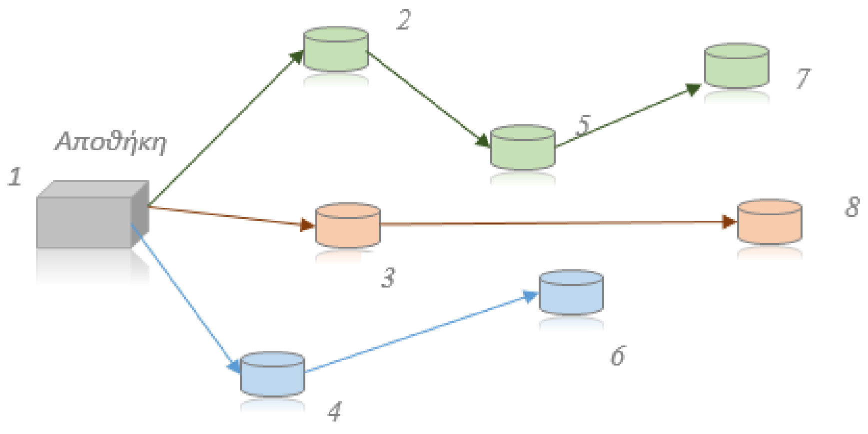
Προκειμένου να βελτιώσουμε περισσότερο την λύση μας, θα χρησιμοποιήσουμε τον αλγόριθμο τοπικής αναζήτησης 1-0 επανατοποθέτηση (1-0 relocate). Σκοπός μας είναι να μειώσουμε τον αριθμό των διαδρομών, για να ελαχιστοποιήσουμε το συνολικό κόστος. Αρχικά χρησιμοποιούμε ένα βοηθητικό διάνυσμα στο οποίο έχουμε υπολογίσει τον αριθμό των πελατών κάθε διαδρομής και βρίσκουμε ποια διαδρομή έχει τους λιγότερους πελάτες. Από αυτή την διαδρομή θα πάρουμε τους πελάτες της, ξεκινώντας από τον τελευταίο, έναν-έναν διαδοχικά να τους τοποθετήσουμε σε άλλες διαδρομές. Στο σχεδιασμένο παράδειγμα, ο πίνακας της λύσης είναι: $\begin{matrix} 1 & 2 & 5 \\ 1 & 7 & 0 \\ 1 & 4 & 6 \\ 1 & 3 & 8 \end{matrix}$

Άρα, θα μετακινήσουμε τον πελάτη 7 και θα τον εντάξουμε στο τέλος κάποιας άλλη



διαδρομής.

Θέλει ιδιαίτερη προσοχή η μετακίνηση του πελάτη, γιατί πρέπει να επιτευχθεί χωρίς να παραβιαστούν οι περιορισμοί. Για να το πετύχουμε αυτό, υπολογίζουμε για όλες τις διαδρομές, το περιθώριο που έχει η κάθε μία πάνω στο μέγιστο μήκος διαδρομής και στην χωρητικότητα του οχήματος. Σε έναν βοηθητικό πίνακα υπολογίζουμε την απόσταση του πελάτη προς μετακίνηση, από όλους τους τελευταίους πελάτες των υπόλοιπων διαδρομών. Διαλέγουμε λοιπόν να τοποθετήσουμε τον υποψήφιο προς μετακίνηση δίπλα στον τελευταίο πελάτη της διαδρομής, που απέχουν λιγότερο και δεν παραβιάζονται οι περιορισμοί όταν συνυπολογίσουμε στο σύνολο την απόσταση αυτή, τον χρόνο εξυπηρέτησης και την ζήτηση του. Στο παράδειγμα μας, ο πελάτης 7 τοποθετήθηκε στην πράσινη διαδρομή, αφού ο τελευταίος πλησιέστερος πελάτης άλλης διαδρομής είναι ο 5, δεχόμαστε αυθαίρετα ότι δεν παραβιάζονται οι περιορισμοί και ο πίνακας της λύσης διαμορφώνεται: $\begin{matrix} 1 & 2 & 5 & 7 \\ 1 & 4 & 6 & 0 \\ 1 & 3 & 8 & 0 \end{matrix}$. Αν μία διαδρομή αδειάσει από πελάτες και έχει μόνο την αποθήκη, την σβήνουμε από τον πίνακα λύσης. Αρχικοποιούμε όλες τις απαραίτητες μεταβλητές και ξεκινάμε την διαδικασία από μία διαφορετική διαδρομή που θα έχει πλέον τους λιγότερους πελάτες. Ο αλγόριθμος θα σταματήσει να τροποποιεί τον πίνακα λύσης, όταν κάποιος πελάτης υποψήφιος προς μετακίνηση δεν θα μπορεί να μετατοπιστεί σε άλλη διαδρομή, γιατί θα παραβιάζει τους περιορισμούς.



4.6 Εύρεση τελικής βέλτιστης λύσης και ψευδοκώδικας

Ολόκληρος ο αλγόριθμος της τυχαιοποιημένης προσαρμοστικής αναζήτησης (GRASP) από την φάση κατασκευής μέχρι και την μετά-βελτιστοποίηση της λύσης, επαναλαμβάνεται για συγκεκριμένο αριθμό επαναλήψεων, τον οποίο ορίζουμε εμείς.
Σε κάθε επανάληψη, αφού βασιζόμαστε στην τυχαιότητα, θα κατασκευαστεί διαφορετική αρχική λύση. Οι διαφορετικές εναρκτήριες λύσεις αυξάνουν την πιθανότητα του αλγορίθμου να αποφύγει κάποιο τοπικό βέλτιστο και να οδηγηθούμε μέσα από τις διαδικασίες βελτιστοποίησης σε ολικό βέλτιστο. Έτσι κάθε βέλτιστη λύση ανά επανάληψη, αποθηκεύεται και μετά το πέρας των επαναλήψεων συγκρίνονται τα συνολικά τους κόστη και κρατάμε σαν τελική βέλτιστη λύση την πιο οικονομική.
Ακολουθεί ο ψευδοκώδικας του αλγορίθμου που κατασκευάσαμε :

ΑΛΓΟΡΙΘΜΟΣ GRASP

Εισαγωγή των δεδομένων του προβλήματος

Για έναν συγκεκριμένο αριθμό επαναλήψεων

Δημιουργία εφικτής λύσης

Αρχικοποίηση μεταβλητών και πινάκων

Μέχρι να εξεταστούν όλοι οι κόμβοι

Εφαρμογή του αλγορίθμου απληστίας

Δημιουργία Λίστας Περιορισμού

Επιλογή πελάτη από Λίστα Περιορισμού

Αν ο πελάτης ικανοποιεί τους περιορισμούς μέγιστου μήκους και χωρητικότητας

Τοποθετούμε τον πελάτη στην διαδρομή

Ενημερώνουμε τις μεταβλητές

Αλλιώς

Η διαδρομή ολοκληρώνεται στον προηγούμενο πελάτη

Δημιουργούμε νέα διαδρομή και ξεκινάμε από την αποθήκη

Αρχικοποιούμε τις μεταβλητές

Τέλος Αν

Τέλος Μέχρι

Υπολογισμός κόστους εφικτής λύσης

Εφαρμογή του αλγορίθμου τοπικής αναζήτησης

Για έναν συγκεκριμένο αριθμό επαναλήψεων

Εύρεση πελάτη με ακριβότερη μετάβαση

Επιλογή τυχαίου πελάτη

Επιλογή τμήματος διαδρομής με άκρα αυτούς τους πελάτες

Αντιστροφή τμήματος διαδρομής και επανένωση

Υπολογισμός κόστους νέας διαδρομής

Αν το νέο κόστος είναι μικρότερο από το αρχικό

Κρατάμε την νέα διαδρομή στην λύση

Αλλιώς

Κρατάμε την αρχική διαδρομή

Τέλος Αν

Αποθήκευση λύσης για κάθε επανάληψη

Υπολογισμός συνολικού κόστους λύσης

Τέλος Για

Επιλογή λύσης με βέλτιστο κόστος

Μείωση αριθμού διαδρομών

Μέχρι συνθήκη τερματισμού

Υπολογισμός αριθμών κόμβων κάθε διαδρομής

Επιλογή διαδρομής με ελάχιστο αριθμό κόμβων

Επιλογή τελευταίου πελάτη της διαδρομής

Υπολογισμός του περιθώριού των διαδρομών πάνω στο μέγιστο μήκος και στη χωρητικότητα

Υπολογισμός του κόστους μετάβασης από τον πελάτη αυτό , στους τελευταίους πελάτες των υπόλοιπων διάδρομών.

Εύρεση ελάχιστου κόστους μετάβασης και αντίστοιχης διαδρομής.

Αν το κόστος της μετάβασης και ο χρόνος εξυπηρέτησης είναι μικρότερα από το περιθώριο και αν η συνολική ζήτηση δεν παραβιάζεται

Πραγματοποιείται η μετακίνηση του πελάτη

Ενημερώνονται όλες οι μεταβλητές

Τέλος Αν

Αν η διαδρομή περιέχει μόνο την αποθήκη

Η διαδρομή διαγράφεται

Αλλάζουν οι διαστάσεις του πίνακα λύσεις

Αρχικοποιούμε τις μεταβλητές

Τέλος Αν

Αν ο πελάτης δεν μπορεί να μεταβιβαστεί σε άλλη διαδρομή

Ενεργοποίηση συνθήκης τερματισμού

Τέλος Αν

Τέλος Μέχρι

Αποθηκεύουμε την βέλτιστη λύση κάθε επανάληψης και το αντίστοιχο κόστος

Τέλος Για

Εύρεση τελικής βέλτιστης λύσης με το ελάχιστο κόστος

Κεφάλαιο 5: Υπολογιστικά αποτελέσματα

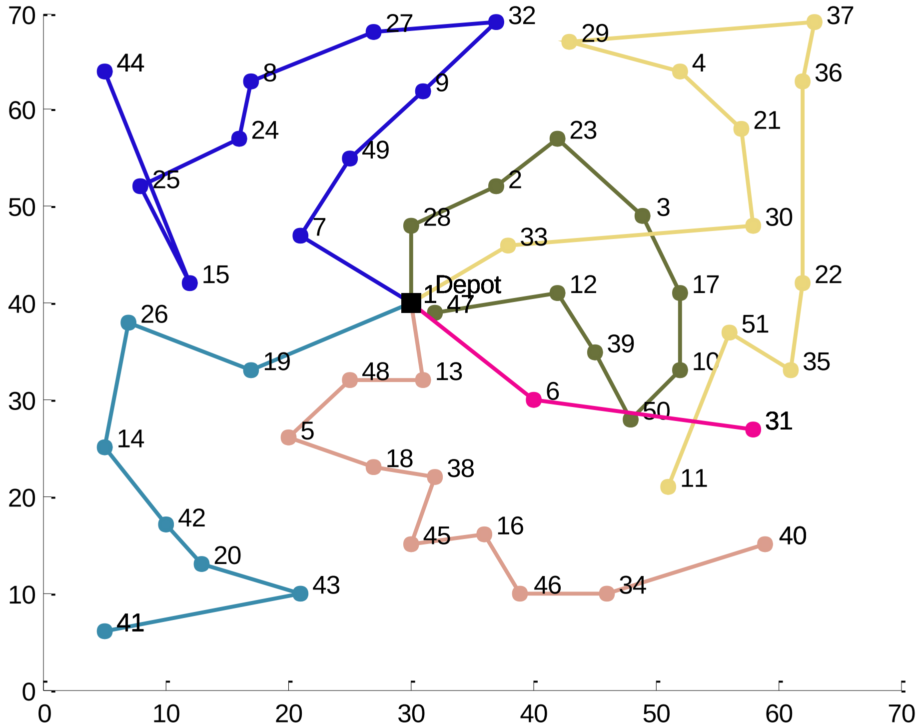
5.1 Περιγραφή και αναπαράσταση αποτελεσμάτων για το Ανοιχτό Πρόβλημα Δρομολόγησης Οχημάτων (OVRP)

Με σκοπό να αξιολογήσουμε τα αποτελέσματα του αλγορίθμου τρέξαμε μία σειρά από προτυποποιημένα προβλήματα αναφοράς (benchmark problems). Για το ανοιχτό πρόβλημα δρομολόγησης οχημάτων χρησιμοποιήσαμε δύο κατηγορίες προβλημάτων, τα 14 προβλήματα αναφοράς που έχουν προταθεί από τον Christoforides [31] και ακόμα 8 μεγάλης κλίμακας που έχουν προταθεί από τους Li, Golden και Wasil [32]. Κάθε πρόβλημα της πρώτης κατηγορίας, περιέχει από 51 μέχρι 200 κόμβους, μαζί με την αποθήκη. Η τοποθεσία των κόμβων καθορίζεται από τις καρτεσιανές τους συντεταγμένες. Κάθε πρόβλημα έχει περιορισμένη χωρητικότητα στο όχημα, ενώ κάποια περιέχουν μέγιστο μήκος επιτρεπόμενης διαδρομής και μη μηδενικό χρόνο εξυπηρέτησης. Το δεύτερο σετ των προβλημάτων περιέχει από 200 μέχρι 480 κόμβους, μαζί με την αποθήκη και περιορισμό χωρητικότητας. Αναλυτικά εμφανίζονται τα χαρακτηριστικά τους στον πίνακα:

Προβλήματα Christoforides				
Πρόβλημα	**Πελάτες**	**Χωρητικότητα**	**Μέγιστο Μήκος Διαδρομής**	**Service Time**
C1	51	160	∞	0
C2	76	140	∞	0
C3	101	200	∞	0
C4	151	200	∞	0
C5	200	200	∞	0
C6	51	160	180	10
C7	76	140	144	10
C8	101	200	207	10
C9	151	200	180	10
C10	200	200	180	10
C11	121	200	∞	0
C12	101	200	∞	0
C13	121	200	648	50
C14	101	200	936	90

Προβλήματα Li et.al.		
Πρόβλημα	**Πελάτες**	**Χωρητικότητα**
O1	201	900
O2	241	550
O3	281	900
O4	321	700
O5	361	900
O6	401	900
O7	441	900
O8	481	1000

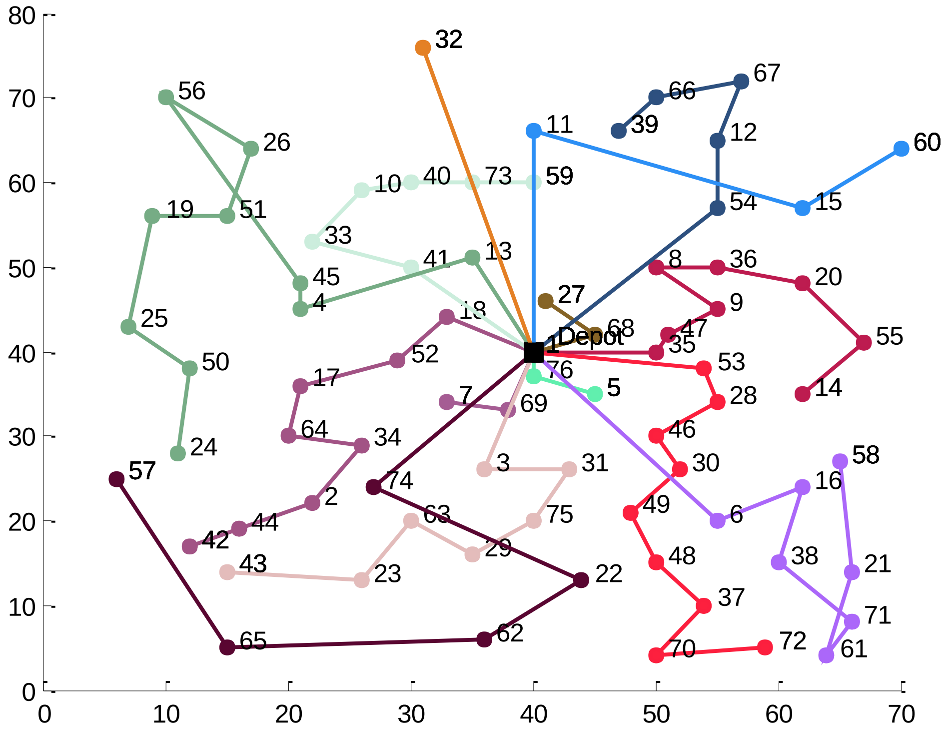
5.1.1 Αποτελέσματα προβλήματος C1



Συνολική απόσταση : 514,018

1	28	2	23	3	17	10	50	39	12	47
1	13	48	5	18	38	45	16	46	34	40
1	7	49	9	32	27	8	24	25	15	44
1	33	30	21	4	29	37	36	22	35	51
1	19	26	14	42	20	43	41			
1	6	31								

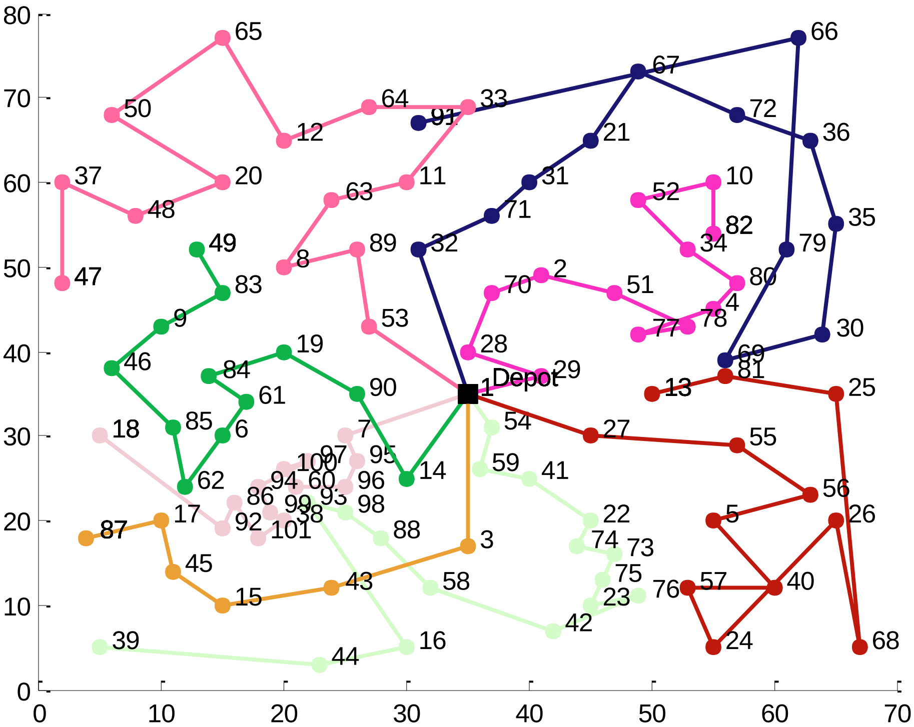
5.1.2 Αποτελέσματα προβλήματος C2



Συνολική απόσταση : 738,014

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

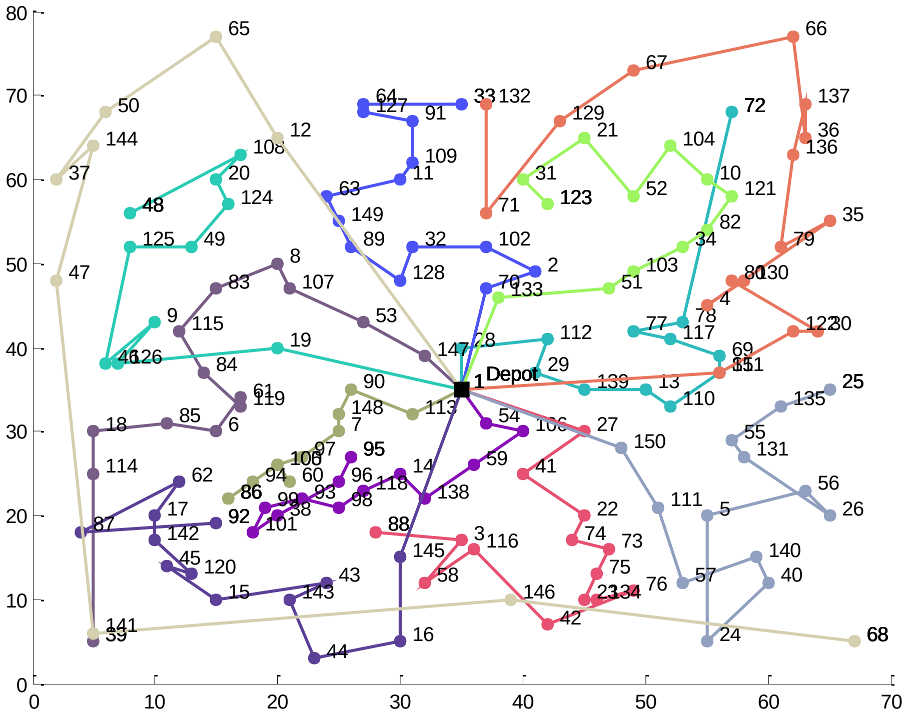
5.1.3 Αποτελέσματα προβλήματος C3



Συνολική απόσταση : 838,765

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

5.1.4 Αποτελέσματα προβλήματος C4

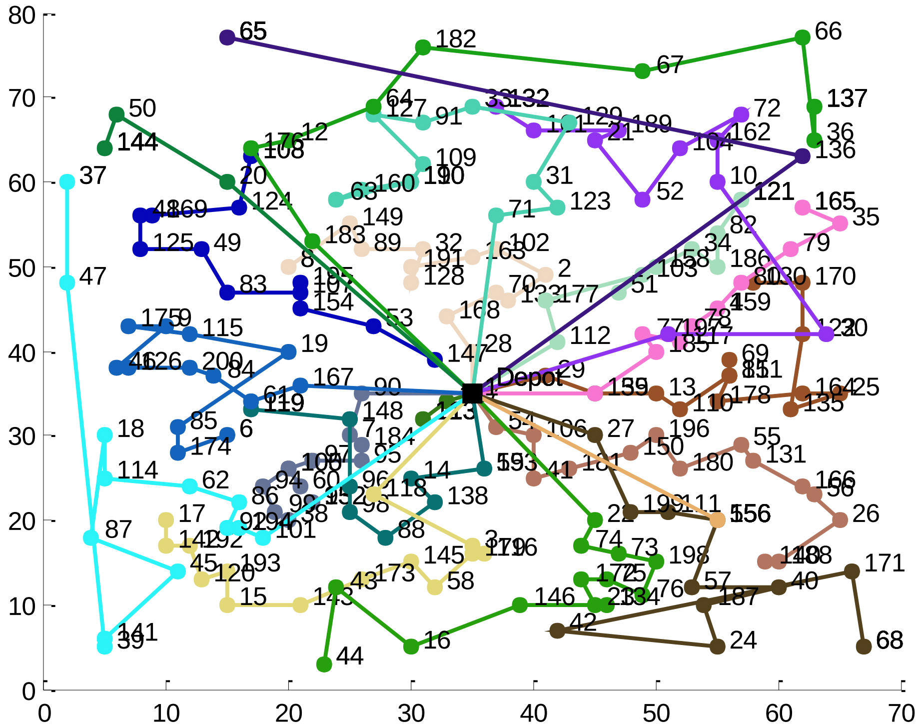


Συνολική απόσταση : 1.082,86

1	28	112	29	139	13	110	81	69	117	77	78	72				
1	113	90	148	7	97	105	60	100	94	86						
1	147	53	107	8	83	115	84	119	61	6	85	18	114	39		
1	54	106	59	138	14	118	98	93	99	101	38	96	95			
1	27	41	22	74	73	75	23	134	76	42	116	58	3	88		
1	70	2	102	32	128	89	149	63	11	109	91	127	64	33		
1	150	111	57	140	40	24	5	56	26	131	55	135	25			
1	133	51	103	34	82	121	10	104	52	21	31	123				
1	19	126	9	46	125	49	124	20	108	48						
1	145	16	44	143	43	15	45	120	142	17	62	87	92			
1	151	122	30	80	130	4	35	79	136	137	36	66	67	129	71	132

1	12	65	50	37	144	47	141	146	68

5.1.5 Αποτελέσματα προβλήματος C5

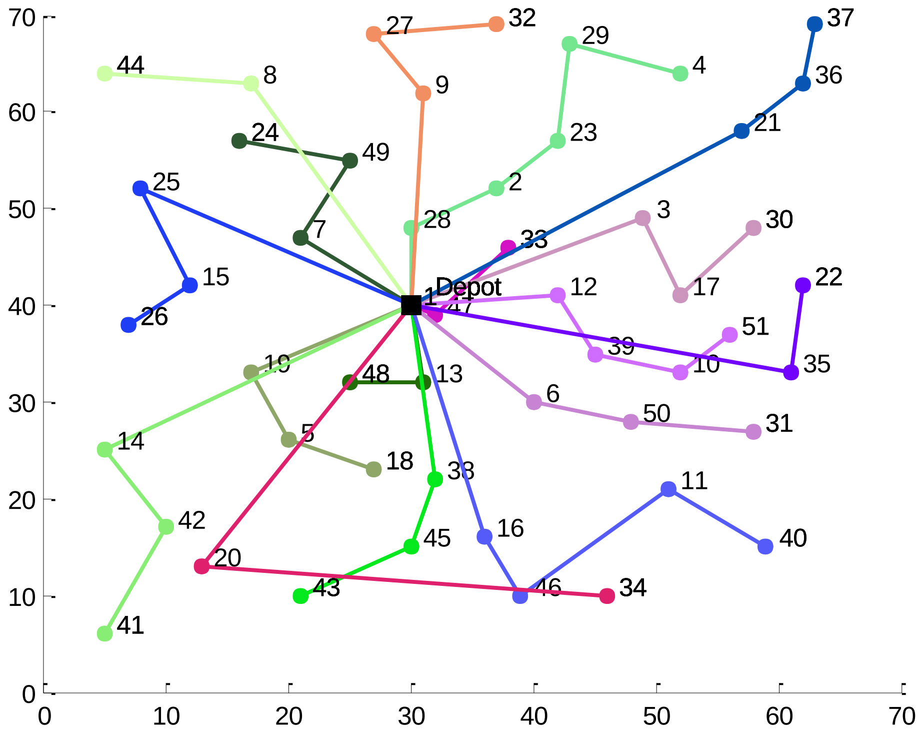


Συνολική απόσταση : 1.299,727

1	54	106	41	181	150	196	180	55	131	56	166	26	188	140
1	28	168	70	133	2	102	163	191	128	32	89	149	8	
1	157	113												
1	29	139	13	110	81	69	151	178	164	25	135	122	170	130
1	90	7	184	95	97	100	60	105	94	99	38	93	152	
1	147	53	154	195	107	83	49	125	48	169	124	108		
1	153	59	14	138	88	98	96	148	119					
1	112	177	103	51	158	34	186	82	121					
1	155	185	77	117	78	4	159	80	79	35	165			
1	167	61	84	200	126	46	9	175	115	19	85	174	6	
1	118	3	116	179	58	145	173	143	15	193	120	192	142	17
1	197	30	10	162	72	104	52	21	189	161	132			
1	22	74	73	198	76	75	172	23	134	146	16	43	44	

1	27	199	111	5	57	40	187	24	42	171	68		
1	71	123	31	129	33	91	127	109	11	160	63	190	
1	183	176	12	64	182	67	66	36	137				
1	101	194	92	86	62	114	18	87	45	141	39	47	37
1	20	50	144										
1	136	65											
1	156												

5.1.6 Αποτελέσματα προβλήματος C6

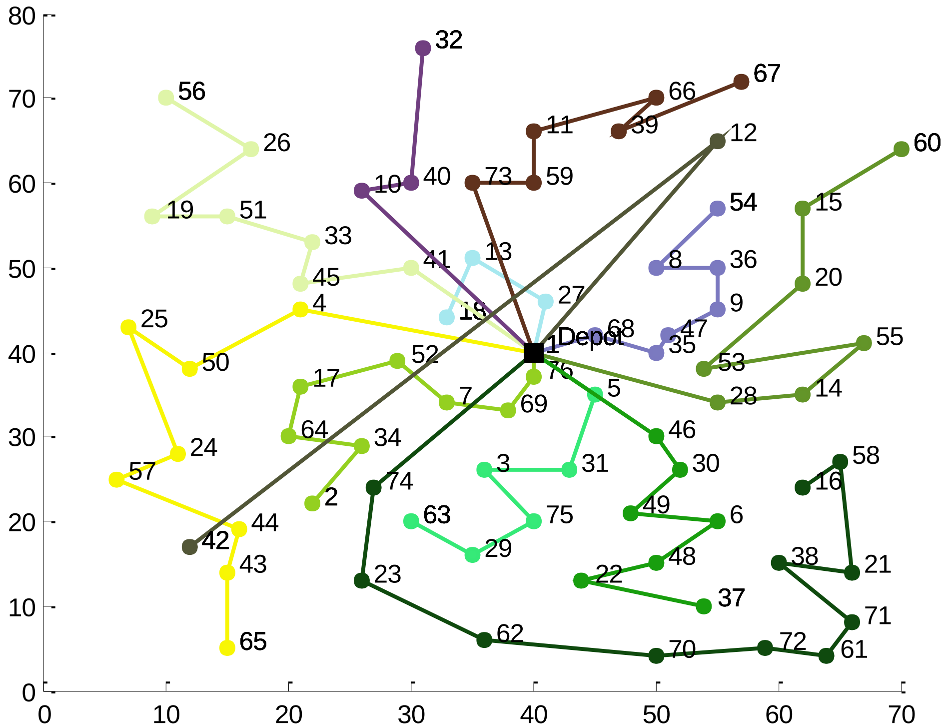


Συνολική απόσταση : 646,206

1	28	2	23	29	4
1	13	48			
1	47	33			
1	7	49	24		
1	12	39	10	51	
1	19	5	18		
1	25	15	26		
1	6	50	31		
1	3	17	30		
1	38	45	43		
1	16	46	11	40	

1	9	27	32
1	8	44	
1	14	42	41
1	20	34	
1	35	22	
1	21	36	37

5.1.7 Αποτελέσματα προβλήματος C7

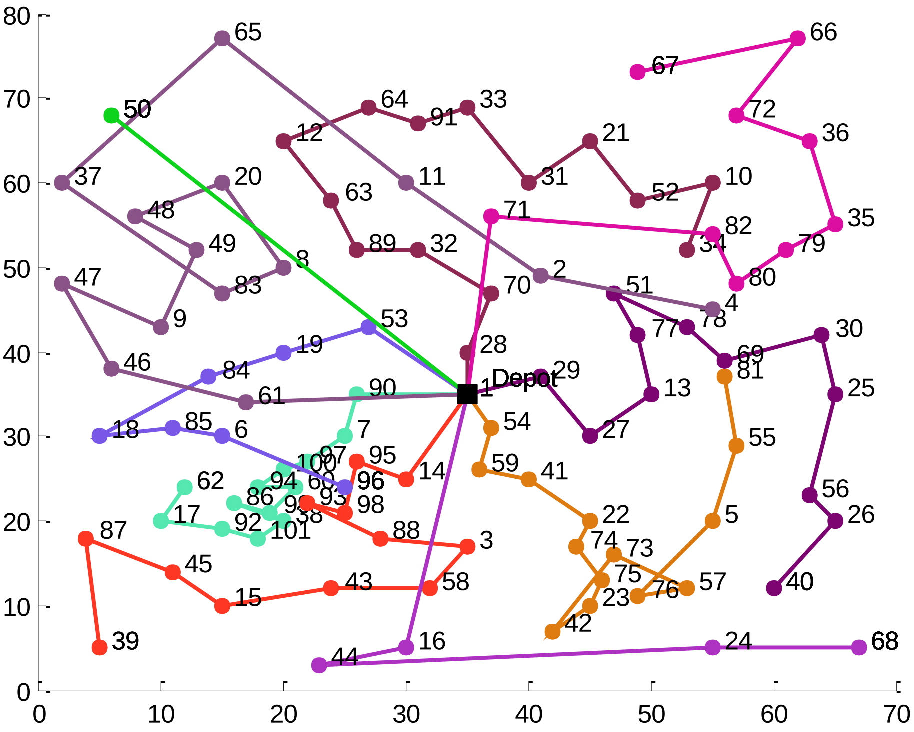


Συνολική απόσταση : 735,041

1	76	69	7	52	17	64	34	2
1	27	13	18					
1	68	35	47	9	36	8	54	

1	5	31	3	75	29	63						
1	41	45	33	51	19	26	56					
1	46	30	49	6	48	22	37					
1	28	14	55	53	20	15	60					
1	4	50	25	24	57	44	43	65				
1	73	59	11	66	39	67						
1	10	40	32	32								
1	74	23	62	70	72	61	71	38	21	58	16	

5.1.8 Αποτελέσματα προβλήματος C8

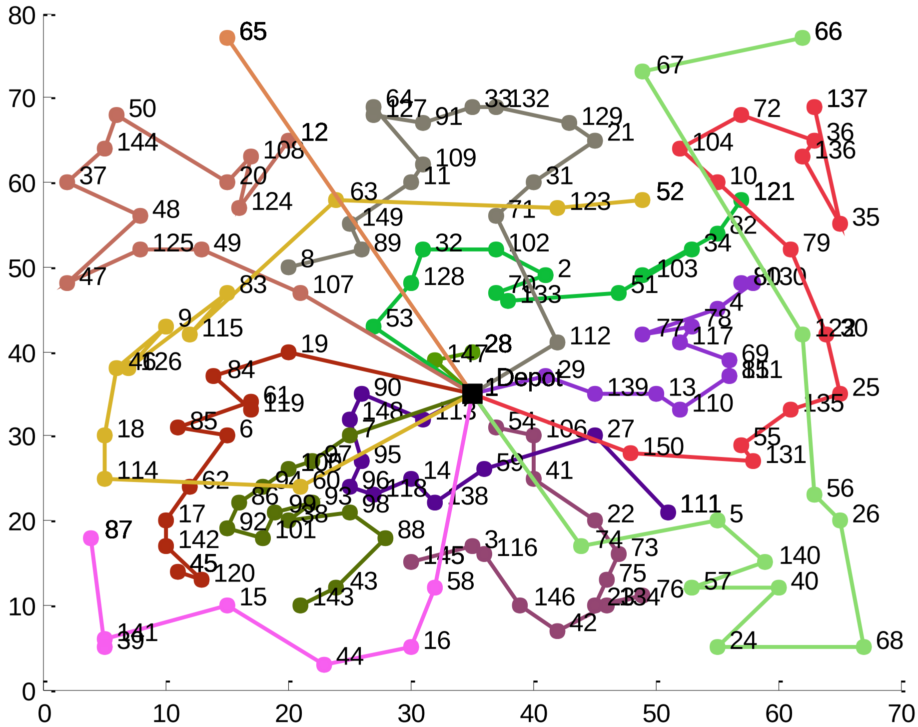


1	28	70	32	89	63	12	64	91	33	31	21	52	10	34
1	54	59	41	22	74	75	23	42	73	57	76	5	55	81
1	90	7	97	100	94	60	99	86	38	101	92	17	62	
1	14	95	98	93	88	3	58	43	15	45	87	39		

1	29	27	13	77	51	78	69	30	25	56	26	40		
1	53	19	84	18	85	6	96							
1	61	46	47	9	49	48	20	8	83	37	65	11	2	4
1	71	82	80	79	35	36	72	66	67					
1	16	44	24	68										
1	50													

Συνολική απόσταση : 876,217

5.1.9 Αποτελέσματα προβλήματος C9

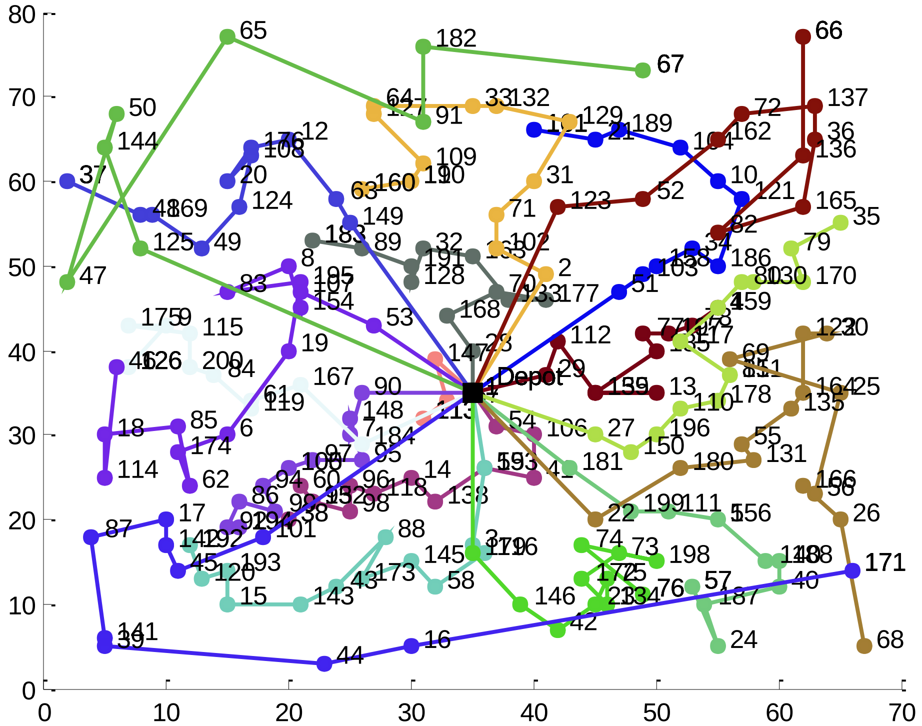


Συνολική απόσταση : 1.098,797

1	54	106	41	22	73	75	23	76	134	42	146	116	3	145	
1	147	28													
1	29	139	13	110	81	151	69	117	78	77	4	130	80		
1	113	90	148	95	96	118	14	138	59	27	111				
1	7	97	100	105	94	86	92	101	99	93	38	98	88	43	143
1	53	128	32	102	2	70	133	51	34	103	82	121			
1	150	131	55	135	25	30	79	10	104	72	36	136	35	137	
1	19	84	119	61	85	6	62	17	142	120	45				
1	112	71	31	21	129	132	33	91	127	64	109	11	149	89	8
1	107	49	125	47	48	37	144	50	20	108	124	12			
1	60	114	18	46	9	126	83	115	63	123	52				

1	74	5	140	57	40	24	68	26	56	122	67	66
1	58	16	44	15	141	39	87					
1	65											

5.1.10 Αποτελέσματα προβλήματος C10

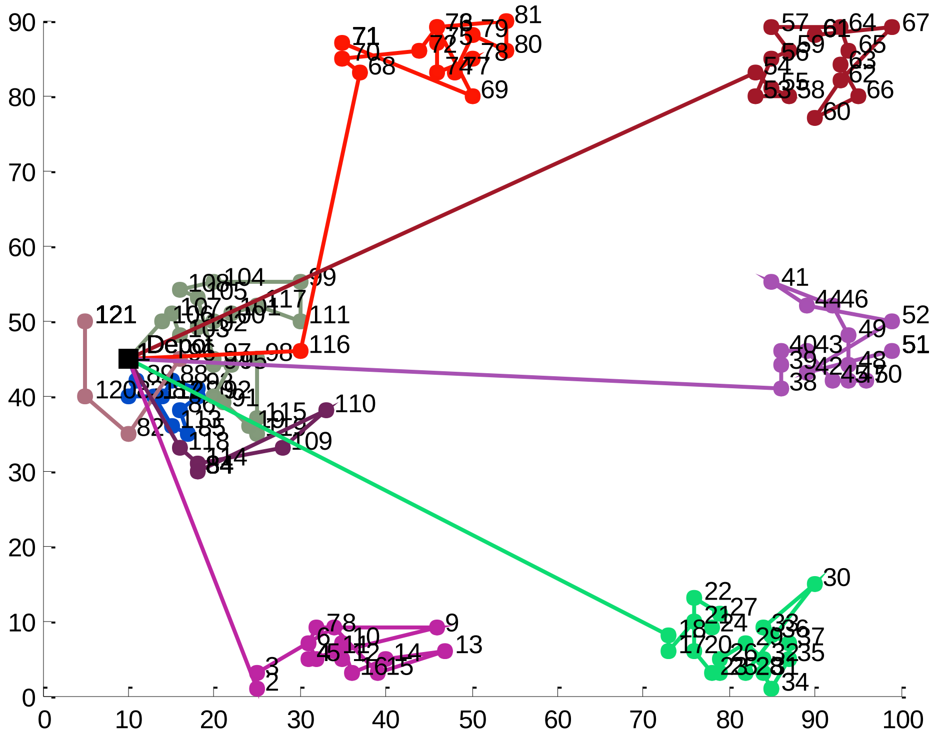


Συνολική απόσταση : 1.241,65

1	54	106	41	59	138	14	118	96	98	152	60	93	38	
1	28	168	70	177	133	163	32	128	191	89	183			
1	147	157	113											
1	29	112	139	13	155	185	77	197	78	4				
1	90	7	148	95	97	105	100	94	99	86	92	194		
1	153	3	116	58	145	173	88	43	143	15	193	120	192	
1	184	167	61	119	84	200	115	175	9	126				
1	53	107	154	8	83	195	19	6	174	62	85	18	114	46

1	27	150	196	110	178	151	81	117	159	80	130	170	79	35
1	181	199	111	156	5	140	188	40	187	24	57			
1	51	158	103	34	186	121	10	104	189	21	161			
1	2	102	71	31	129	132	33	64	127	109	190	11	160	
1	179	146	42	23	75	134	172	73	198	74	76			
1	22	180	131	55	135	164	122	30	69	25	56	166	26	68
1	123	52	162	72	137	36	165	82	136	66				
1	101	45	142	17	87	141	39	44	16	171				
1	149	63	12	176	20	108	124	49	169	48	37			
1	125	144	50	47	65	91	182	67						

5.1.11 Αποτελέσματα προβλήματος C11

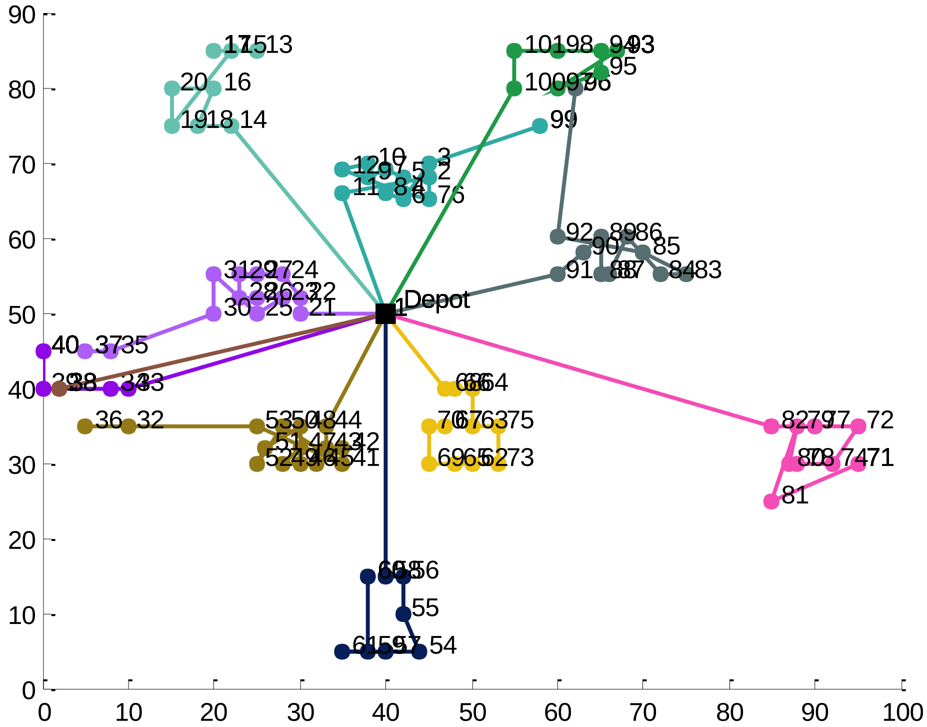


Συνολική απόσταση : 906,768

1	89	83	87	88	93	90	86	85	113	112					
1	106	107	103	102	100	101	117	111	99	104	108	105	97	94	95
92	91	19	119	115	98										
1	96	82	120	121											
1	118	114	109	110	84										

1	2	3	8	15	13	14	16	10	12	11	5	6	7	9	4
1	116	68	70	72	73	75	74	78	77	79	80	81	76	69	71
1	18	17	21	24	27	22	20	23	25	26	29	32	31	34	35
37	36	33	30	28											
1	38	39	40	43	45	50	47	48	49	46	41	44	52	42	51
1	54	55	58	53	56	59	57	64	65	63	66	60	62	67	61

5.1.12 Αποτελέσματα προβλήματος C12

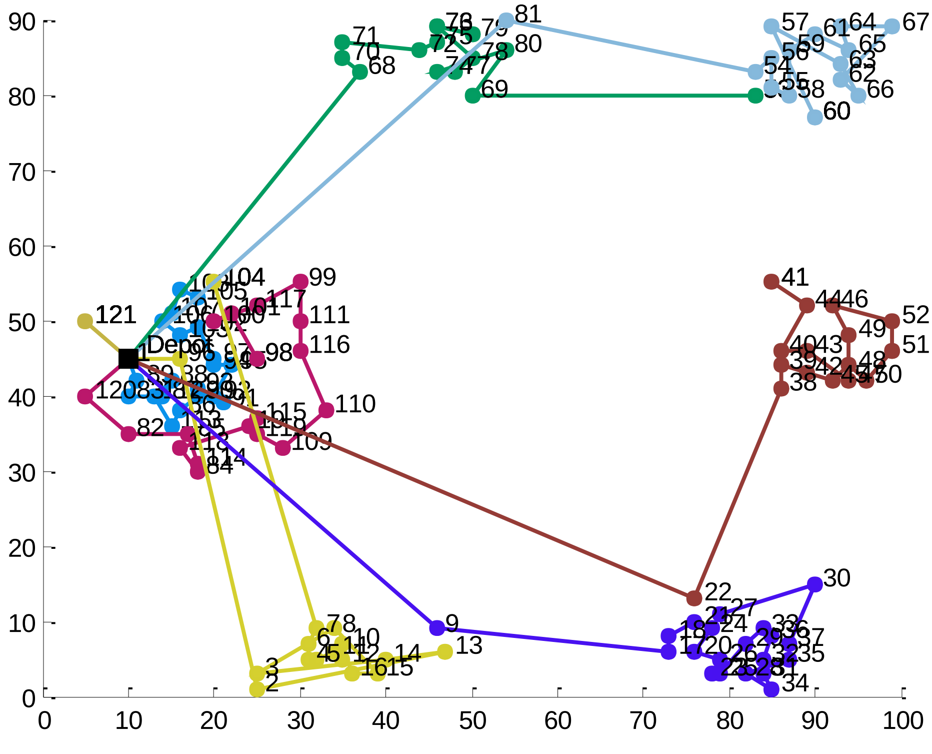


1	21	22	24	27	29	28	26	23	25	31	30	35	37	
1	68	66	64	63	75	73	65	62	69	70	67			
1	11	2	76	4	12	10	9	7	5	8	6	3	99	
1	44	43	42	45	46	49	47	48	51	52	50	41	53	32	36

1	91	90	89	88	87	86	84	83	85	92	96
1	14	18	16	20	19	15	13	17			
1	33	34	39	40							
1	58	56	55	54	57	61	59	60			
1	100	101	98	94	95	97	93				
1	82	77	72	74	78	80	79	81	71		

Συνολική απόσταση : 741,338

5.1.13 Αποτελέσματα προβλήματος C13

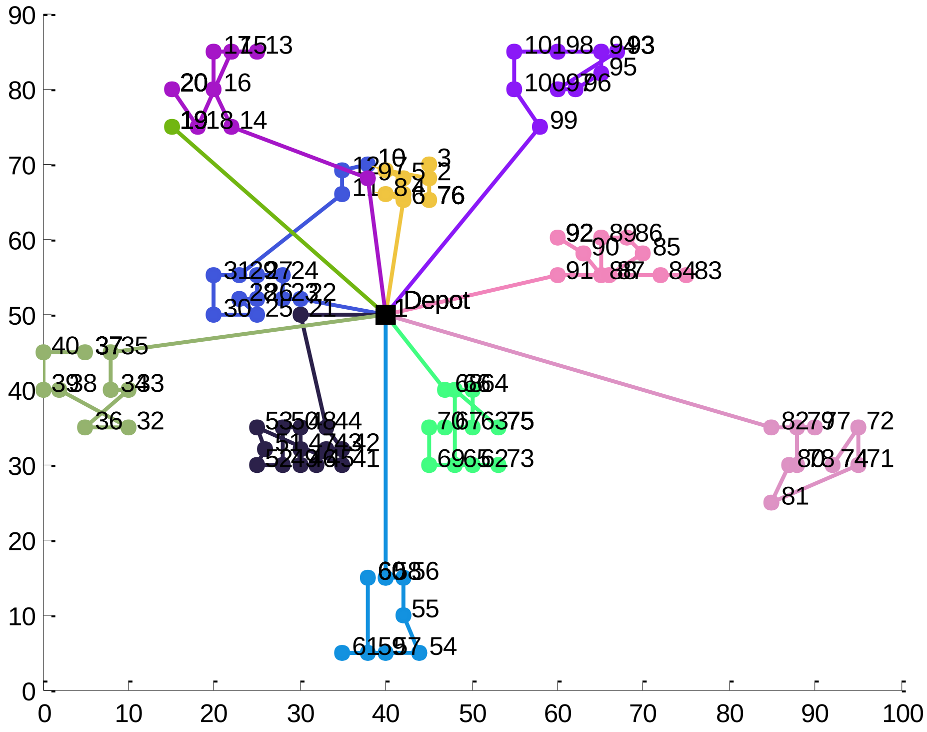


Συνολική απόσταση : 936,952

1	89	83	87	88	112	113	86	90	93	91	92	95	94	97	102	103

106	107	108	105													
1	120	82	85	114	84	118	19	115	119	109	110	116	111	99	117	100
101	98															
1	121															
1	96	2	13	14	3	8	15	16	10	12	11	5	4	6	7	104
1	68	70	71	72	75	73	79	76	78	77	74	80	69	53		
1	9	17	18	21	24	20	26	23	25	29	33	36	32	31	34	28
35	37	30	27													
1	22	38	39	42	45	48	50	51	52	46	49	47	43	40	44	41
1	81	54	56	55	58	59	61	65	64	67	62	66	63	57	60	

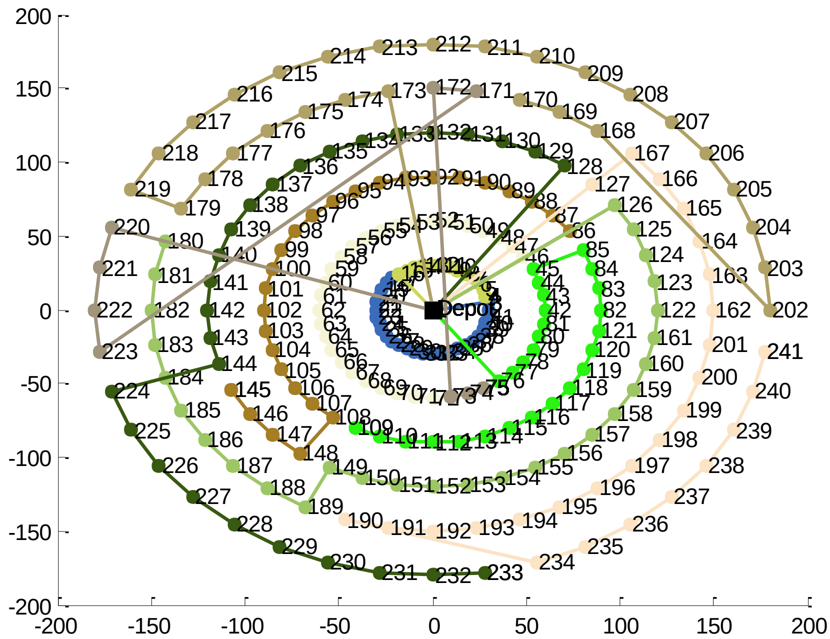
5.1.14 Αποτελέσματα προβλήματος C14



Συνολική απόσταση : 734,770

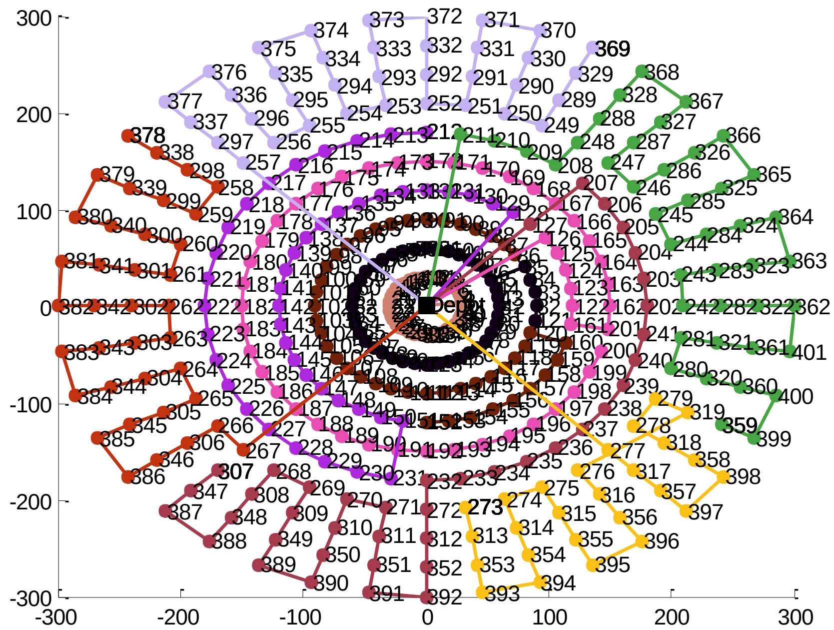
1	22	23	24	27	26	28	25	30	31	29	11	12	10	
1	68	64	63	67	70	69	62	73	65	66	75			
1	6	8	4	5	7	2	3	76						
1	21	44	42	43	45	46	47	48	50	49	52	51	53	41
1	9	14	16	17	13	15	18	20						
1	91	83	84	87	85	86	89	88	90	92				
1	99	100	101	98	94	95	96	97	93					
1	35	34	33	36	32	38	39	40	37					
1	58	56	55	54	57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

5.1.15 Αποτελέσματα προβλήματος Olarge2



Συνολική απόσταση: 5.487,545

5.1.16 Αποτελέσματα προβλήματος Olarge6



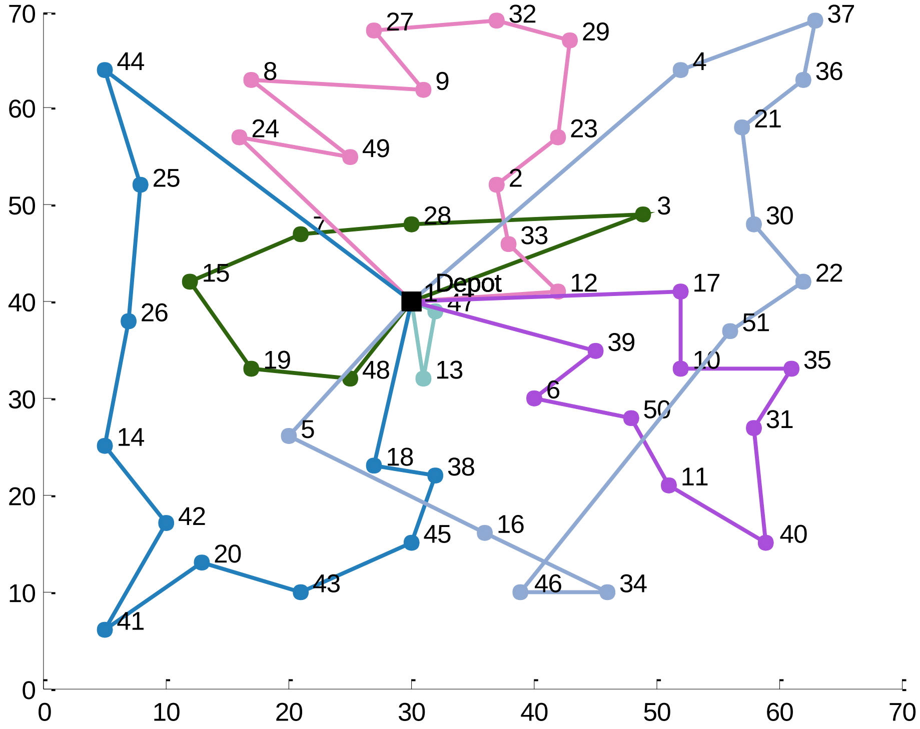
Συνολική απόσταση: 10.452,32

5.2 Περιγραφή και αναπαράσταση αποτελεσμάτων για το Πρόβλημα Δρομολόγησης Οχημάτων Περιορισμένης Χωρητικότητας (CVRP)

Αντίστοιχα δοκιμάσαμε και τον αλγόριθμο για το πρόβλημα περιορισμένης χωρητικότητας πάνω σε προβλήματα αναφοράς. Συγκριμένα χρησιμοποιήσαμε δυο σετ προβλημάτων αναφοράς, τα 14 προβλήματα αναφοράς που έχουν προταθεί από τον Christoforides και 20 προβλήματα μεγάλης κλίμακας όπως έχουν αναφερθεί από τους Golden et .al. [33]. Αναλυτικά εμφανίζονται τα χαρακτηριστικά τους στον πίνακα:

Προβλήματα Christoforides					Προβλήματα Golden et.al.			
Πρόβλημα	**Πελάτες**	**Χωρητικότητα**	**Μέγιστο Μήκος Διαδρομής**	**Service Time**	**Πρόβλημα**	**Πελάτες**	**Χωρητικότητα**	**Μέγιστο Μήκος**
vrpnc1	51	160	∞	0	**kel1**	240	550	650
vrpnc2	76	140	∞	0	**kel2**	320	700	900
vrpnc3	101	200	∞	0	**kel3**	400	900	1200
vrpnc4	151	200	∞	0	**kel4**	480	1000	1600
vrpnc5	200	200	∞	0	**kel5**	200	900	1800
vrpnc6	51	160	200	10	**kel6**	280	900	1500
vrpnc7	76	140	160	10	**kel7**	360	900	1300
vrpnc8	101	200	230	10	**kel8**	440	900	1200
vrpnc9	151	200	200	10	**kel9**	255	1000	∞
vrpnc10	200	200	200	10	**kel10**	323	1000	∞
vrpnc11	121	200	∞	0	**kel11**	399	1000	∞
vrpnc12	101	200	∞	0	**kel12**	483	1000	∞
vrpnc13	121	200	720	50	**kel13**	252	1000	∞
vrpnc14	101	200	1040	90	**kel14**	320	1000	∞
					kel15	396	1000	∞
					kel16	480	1000	∞
					kel17	240	200	∞
					kel18	300	200	∞
					kel19	360	200	∞
					kel20	420	200	

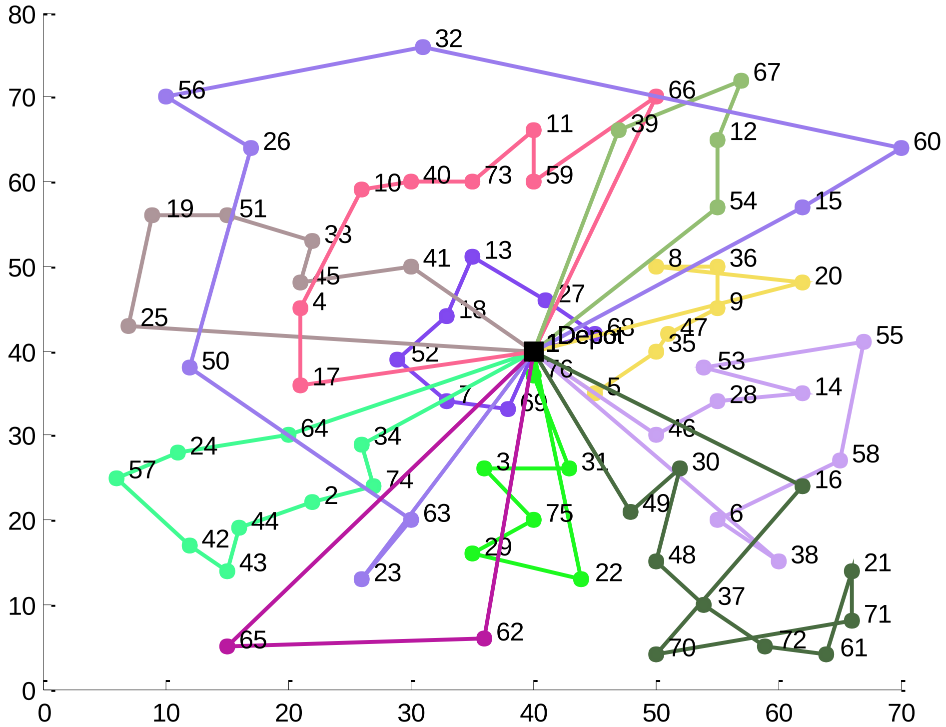
5.2.1 Αποτελέσματα προβλήματος vrpnc1



Συνολική απόσταση : 657,253

1	47	13	1									
1	48	19	15	7	28	3	1					
1	12	33	2	23	29	32	27	9	8	49	24	1
1	39	6	50	11	40	31	35	10	17	1		
1	18	38	45	43	20	41	42	14	26	25	44	1
1	5	16	34	46	51	22	30	21	36	37	4	1

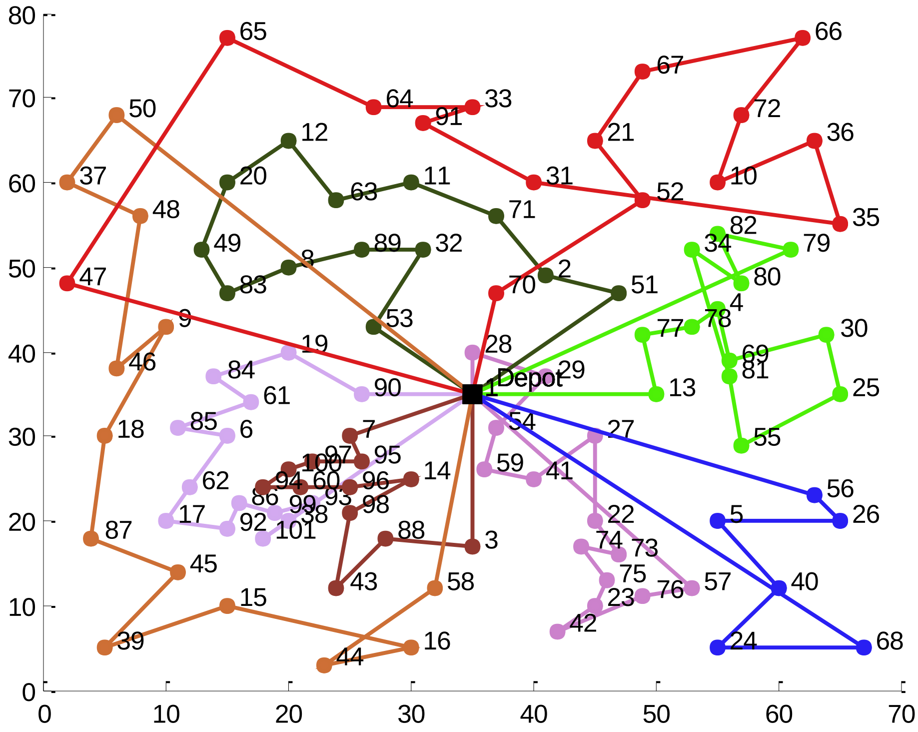
5.2.2 Αποτελέσματα προβλήματος vrpnc2



Συνολική απόσταση : 1.112,606

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

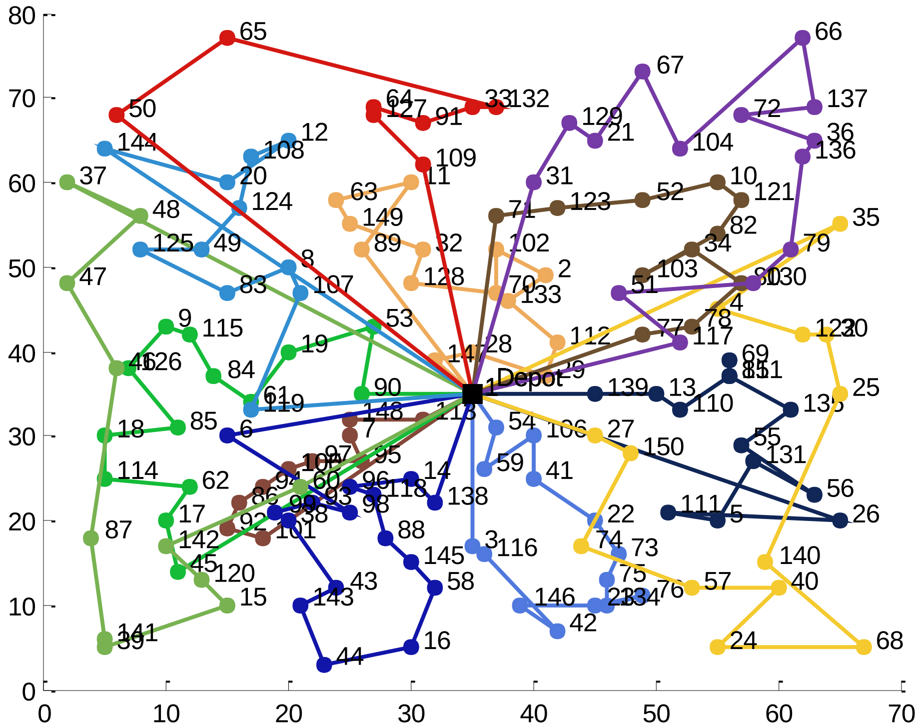
5.2.3 Αποτελέσματα προβλήματος vrpnc3



Συνολική απόσταση : 1.075,575

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

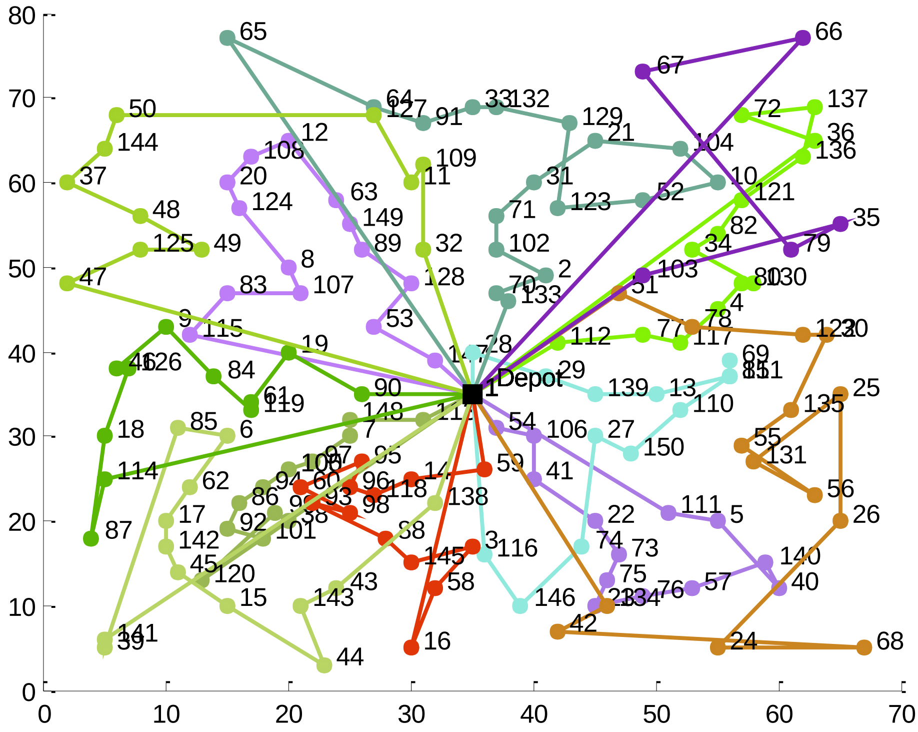
5.2.4 Αποτελέσματα προβλήματος vrpnc4



Συνολική απόσταση : 1.396,348

1	54	59	106	41	22	73	75	134	76	23	146	42	116	3	1		
1	113	148	7	95	97	100	105	94	86	92	101	1					
1	147	28	29	112	133	2	102	70	128	32	149	63	11	89	1		
1	139	13	110	151	69	81	135	55	56	131	5	111	26	1			
1	90	53	19	61	84	115	9	126	85	18	114	62	17	45	1		
1	27	150	74	57	40	24	68	140	25	30	122	4	35	1			
1	138	14	96	118	88	145	58	16	44	143	43	38	99	93	98	6	1
1	77	78	80	34	103	82	121	10	52	123	71	1					
1	60	142	120	15	39	141	87	46	47	48	37	1					
1	117	51	130	79	136	36	72	137	66	104	67	21	129	31	1		
1	119	107	8	83	125	49	124	108	12	20	144	1					
1	109	127	64	91	33	132	65	50	1								

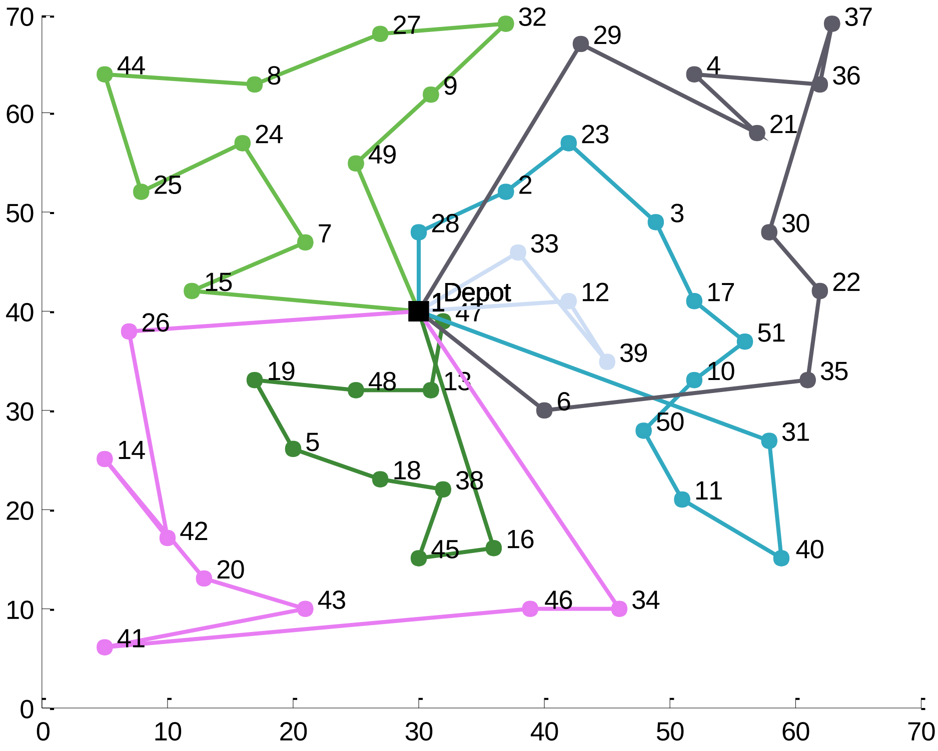
5.2.5 Αποτελέσματα προβλήματος vrpnc 5



Συνολική απόσταση : 1.394,132

1	54	106	41	22	73	75	23	76	57	140	40	5	111	1				
1	113	148	7	97	100	105	94	86	92	101	38	99	120	1	1			
1	147	53	128	89	149	63	12	108	20	124	8	107	83	115	1			
1	28	29	139	13	81	69	151	110	150	27	74	146	116	1				
1	59	14	118	96	95	60	98	93	88	145	3	58	16	1				
1	112	77	117	4	80	130	34	82	121	136	137	72	36	1				
1	133	70	2	102	71	31	21	104	10	52	123	129	132	33	91	64	65	1
1	138	43	143	44	15	45	142	17	62	6	85	39	141	1				
1	90	19	61	119	84	9	46	126	18	87	114	1						
1	51	78	122	30	135	55	56	131	25	26	24	68	42	134	1			
1	32	109	11	127	50	144	37	48	49	125	47	1						
1	103	35	79	67	66	1												

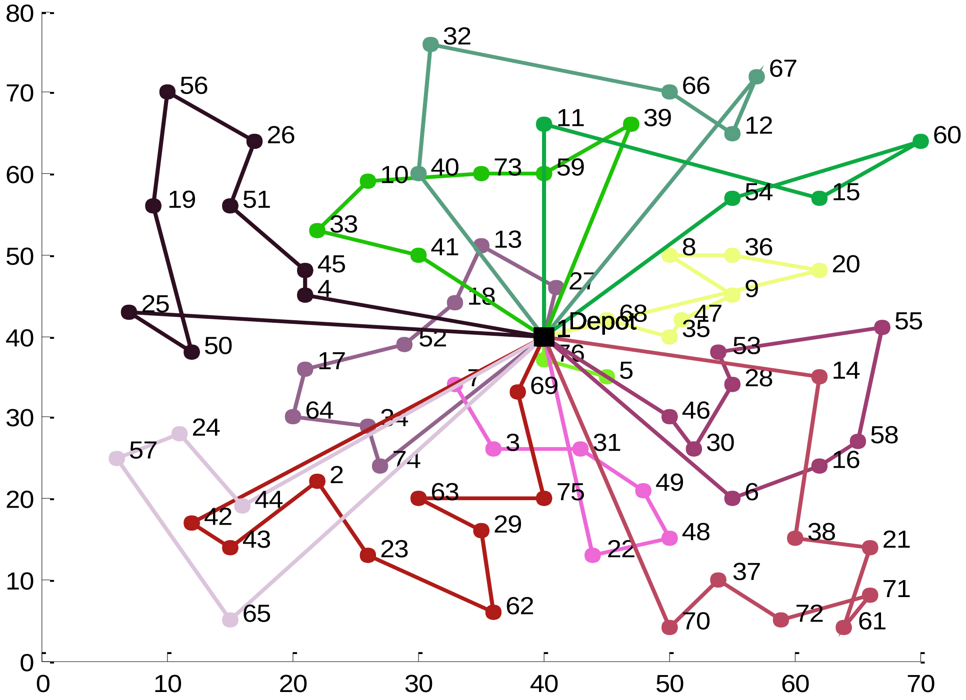
5.2.6 Αποτελέσματα προβλήματος vrpnc 6



Συνολική απόσταση : 685,243

1	47	13	48	19	5	18	38	45	16	1		
1	28	2	23	3	17	51	10	50	11	40	31	1
1	33	39	12	1								
1	49	9	32	27	8	44	25	24	7	15	1	
1	26	42	14	20	43	41	46	34	1			
1	29	21	4	36	37	30	22	35	6	1		

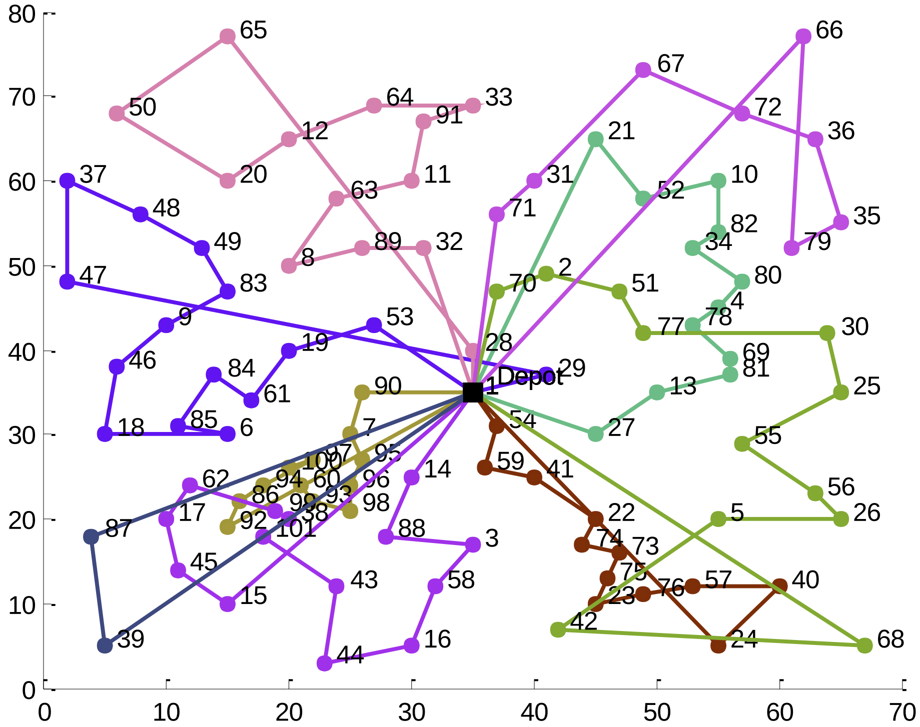
5.2.7 Αποτελέσματα προβλήματος vrpnc 7



Συνολική απόσταση : 1,084.986

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

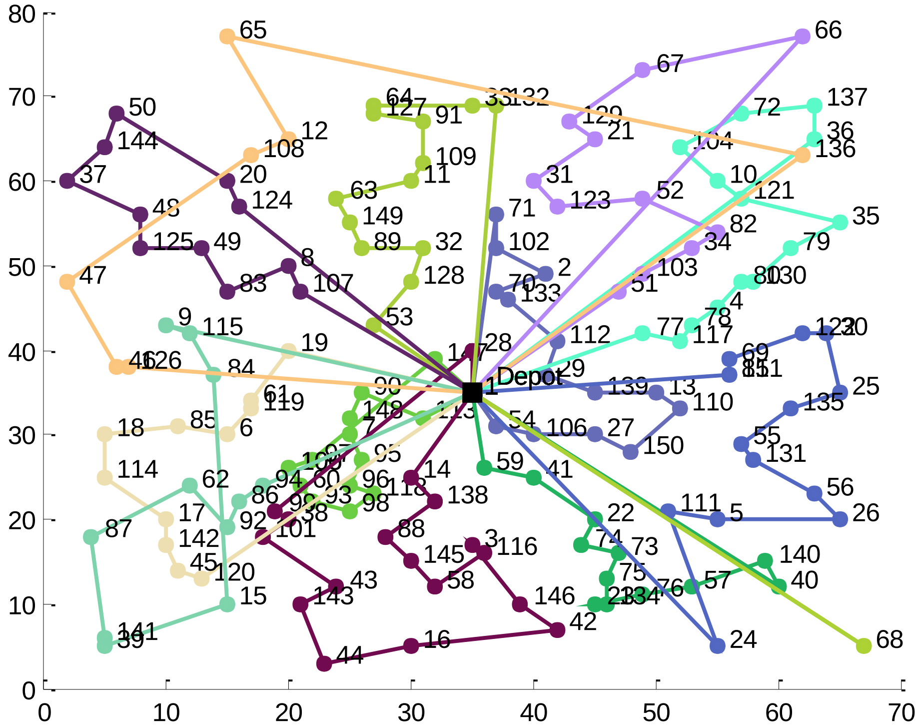
5.2.8 Αποτελέσματα προβλήματος vrpnc 8



Συνολική απόσταση : 1.094,112

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

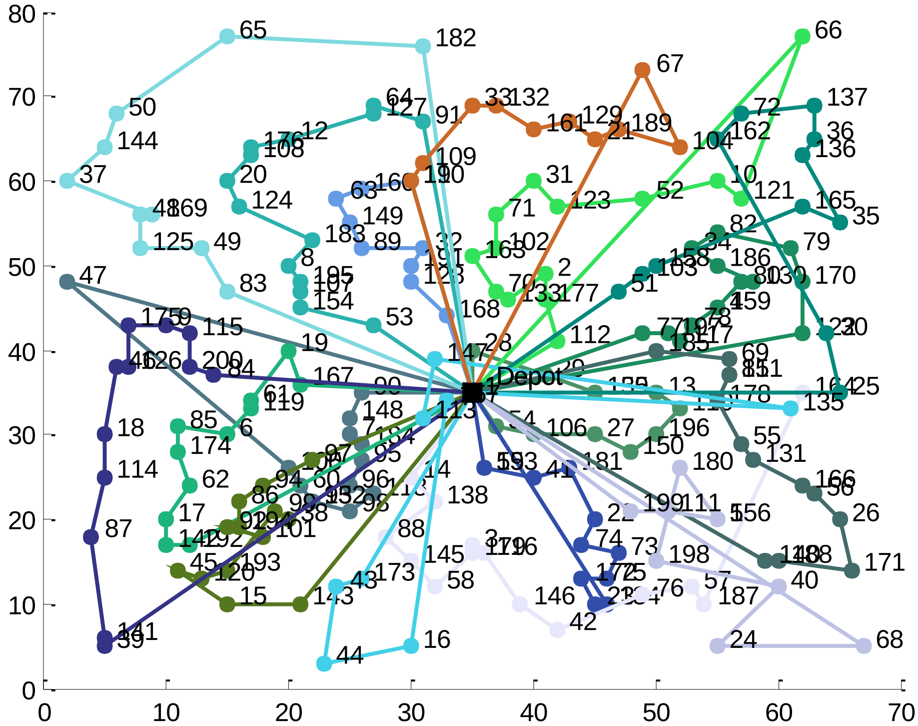
5.2.9 Αποτελέσματα προβλήματος vrpnc 9



Συνολική απόσταση : 1.380,074

1	54	106	27	150	110	13	139	29	112	133	70	2	102	71	1			
1	113	90	148	7	95	96	118	98	93	60	100	105	97	147	1			
1	59	41	22	74	73	75	134	23	76	57	140	40	1					
1	14	138	88	145	58	116	3	146	42	16	44	143	43	101	38	99	28	1
1	53	128	32	89	149	63	11	109	91	127	64	33	132	1				
1	77	117	78	4	80	130	79	35	121	10	104	72	137	36	1			
1	19	61	119	6	85	18	114	17	142	45	120	1						
1	51	103	34	82	52	123	31	21	129	67	66	1						
1	107	8	83	49	125	48	37	144	50	20	124	1						
1	94	86	92	62	87	141	39	15	84	115	9	1						
1	81	151	69	122	30	25	135	55	131	56	26	5	111	24	1			
1	126	46	47	108	12	65	136	1										
1	68	1																

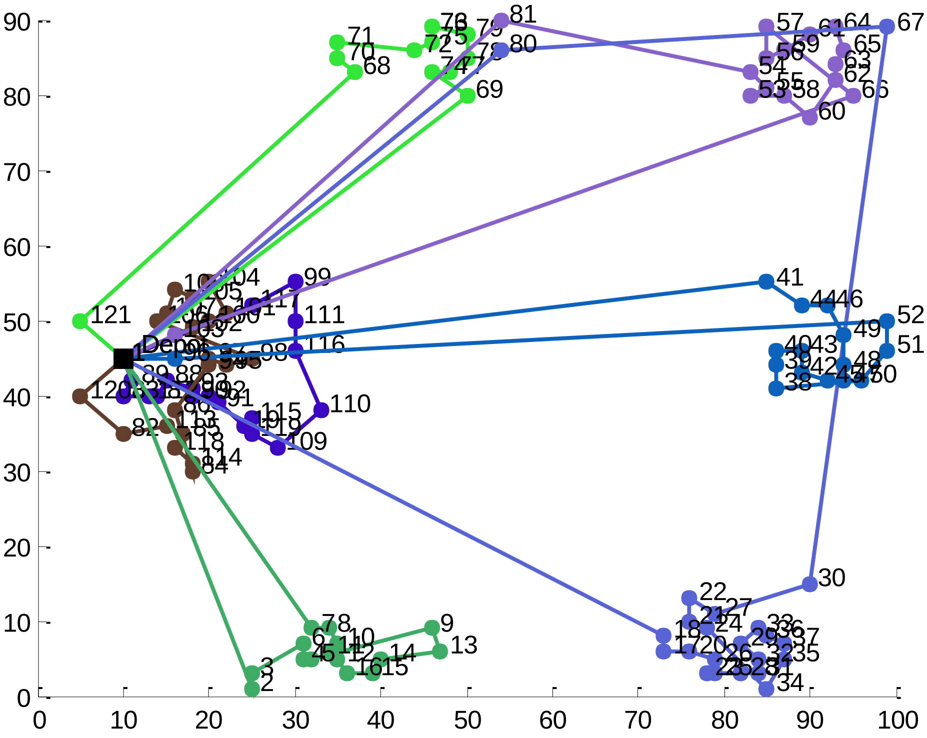
5.2.10 Αποτελέσματα προβλήματος vrpnc 10



Συνολική απόσταση : 1.667,920

1	54	106	27	150	196	110	13	139	155	29	28	1			
1	90	148	7	184	95	96	118	98	93	152	60	100	105	47	1
1	59	153	41	181	22	74	73	75	172	23	134	1			
1	112	177	2	133	70	163	102	71	31	123	52	10	121	66	1
1	168	128	191	32	89	149	63	160	11	1					
1	14	138	88	145	58	179	3	116	146	42	76	57	187	164	1
1	53	154	107	195	8	183	124	20	108	176	12	127	64	91	1
1	167	19	61	119	6	85	174	62	17	142	192	1			
1	97	94	86	194	92	101	38	99	193	120	45	15	143	1	1
1	77	197	117	78	4	159	80	130	186	34	82	79	170	122	1
1	185	69	81	151	178	55	131	166	56	26	171	188	140	1	
1	51	103	158	165	35	136	36	137	72	162	30	25	1		
1	199	111	5	156	180	198	40	24	68	1					
1	84	200	115	9	175	126	46	18	114	87	141	39	1		
1	83	49	125	48	169	37	144	50	65	182	1				
1	173	43	44	16	157	113	147	135	1						
1	190	109	33	132	161	129	21	189	104	67	1				

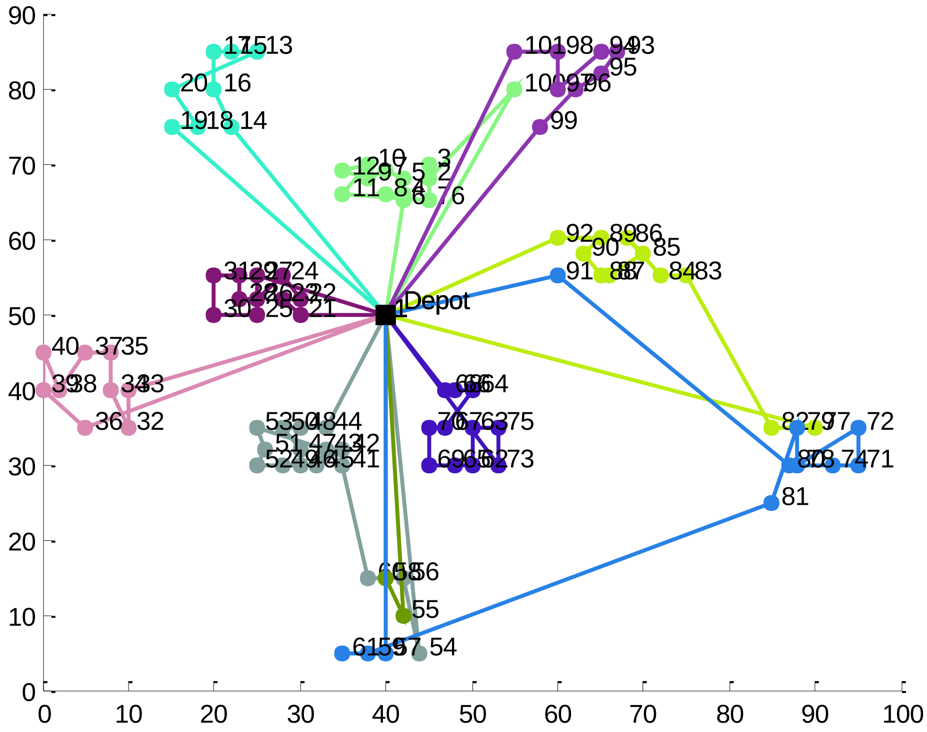
5.2.11 Αποτελέσματα προβλήματος vrpnc 11



Συνολική απόσταση : 1.223,131

1	89	83	112	87	88	93	90	92	91	19	115	119	109	110
116	111	99	117	1										
1	120	82	113	85	118	114	84	86	94	97	95	98	106	107
108	105	104	101	100	102	1								
1	121	68	70	71	72	75	73	76	79	78	77	74	69	1
1	7	8	10	11	12	16	15	14	13	9	5	4	6	3
2	1													
1	81	54	55	53	58	60	62	63	65	64	61	59	56	57
66	103	1												
1	18	17	20	26	25	23	28	31	32	29	33	36	37	35
34	24	21	22	27	30	67	80	1						
1	41	44	46	49	48	47	45	42	43	40	39	38	50	51
52	96	1												

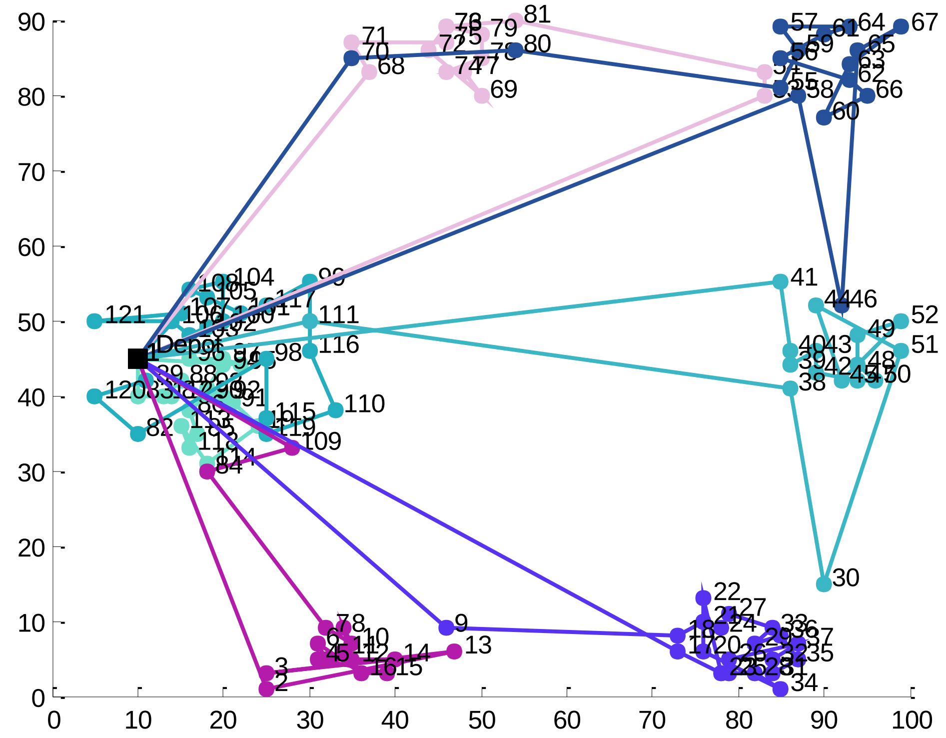
5.2.12 Αποτελέσματα προβλήματος vrpnc 12



Συνολική απόσταση : 1.088,429

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

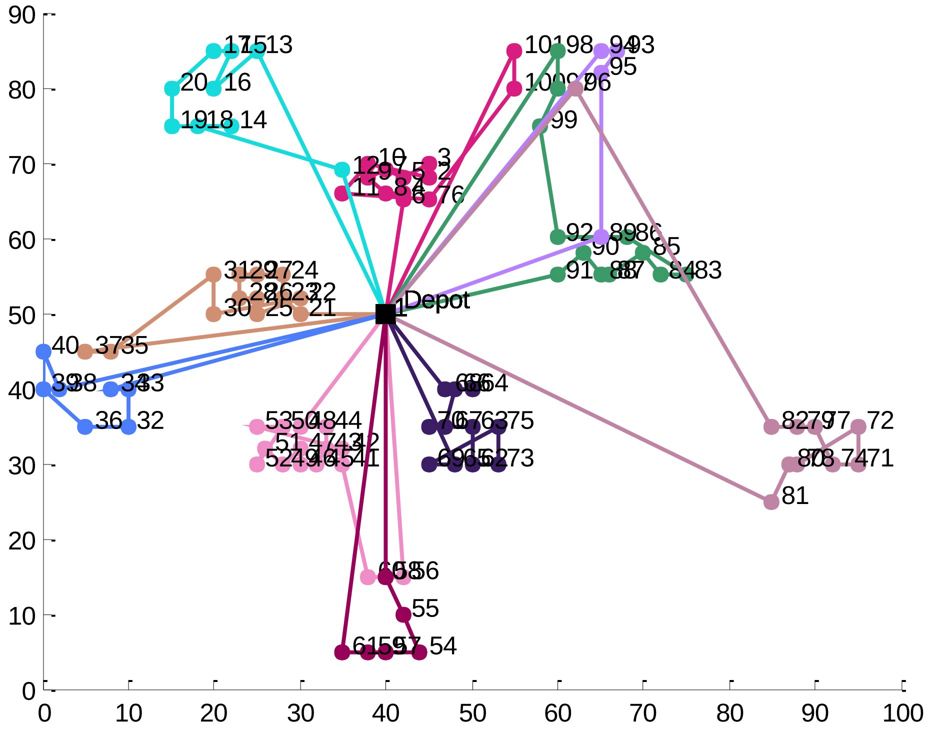
5.2.13 Αποτελέσματα προβλήματος vrpnc 13



Συνολική απόσταση 1.544,308

1	83	87	112	88	90	86	85	118	113	114	19	91	92	93	94	96	97
95	1																
1	89	120	82	98	115	119	110	116	99	117	101	105	108	104	100	102	103
106	121	107	1														
1	109	84	7	10	8	12	11	5	6	16	15	4	14	3	13	2	1
1	68	71	75	76	79	78	74	77	69	72	73	81	54	53	1		
1	70	80	55	59	57	64	61	56	62	66	60	63	67	65	46	58	1
1	9	18	21	24	27	33	29	36	35	31	32	37	26	25	28	34	20
22	23	17	1														
1	111	38	30	51	44	45	52	49	48	50	47	42	43	39	40	41	1

5.2.14 Αποτελέσματα προβλήματος vrpnc 14

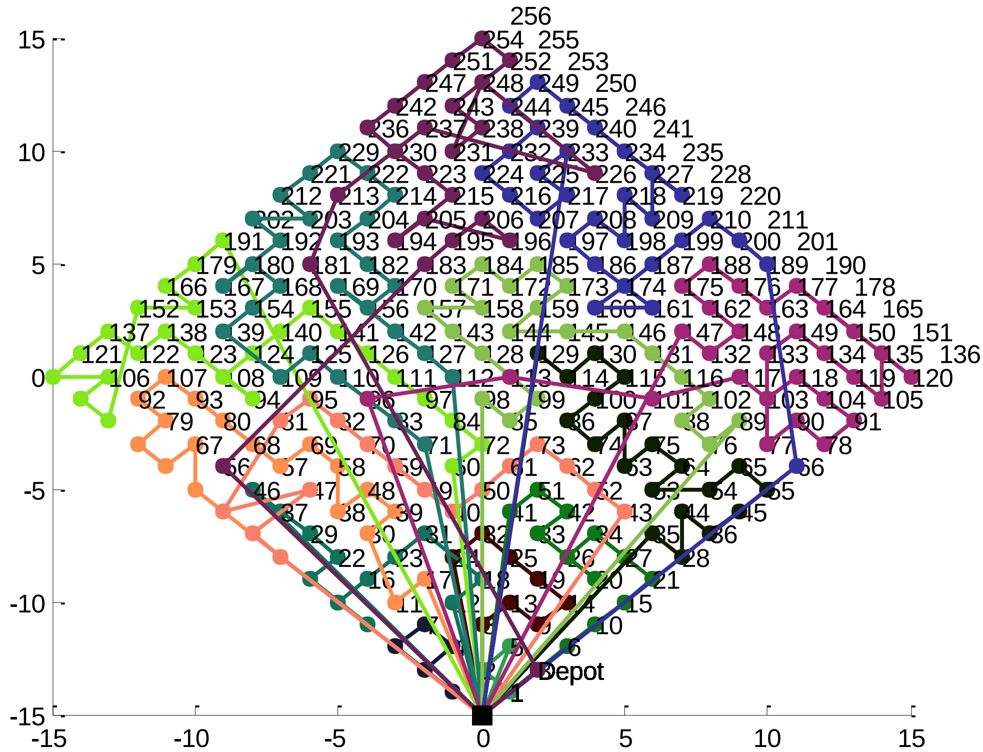


Συνολική απόσταση : 1.116,163

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

5.2.15 Αποτελέσματα προβλήματος kel09

Από τα 20 προβλήματα μεγάλης κλίμακας των Golden et .al., καλύτερο αποτέλεσμα βρήκαμε στο kel09.

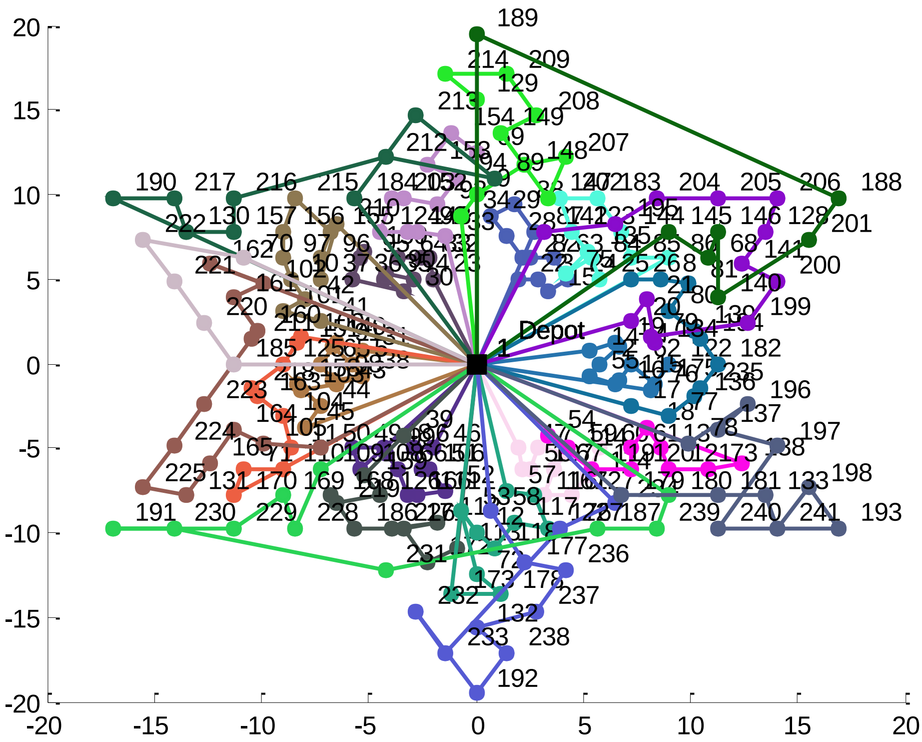


Συνολική απόσταση : 749,652

1	2	4	8	12	7	1							
1	3	5	9	1									
1	13	19	14	20	26	33	41	32	1				
1	10	15	21	28	35	43	34	27	42	52	62	51	1
1	18	25	40	31	23	16	11	22	30	38	47	57	1
1	24	17	39	49	59	48	70	82	69	81	94	108	123
107	93	79	67	80	56	1							
1	29	37	58	46	95	110	96	83	71	60	50	61	73
86	74	63	53	1									
1	44	36	54	45	55	66	77	65	64	76	88	75	101
87	10	11	13	14	13	14	1						

	0	5	1	6	0	5										
1	72	85	98	112	127	142	156	169	155	140	124	109	139			
153	138	152	137	121	122	106	92	166	167	179	191	202	1			
1	84	97	126	141	125	154	168	181	192	180	203	212	213			
221	229	236	230	223	214	204	194	183	170	182	157	143	128	1		
1	113	99	114	144	158	171	172	184	196	185	197	186	173			
159	160	161	147	132	117	102	89	103	1							
1	111	129	116	133	118	134	119	105	91	104	90	149	164			
150	135	120	151	136	178	165	190	177	189	200	188	176	163	148	162	1
1	78	201	211	220	210	199	187	174	175	198	208	218	209	227		
219	235	228	241	246	250	253	249	245	239	232	225	217	226	233	240	1
1	68	195	206	216	207	215	205	224	231	237	242	247	251	254		
256	255	248	244	238	252	234	243	222	193	6	1					

5.2.16 Αποτελέσματα προβλήματος kel17



Συνολική απόσταση : 918,225

Κεφάλαιο 6: Σύγκριση αποτελεσμάτων και συμπεράσματα

Στο προηγούμενο κεφάλαιο παρουσιάσαμε τα βέλτιστα αποτελέσματα του αλγορίθμου για τα διάφορα προβλήματα αναφοράς. Ακολουθεί η σύγκριση τους και για τα δύο προβλήματα που εξετάστηκαν, με τις αντίστοιχες παγκόσμιες βέλτιστες τιμές και υπολογίζουμε την μεταξύ τους ποσοστιαία απόκλιση.

6.1 Σύγκριση αποτελεσμάτων για το Ανοιχτό Πρόβλημα Δρομολόγησης Οχημάτων (OVRP)

Πρόβλημα	Πελάτες	Χωρητικότητα	Μέγιστο Μήκος Διαδρομής	Service Time	Αποτελέσματα αλγόριθμου GRASP	Παγκοσμίως βέλτιστες τιμές	Απόκλιση (%)
C1	51	160	∞	0	514,018	416,06	23,5%
C2	76	140	∞	0	738,014	567,14	30,1%
C3	101	200	∞	0	838,765	639,74	31,1%
C4	151	200	∞	0	1.082,86	733,13	47,7%
C5	200	200	∞	0	1.299,73	893,39	45,5%
C6	51	160	180	10	646,206	412,96	56,5%
C7	**76**	**140**	**144**	**10**	**735,041**	**583,19**	**10,8%**
C8	101	200	207	10	876,217	644,63	14,0%
C9	151	200	180	10	1.098,80	757,84	15,6%
C10	200	200	180	10	1.241,65	875,67	41,8%
C11	121	200	∞	0	906,768	682,14	32,9%
C12	101	200	∞	0	741,338	534,24	38,8%
C13	**121**	**200**	**648**	**50**	**936,952**	**904,04**	**3,6%**
C14	101	200	936	90	734,77	591,87	24,1%
Olarge2	241	550	∞	0	5.487,545	4.557.38	20,4%
Olarge6	**401**	**900**	∞	**0**	**10.452,32**	**9.793,72**	**6,7%**

6.2 Σύγκριση αποτελεσμάτων για το Πρόβλημα Δρομολόγησης Οχημάτων Περιορισμένης Χωρητικότητας (CVRP)

Πρόβλημα	Πελάτες	Χωρητικότητα	Μέγιστο Μήκος Διαδρομής	Service Time	Αποτελέσματα αλγόριθμου GRASP	Παγκοσμίως βέλτιστες τιμές	Απόκλιση (%)
vrpnc1	51	160	∞	0	657,253	524,61	25,3%
vrpnc2	76	140	∞	0	1.112,61	835,26	33,2%
vrpnc3	101	200	∞	0	1.075,58	826,14	30,2%

vrpnc4	151	200	∞	0	1.396,35	1028,42	35,8%
vrpnc5	**200**	**200**	**∞**	**0**	**1.394,13**	**1291,29**	**8,0%**
vrpnc6	51	160	200	10	685,243	555,43	23,4%
vrpnc7	76	140	160	10	1.084,99	909,68	19,3%
vrpnc8	101	200	230	10	1.094,11	865,94	26,3%
vrpnc9	151	200	200	10	1.380,07	1162,55	18,7%
vrpnc10	200	200	200	10	1.667,92	1395,85	19,5%
vrpnc11	121	200	∞	0	1.223,13	1042,11	17,4%
vrpnc12	101	200	∞	0	1.088,43	819,56	32,8%
vrpnc13	**121**	**200**	**720**	**50**	**1.544,31**	**1541,14**	**0,2%**
vrpnc14	101	200	1040	90	1.116,16	866,37	28,8%
kel9[1]	255	1000	∞	-	749,652	583,39	28,5%
Kel17	240	200	∞	-	918,225	707,79	29.7%

6.3 Συμπεράσματα

Από τους παραπάνω πίνακες για την σύγκριση των βέλτιστων τιμών του αλγορίθμου, μπορούμε να εξάγουμε τα εξής συμπεράσματα :

- Η κατά μέσο όρο απόκλιση για το ανοιχτό πρόβλημα δρομολόγησης (OVRP) είναι 29,71%, η οποία κυμαίνεται από ελάχιστο 3,6%, έως μέγιστο 56,5%.
- Ενώ για το πρόβλημα δρομολόγησης οχημάτων περιορισμένης χωρητικότητας (CVRP), ο μέσος όρος των αποκλίσεων είναι 22,78%, με εύρος από 0,2% έως 35,8%.

Άρα ο αλγόριθμος λειτούργησε πολύ καλύτερα για το CVRP στα 14 προβλήματα του Christoforides, αφού έχει αρκετά μικρές αποκλίσεις από τις παγκοσμίως βέλτιστες τιμές .Όμως στο προβλήματα μεγάλης κλίμακας δεν ανταπεξέρχεται το ίδιο καλά, με αποκλίσεις πάνω από 28%. Σε αντίθεση στα προβλήματα μεγάλης κλίμακας για το

[1] Εξαιτίας του μεγάλου όγκου δεδομένων, δεν παρουσιάζονται αναλυτικά όλα τα προβλήματα μεγάλης κλίμακας, τύπου kel και Olarge.

OVRP, λειτούργησε καλύτερα ο αλγόριθμος Grasp, όπου εμφανίζει ελάχιστη απόκλιση 6,7%.

Ακόμα κατά την διάρκεια εφαρμογής του αλγορίθμου στα διάφορα προβλήματα έγιναν οι παρακάτω παρατηρήσεις :

- Ο αλγόριθμος GRASP λειτουργεί καλύτερα όταν αυξάνουμε τις επαναλήψεις του, άρα όταν δημιουργούμε πολλές τυχαίες αρχικές λύσεις, με την μέθοδο της τυχαιοποιημένης απληστίας.
- Η διατήρηση σταθερού αριθμού εναρκτήριων λύσεων με ταυτόχρονη αύξηση των επαναλήψεων της τοπικής αναζήτησης δεν απέφερε καρπούς.
- Η μετα-βελτιστοποίηση, με την μείωση των αριθμών διαδρομών βελτιώνει αρκετά την λύση.
- Τέλος, παρατηρήθηκε ότι η αύξηση στο μέγεθος της λίστας περιορισμού, οδηγούσε σε χειρότερη λύση.

Μελλοντική ενέργεια για την βελτίωση της ποιότητας λύσης του αλγορίθμου GRASP αποτελεί η χρήση διαφορετικού αλγόριθμου τοπικής αναζήτησης ή προσθήκη επιπλέον αλγορίθμου (για παράδειγμα 3-opt ή ανταλλαγή κόμβων μεταξύ διαδρομών ή ανακύκλωση μονοπατιών), η αύξηση των επαναλήψεων καθώς και η μείωση του μεγέθους της λίστας περιορισμού.

Βιβλιογραφία

[1] https://cscmp.org/sites/default/files/user_uploads/resources/downloads/glossary-2013.pdf

[2] http://www.dhldiscoverlogistics.com/cms/en/course/origin/historical_development.jsp

[3] Ι.Μαρινάκης, Α.Μυγδαλάς, Σχεδιασμός και Βελτιστοποίηση της Εφοδιαστικής Αλυσίδας. Εκδόσεις “σοφία”, Θεσσαλονίκη 2008.

[4] Dantzig and Wolfe, 1960 ,G.B. Dantzig, P. Wolfe, Decomposition principle for linear programming, Operations Research, 8 (1960), pp. 101–111

[5] Laporte, G., Gendreau, M., Potvin, J.Y., Semet, F.: Classical and Modern Heuristics for the Vehicle Routing Problem. International Transactions in Operational Research 7, 285–300 (2000)

[6] P. Toth and D. Vigo. An overview of vehicle routing problems, in The Vehicle Routing Problem (P. Toth and D. Vigo eds.). SIAM Monographs on Discrete Mathematics and Applications, 2002.

[7] E.L. Lawler, J.K. Lenstra, A. Rinnooy, H.G. Kan, D.B. Shmoys The Traveling Salesman Problem: A Guided Tour of Combinatorial Optimization

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

[9] S.A. MirHassani, N. Abolghasemi. A particle swarm optimization algorithm for open vehicle routing problem. Expert Systems with Applications, Volume 38, Issue 9, September 2011, Pages 11547-11551

[10] Z. Fu, R. Egles, L.Y.O. Li, A new tabu search heuristic for the open vehicle routing problem, Journal of the Operational Research Society, 56 (2005), pp. 267–274

[11] M. Dror, P. Trudeau. Savings by split delivery routing. Transportation Science,

23: 141-145, 1989.

[12] M. Dror, P. Trudeau. Split delivery routing. Naval Research Logistics, 37: 383-

402, 1990.

[13] Branch-and-cut algorithms for the split delivery vehicle routing problem

Claudia Archetti, Nicola Bianchessi , M. Grazia Speranza, 2014.

[14] B. Kallehauge, J. Larsen, O.B.G. Madsen, M. Solomon,Vehicle Routing Problem with Time Windows, Springer, Column Generation (2005), pp. 67–98

[15] G. Desaulniers, J. Desrosiers, A. Erdmann, M.M. Solomon and F. Soumis. VRP with Pickup and Delivery, Chapter 9 in The Vehicle Routing Problem, P. Toth and D. Vigo, eds., SIAM Monographs on Discrete Mathematics and Applications,

Philadelphia, PA, 225–242, 2002

[16] P. Toth and D. Vigo. VRP with Backhauls, Chapter 8 in The Vehicle Routing

Problem, P. Toth and D. Vigo, eds., SIAMMonographs on Discrete Mathematics and Applications, Philadelphia, PA, 195–224, 2002

[17] A. Mingozzi, S. Giorgi, An exact algorithm for the vehicle routing problem with backhauls, Transportation Science, 33 (3) (1999), pp. 315–329

[18] Yang, W.H., Mathur, K., Ballou, R. H.: Stochastic vehicle routing problem with restocking. Transportation Science 34, 99–112 (2000)

[19] H.N. Psaraftis. Dynamic Vehicle Routing Problems. Chapter 11 in Vehicle Routing: Methods and Studies, B.L. Golden and A.A. Assad, eds., North-Holland,

Amsterdam, 223–248, 1988

[20] Laporte, G., Louveaux, F.V., Van hamme, L. (2002). An integer-shaped algorithm for the capacitated vehicle routing problem with stochastic demands. Operations Research 50, 415–423.

[21] Clarke, G., Wright, J.W. (1964). Scheduling of vehicles from a central depot to a number of delivery points. Operations Research 12, 568–581

[22] Gillett, B.E., Miller, L.R. (1974). A heuristic algorithm for the vehicle-dispatch problem. Operations Research21, 340–349

[23] Fisher, M.L., Jaikumar, R. (1981). A generalized assignment heuristic for the vehicle routing problem. Networks 11, 109–124

[24] Papadimitriou, C. H., Steiglitz, K. (1982). Combinatorial Optimization - Algorithms and Complexity, Prentice-Hall, Englewood Cliffs, N. J.

[25] Talbi, E.-G. (2009).Metaheuristics: From Design to Implementation, John Wiley and Sons, USA.

[26] T.A. Feo and M.G.C. Resende, “Greedy randomized adaptive search procedure,” Journal of Global Optimization, vol. 6, pp. 109–133, 1995.

[27] Festa, P., Resende, M. G. C. (2002). GRASP: An A notated Bibliography, Essays and Surveys on Metaheuristics, Ribeiro, C. C., Hansen, P. (Editors), Kluwer Academic Publishers, Norwell, 325-367.

[28] M.G.C. Resende and C.C. Ribeiro, “Greedy randomized adaptive search procedures,” in Handbooks of Metaheuristics ,F. Glover and G.A. Kochenberger (Eds.), Kluwer Academic Publishers Dordrecht, 2003, pp. 219–249.

[29] Waters, C.D.J. (1987) A Solution Procedure for the Vehicle Scheduling Problem Based on Iterative Route Improvement. Journal of Operational Research Society, 38, 833-839.

[30] Lin, S. (1965). Computer Solutions of the Traveling Salesman Problem. Bell Systems, Technical Journal, 44, 2245-2269

[31] N. Christofides, A.Mingozzi,P. Toth, The vehicle routing problem, in: N. Christofides, A. Mingozzi, P. Toth, C. Sandi(Eds.),Combinatorial Optimization, Wiley,Chichester,1979.

[32] F.Li, B.Golden, E.Wasil, The open vehicle routing problem: algorithms, large- scale test problems, and computational results,Comput.Oper.Res.34(2007) 2918–2930

[33] Golden, B. L., Wasil, E. A., Kelly, J. P., & Chao, I. M. (1998). The impact of Metaheuristics on solving the vehicle routing problem: Algorithms, problem Sets, and computational results. In T. G. Critic & G. Laporte (Eds.), Fleet Management and logistics (pp. 33–56). Boston: Kluwer Academic Publishers