ΠΟΛΥΤΕΧΝΕΙΟ ΚΡΗΤΗΣ

ΣΧΟΛΗ ΜΗΧΑΝΙΚΩΝ ΠΑΡΑΓΩΓΗΣ ΚΑΙ ΔΙΟΙΚΗΣΗΣ

Αλγόριθμος Άπληστης Τυχαιοποιημένης Προσαρμοστικής Αναζήτησης Για Το Πρόβλημα Δρομολόγησης Οχημάτων Με Χρονικά Παράθυρα

Greedy Randomized Adaptive Search Procedure For Vehicle Routing Problem With Time Windows

Συγγραφέας:

Σταμούλης Θωμάς

Διπλωματική Εργασία

Επιβλέπων Καθηγητής: **Δρ Ιωάννης Μαρινάκης**, Επίκουρος Καθηγητής

Χανιά 2016

"Η διπλωματική αυτή εργασία αφιερώνεται στην αγαπημένη μου θεία Γεωργία, που την έχασα τόσο νωρίς"

Ευχαριστίες

Αρχικά θέλω να ευχαριστήσω τον επιβλέποντα καθηγητή μου κ. Ιωάννη Μαρινάκη για όλη τη βοήθεια και την υποστήριξη του όλο αυτό το διάστημα της εκπόνησης της διπλωματικής μου εργασίας. Επίσης ένα μεγάλο ευχαριστώ στην κα.Μάγδα Μαρινάκη και στον κ. Ψύχα Ηρακλή-Δημήτρη για την πολύτιμη βοήθεια τους.

Πολλά ευχαριστώ θέλω να δώσω στους φίλους μου και στους κοντινούς μου ανθρώπους τόσο στα Χανιά όσο και στον τόπο καταγωγής μου, που όλα αυτά τα χρόνια ήταν εκεί για μένα και στα εύκολα και στα δύσκολα, προσφεροντάς μου πολλές όμορφες στιγμές και αναμνήσεις που θα με ακολουθούν για πάντα.

Τα μεγαλύτερα ευχαριστώ όμως πάνε στους πραγματικούς ήρωες της ζωής μου, στην οικογένειά μου, για όλη την υποστήριξη σε κάθε μου απόφαση και σε κάθε μου βήμα στη ζωή μου. Ένα μεγάλο ευχαριστώ λοιπόν στον πατέρα μου Αναστάσιο και στη μητέρα μου Αναστασία για τις θυσίες που έκαναν για να τα καταφέρω. Τέλος θα ήθελα να ευχαριστήσω τον αδερφό μου Δημήτρη, που πάντα ως μεγαλύτερος αδερφός είναι πάντα εκεί ως καθοδηγητής και ως πρότυπο για μένα.

Σας ευχαριστώ.

Περιεχόμενα

Κατάλογος Σχημάτων

Κατάλογος Πινάκων

ΚΕΝΗ ΣΕΛΙΔΑ

Περίληψη

Η συγκεκριμένη διπλωματική έχει σαν αντικείμενο το πρόβλημα της δρομολόγησης οχημάτων με χρονικά παράθυρα (Vehicle routing problem with time windows). Στόχος είναι η αποδοτική επίλυση του προβλήματος για τη μείωση της ευκλείδειας απόστασης που τα οχήματα της Εφοδιαστικής αλυσίδας καλούνται να διανύσουν. Για την περιγραφή του προβλήματος ορίζονται κατάλληλα οι απαραίτητοι περιορισμοί για τον αριθμό και τις τοποθεσίες των πελατών, τη ζήτηση ανά πελάτη, και το χρονικό περιθώριο στο οποίο θα πρέπει να έχει εξυπηρετηθεί. Επίσης είναι γνωστά ο αριθμός και η χωρητικότητα των διαθέσιμων οχημάτων. Στα πλαίσια της παρούσας εργασίας χρησιμοποιείται και υλοποιείται η Διαδικασία Άπληστης Τυχαιοποιημένης Προσαρμοστικής Αναζήτησης (Greedy Randomized Adaptive Search Procedure (GRASP)). Στο πρώτο μέρος της επίλυσης, χρησιμοποιείται ένας απλός αλγόριθμος απληστείας, κατάλληλα τροποποιημένος, για την εύρεση αρχικής εφικτής λύσης που ικονοποιεί τους περιορισμούς. Η αρχική λύση δεν εγγυάται τοπικό ελάχιστο, συνεπώς στο δεύτερο μέρος της υλοποίησης αναπτύσονται υπορουτίνες τοπικής αναζήτησης 1-1 ανταλλαγή (1-1 exchange) και 2-opt. Στην εργασία παρουσιάζεται η προτεινόμενη υλοποίηση της μεθόδου, καθώς και τα αποτελέσματα από τη χρήση των αλγορίθμων. Για την ανάπτυξη των ρουτινών χρησιμοποιήθηκε το προγραμματιστικό περιβάλλον την MATLAB.

Λέξεις Κλειδιά

Πρόβλημα Δρομολόγησης Οχημάτων, Διαδικασία Άπληστης Τυχαιοποιημένης Προσαρμοστικής Αναζήτησης (GRASP), Μέθοδος 2-opt, Χρονικά Παράθυρα, Μεθευρετικοί Αλγόριθμοι βασισμένοι στη Γειτονιά Αναζήτησης, Απλοί Ευρετικοί Αλγόριθμοι, Αλγόριθμοι Τοπικής Αναζήτησης

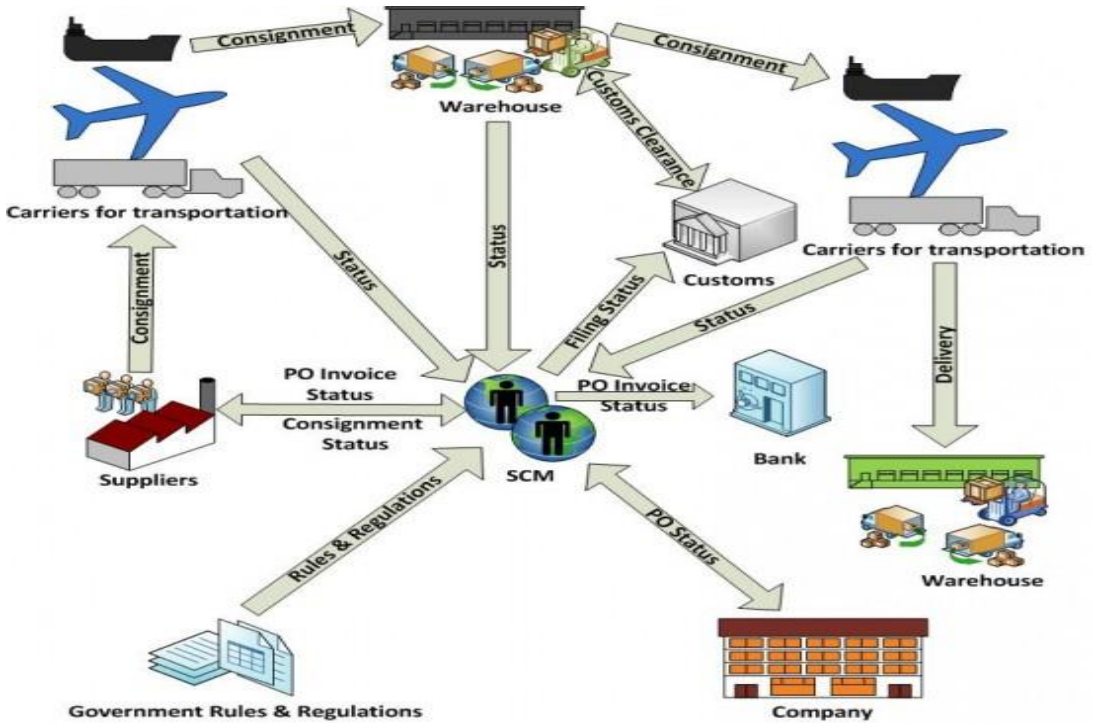
Κεφάλαιο 1
Εισαγωγή

1.1 Η εφοδιαστική αλυσίδα

Οδηγούμενοι από την παγκοσμιοποίηση και τις διαρκώς εκτεινόμενες ανάγκες των πελατών, η **Εφοδιαστική Αλυσίδα (Supply Chain)** παίζει καθοριστικό ρόλο στη δημιουργία πλεονεκτήματος για όλες τις επιχειρήσεις. Το ανταγωνιστικό περιβάλλον που έχει δημιουργηθεί, έχει οδηγήσει τις επιχειρήσεις στην ανακάλυψη νέων πηγών κερδοφορίας όπου μπορούν να αποκτήσουν πλεονέκτημα έναντι άλλων επιχειρήσεων. [1]

Με τον όρο **Εφοδιαστική Αλυσίδα** εννοούμε τη ροή υλικών και υπηρεσιών από τον προμηθευτή πρώτων υλών ή τον κατασκευαστή μέχρι τον τελικό καταναλωτή, αλλά παράλληλα και τη ροή πληροφοριών μεταξύ των μελών της ίδιας αλυσίδας. Η διαχείρησή της γίνεται σε πολλά στάδια και περιλαμβάνει εμπλεκόμενους από τη στιγμή που προμηθεύονται τις πρώτες ύλες, μέχρι τη στιγμή που το τελικό προϊόν θα φτάσει στον πελάτη ή στον καταναλωτή. Δηλαδή, η εφοδιαστική αλυσίδα αποτελείται από τους κατασκευαστές και προμηθευτές υλικών, την εφοδιαστική διαχείρηση δηλαδή τις πρώτες ύλες, τα αποθέματα και τα έτοιμα προϊόντα, τους χώρους αποθήκευσης, τα κέντρα διανομών, τους μεταφορείς, τους πωλητές και τους πελάτες. Παρατηρούμε πως δεν είναι στατική η εφοδιαστική αλυσίδα, αλλά είναι ένα δυναμικό σύστημα που εξελίσσεται μέσα στο χρόνο. [1]

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



Σχήμα 1.1: *Εφοδιαστική αλυσίδα* [2]

Η κυριότερη πρόκληση που έχει ν'αντιμετωπίσει σ'αυτήν την ολοκλήρωση είναι η επίτευξη μιας ισορροπίας μεταξύ απόδοσης και κόστους που να βελτιστοποιούν τους στόχους της επιχείρησης.

1.2 Εφοδιαστική (Logistics)

Ιστορικά τεκμηριωμένα, η εφοδιαστική (logistics), με την έννοια που πλησιάζει αυτήν που της αποδίδουμε σήμερα, ανάγεται στην εποχή των Ρωμαϊκών χρόνων όταν η ταχεία ανάπτυξη των λεγεώνων στα διάφορα σημεία των αυτοκρατορικών συνόρων για την αντιμετώπιση επιδρομών ήταν αναγκαία και έθετε ιδιαίτερες απαιτήσεις στην ύπαρξη οδών και στην επιτυχή και ταχεία μεταφορά διαταγών, υλικών και ανδρών. Έκτοτε υπάρχουν πολλές αναφορές στον όρο εφοδιαστική, και φτάνουμε στο σήμερα όπου παίζει καθοριστικό ρόλο στην απόκτηση πλεονεκτήματος των επιχειρήσεων έναντι των ανταγωνιστών τους και κυρίως στη μείωση των δαπανών τους. [4]

Ο όρος εφοδιαστική αναφέρεται στη διαχείριση της ροής αγαθών ή υλικών από την πηγή στο σημείο κατανάλωσης ή ακόμα και στο σημείο απόθεσης. Είναι δηλαδή το τμήμα που σχεδιάζει, υλοποιεί και ελέγχει την αποδοτική και αποτελεσματική ροή και αποθήκευση των προϊόντων, των υπηρεσιών και των πληροφοριών από το σημείο προέλευσης τους μέχρι το σημείο κατανάλωσης τους, με στόχο τη βέλτιστη ικανοποίηση των πελατών.

1.3 Κόστος και Ανάλυση Αλληλοπαραχώρησης

Εφοδιαστικές δαπάνες δημιουργούνται από τις δραστηριότητες που υποστηρίζουν την εφοδιαστική διαδικασία. Έχουμε δηλαδή δαπάνες που προέρχονται από την εξυπηρέτηση του πελάτη, από τις μεταφορές και τις διανομές, από την εκπαίδευση του προσωπικού, από την διαχείριση των εγκαταστάσεων, την επεξεργασία των παραγγελιών και των πληροφοριών, τις επικοινωνίες και φυσικά τη διαχείριση των αποθεμάτων.

Η μεμονωμένη βελτιστοποίηση του κόστους όμως, δε συνεπάγεται βελτιστοποίηση του συνολικού εφοδιαστικού κόστους. Αντίθετα, η υποβελτιστοποίηση μιας ή περισσότερων δραστηριοτήτων, λαμβάνοντας πάντοτε υπ'όψιν τις μεταξύ τους σχέσεις και τις αλληλεπιδράσεις με άλλες εφοδιαστικές δραστηριότητες, μπορεί να οδηγήσει στη βελτίωση της απόδοσης της αλυσίδας και του ολικού εφοδιαστικού κόστους.

Συμπεραίνουμε συνεπώς ότι η βελτιστοποίηση της απόδοσης της εφοδιαστικής αλυσίδας απαιτεί να ληφθεί υπ'όψιν η αλληλεξάρτηση των εφοδιαστικών δραστηριοτήτων και προϋποθέτει την ανάλυση του βαθμού παραχώρησης για τις δαπάνες που προκύπτουν από αυτές τις δραστηριότητες. Με άλλα λόγια η ελαχιστοποίηση του συνολικού εφοδιαστικού κόστους, δίνει τη δυνατότητα στην επιχείρηση να έχει τη δυνατότητα να εξυπηρετεί στο βέλτιστο βαθμό τους πελάτες της, με σκοπό τη βιωσιμότητα της αρχικά και στη συνέχεια τη μεγιστοποίηση της μακροπρόθεσμης κερδοφορίας της εταιρείας. [1]

1.4 Μεταφορές και Αποθέματα

Οι μεταφορές και τα αποθέματα είναι κομβικής σημασίας για την εφοδιαστική αλυσίδα, καθώς αποτελούν τις πιο δαπανηρές δραστηριότητες της. Οι μεταφορές μόνο καταλαμβάνουν το μισό από το συνολικό κόστος της εφοδιαστικής αλυσίδας. Επίσης αποτελούν το σύνδεσμο μεταξύ της παραγωγής, της αποθήκευσης και της κατανάλωσης. Εξαιτίας της σημαντικότητας τους, οι επιχειρήσεις δίνουν μεγάλη σημασία στα προβλήματα και τις ανάγκες που σχετίζονται με τις μεταφορές, προκειμένου να επιτευχθεί η ομαλή τους λειτουργία και κυρίως να περιοριστεί το κόστος. [1]

Οι μεταφορές χωρίζονται σε κατηγορίες, ανάλογα με αυτές που περιλαμβάνουν τη διακίνηση των πρώτων υλών στους χώρους επεξεργασίας και στα εργοστάσια και σε αυτές που περιλαμβάνουν τη μεταφορά από τα εργοστάσια ή την αποθήκη του τελικού προϊόντος προς τον καταναλωτή. Τα προβλήματα που συναντώνται σε αυτές τις κατηγορίες περιλαμβάνουν το στόλο των οχημάτων, τη δρομολόγηση, το σχεδιασμό δικτύου διανομής, την επιλογή προσωπικού που θα εκτελέσει τα δρομολόγια κ.α..

ΚΕΝΗ ΣΕΛΙΔΑ

Κεφάλαιο 2
Το Πρόβλημα Δρομολόγησης Οχημάτων

2.1 ΕΙΣΑΓΩΓΗ

Δρομολόγηση Οχημάτων

Όπως αναφέρθηκε και παραπάνω, η παγκοσμιοποίηση της οικονομίας έχει οδηγήσει στη συνεχιζόμενη ανάπτυξη του εμπορίου και των μεταφορών σε παγκόσμιο επίπεδο. Το υψηλό κόστος των μεταφορών, η πολυπλοκότητα των προβλημάτων, οι περιορισμοί που υπάρχουν, καθώς και οι υψηλές απαιτήσεις των πελατών έχουν οδηγήσει τις επιχειρήσεις στην ανάπτυξη μεθόδων και στρατηγικών επίλυσης για τη βελτιστοποίηση του κόστους των προβλημάτων δρομολόγησης οχημάτων. Στις επόμενες υποενότητες θα γίνει πλήρης αναφορά στις διάφορες περιπτώσεις δρομολόγησης οχημάτων.

2.2 Το Πρόβλημα Δρομολόγησης Οχημάτων

Τα προβλήματα που έχουν σαν αντικείμενο τη δρομολόγηση οχημάτων εντός των επιχειρησιακών δραστηριοτήτων και περιλαμβάνουν τόσο την παράδοση, όσο και την παραλαβή προϊόντων από τους πελάτες, είναι γνωστά ως **Προβλήματα Δρομολόγησης Οχημάτων (Vehicle Routing Problems)**. Τα προβλήματα αυτά σχετίζονται με τη διανομή των προϊόντων σ'ένα συγκεκριμένο αριθμό πελατών που είναι διασκορπισμένοι γεωγραφικά στο χώρο, από μια ή περισσότερες αποθήκες με ένα συγκεκριμένο αριθμό διαθέσιμων οχημάτων και οδηγών για μια δεδομένη χρονική στιγμή και μέσα σε συγκεκριμένο οδικό δίκτυο. Σκοπός της επίλυσης του προβλήματος δρομολόγησης οχημάτων είναι η εύρεση των διαδρομών εκείνων η εκτέλεση των οποίων ελαχιστοποιεί το συνολικό κόστος, που στη συγκεκριμένη περίπτωση είναι η συνολική απόσταση που θα διανύσουν όλα τα οχήματα περνώντας από όλους τους πελάτες. Ταυτόχρονα στόχος είναι η ελαχιστοποίηση του αριθμού των οδηγών και οχημάτων που θα χρησιμοποιηθούν. Η λύση του προβλήματος θα πρέπει να ικανοποιεί όλους τους τυχόν περιορισμούς που μπορεί να υπάρχουν. Πιο αναλυτικά οι περιορισμοί από τα χαρακτηριστικά των οχημάτων, των πελατών, των οδηγών και των διαδρομών:

- Μερικά χαρακτηριστικά των οχημάτων:
    1. Από ποια αποθήκη προέρχονται και αν υπάρχει πιθανότητα να τερματίσουν σε μια άλλη αποθήκη
    2. Η χωρητικότητα των οχημάτων
    3. Αν υπάρχουν οχήματα με εξειδικευμένα προϊόντα
    4. Αν υπάρχουν διαθέσιμα μηχανήματα για τη φόρτωση-εκφόρτωση
    5. Δρόμοι που μπορούν να προσπελαστούν από τα οχήματα
    6. Κόστος του κάθε οχήματος

- Μερικά χαρακτηριστικά των πελατών:
    1. Το σημείο που βρίσκεται ο πελάτης
    2. Η ποσότητα των αγαθών που πρέπει να παραλάβει ή να παραδώσει ο πελάτης
    3. Τα χρονικά περιθώρια που μπορεί να εξυπηρετηθεί ο πελάτης
    4. Ο χρόνος που απαιτείται για την παράδοση ή την παραλαβή του προϊόντος από τον πελάτη
    5. Το είδος του οχήματος που απαιτεί η εξυπηρέτηση ενός συγκεκριμένου πελάτη

- Μερικά χαρακτηριστικά των οδηγών
    1. Οκτάωρο ύπνου
    2. Απαγόρευση 10 ωρών συνεχούς οδήγησης
    3. Απαγόρευση 6 συνεχόμενων ημερών εργασίας
    4. Απαγόρευση 15 ωρών συνεχόμενης οδήγησης την ημέρα

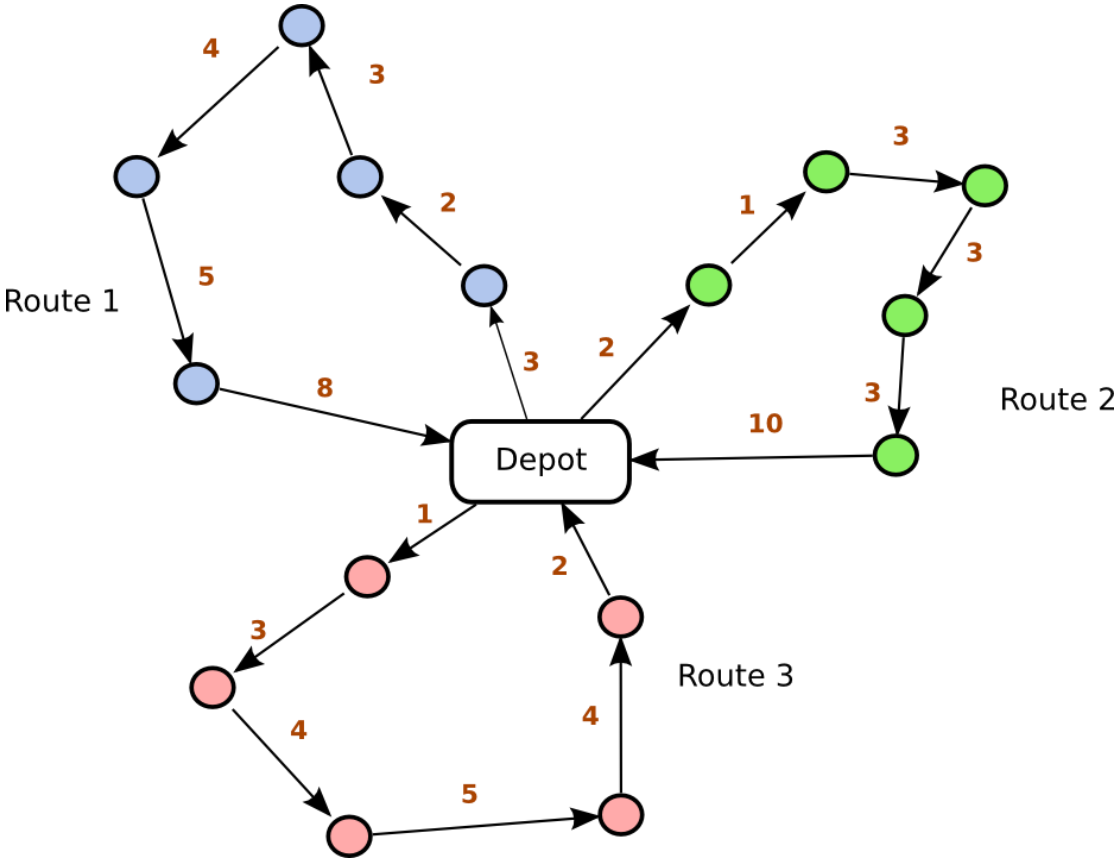
- Χαρακτηριστικά που αφορούν τις διαδρομές
    1. Η συνολική ποσότητα που μεταφέρει το όχημα δεν πρέπει να ξεπερνά τη χωρητικότητα του
    2. Υπάρχουν πελάτες που ζητούν παράδοση ή παραλαβή ή και τα δύο ταυτόχρονα
    3. Τα χρονικά παράθυρα εξυπηρέτησης που θέλουν οι πελάτες θα πρέπει να ικανοποιούνται
    4. Οι οδηγοί μπορεί να δουλεύουν κάποια συγκεκριμένη χρονική περίοδο
    5. Τα οχήματα μπορεί να μεταφέρουν παραπάνω από ένα προϊόντα

Από τα παραπάνω χαρακτηριστικά προκύπτουν και συγκεκριμένοι περιορισμοί που πρέπει να ληφθούν υπ'όψιν κατά την υλοποίηση και την αναζήτηση της βέλτιστης λύσης, ώστε η τελική λύση να είναι εφικτή. Οι στόχοι ορίζονται από την εκάστοτε επιχείρηση ανάλογα με το τι θέλει να πετύχει, τη ζήτηση, τις απαιτήσεις των πελατών και τις προτεραιότητες που η ίδια βάζει. Οι στόχοι μπορεί να είναι:

- Ελαχιστοποίηση του συνολικού κόστους μεταφοράς
- Ελαχιστοποίηση οχημάτων και οδηγών
- Η ελαχιστοποίηση του συνολικού χρόνου στις διαδρομές
- Η ελαχιστοποίηση της ενέργειας για τη μεταφορά των οχημάτων, όπως είναι η βενζίνη

Για την μοντελοποίηση του Προβλήματος Δρομολόγησης Οχημάτων αναπαριστάται γραφικά σε ένα δίκτυο όπου $G = (V,A)$, με $V = \{0,...,n\}$ είναι το σύνολο των πελατών και Α είναι το σύνολο των τόξων. Οι κόμβοι $i = 1,...,n$ είναι οι πιθανοί πελάτες και ο κόμβος 0 είναι η αποθήκη. Κάθε πελάτης i $(i = 1,...,n)$ απαιτεί μια μη αρνητική ποσότητα αγαθών i d , ενώ για την αποθήκη ισχύει οτι 0 $d_0 = 0$. Επιπλέον ορίζεται το

κόστος c_{ij}, όπου το c αντιστοιχεί στο τόξο *(i,j)* και αναπαριστά το κόστος μεταφοράς απο τον κόμβο (πελάτη) *i* στον κόμβο *j*. Πρέπει να σημειωθεί ότι δεν επιτρέπεται ένα τόξο να άπτεται και να καταλήγει στον ίδιο κόμβο και γι'αυτό ο πίνακας κόστους (c_{ij}) ορίζεται για $i \neq j$. Μια γραφική αναπαράσταση διαδρομών σε ένα πρόβλημα δρομολόγησης οχημάτων φαίνεται στο παρακάτω σχήμα.



Σχήμα 2.1: *Πρόβλημα δρομολόγησης οχημάτων - Vehicle Routing Problem* [2]

2.2.1 Πρόβλημα Δρομολόγησης Οχημάτων με Χρονικά Παράθυρα (Vehicle Routing Problem with Time Windows)

Το πρόβλημα δρομολόγησης οχημάτων με χρονικά παράθυρα (VRP) είναι μια παραλλαγή του γενικότερου προβλήματος δρομολόγησης οχημάτων. Στο συγκεκριμένο πρόβλημα οι πελάτες έχουν περισσότερα χαρακτηριστικά και για να επιτευχθεί λύση θα πρέπει να ικανοποιούνται οι περιορισμοί. Σε αυτή την κατηγορία προβλημάτων έχουμε ένα σύνολο πελατών γεωγραφικά διασκορπισμένους σε μια συγκεκριμένη περιοχή. Ο καθένας από αυτούς πρέπει να παραλάβει μια συγκεκριμένη ποσότητα φορτίου και η ιδιαιτερότητα του συγκεκριμένου προβλήματος ειναι ότι ο πελάτης πρέπει να εξυπηρετηθεί εντός ενός συγκεκριμένου χρονικού διαστήματος, το οποίο ονομάζεται χρονικό παράθυρο (Time Window). Πριν ή μετά το χρονικό παράθυρο δεν μπορεί να γίνει η εξυπηρέτηση του πελάτη. Δηλαδή ο κάθε πελάτης έχει έναν ενωρίτερο και έναν βραδύτερο χρόνο εξυπηρέτησης. Η εξυπηρέτηση του πελάτη γίνεται από ένα μόνο φορτηγό, μία μόνο φορά και θα πρέπει

να παραλάβει όλο το φορτίο. Όλα τα φορτηγά ξεκινούν τη διαδρομή τους από την αποθήκη τη χρονική στιγμή *t=0* και έχουν περιορισμένη χωρητικότητα. Επίσης η αποθήκη έχει και αυτή χρονικό παράθυρο στη λειτουργία της, δηλαδή μια χρονική στιγμή που κλείνει και τα φορτηγά θα πρέπει στο τέλος της διαδρομής να έχουν επιστρέψει όλα σε αυτήν. Σκοπός του προβλήματος είναι η εύρεση των διαδρομών εξυπηρέτησης πελατών, που ξεκινούν και τελειώνουν στην αποθήκη, χωρίς να γίνεται παραβίαση των περιορισμών του προβλήματος. Δηλαδή περιορισμοί χρόνου και χωρητικότητας. Σε κάθε παράδοση το φορτηγό αδειάζει το φορτίο, αυτό σημαίνει ότι η ζήτηση του πελάτη δεν πρέπει να υπερβαίνει το συνολικό διαθέσιμο φορτίο που απομένει. Σε αντίθετη περίπτωση το φορτηγό δεν πραγματοποιεί την εξυπηρέτηση.

Στο συγκεκριμένο πρόβλημα έχουμε αυστηρά χρονικά παράθυρα, που σημαίνει πως ο πελάτης θα πρέπει να έχει εξυπηρετηθεί μέσα σε αυτά. Στην περίπτωση που το όχημα φτάσει πριν τον ενωρίτερο χρόνο εξυπηρέτησης, το όχημα θα πρέπει να περιμένει στην τοποθεσία του πελάτη, μέχρι τη χρονική στιγμή που θα μπορεί να ξεκινήσει η εξυπηρέτηση. Στην περίπτωση που το όχημα φτάσει μετά τον βραδύτερο χρόνο εξυπηρέτησης σε έναν πελάτη, τότε δεν μπορεί να πραγματοποιήσει την παράδοση και ο πελάτης θα παραλάβει το φορτίο από άλλο όχημα.

Το πρόβλημα δρομολόγησης με χρονικά παράθυρα μπορεί να αναπαρασταθεί σε ένα γράφημα *G=(V,A)*, όπου η αποθήκη συμβολίζεται με τον κόμβο 0 όταν είναι σημείο εκκίνησης και *n+1* όταν είναι σημείο τερματισμού. Δεδομένου ότι η αποθήκη έχει χρονικά παράθυρα, θα ισχύει ότι $\{a_0, b_0\} = \{a_{n+1}, b_{n+1}\} = \{E,L\}$ με τα E και L να είναι η ελάχιστη πιθανή αναχώρηση από την αποθήκη και η αργότερη δυνατή άφιξη αντίστοιχα. Επίσης η αποθήκη όπως είναι φυσικό έχει μηδενικό χρόνο εξυπηρέτησης και μηδενική ζήτηση $d_0 = d_{n+1} = s_0 = s_{n+1} = 0$. Επίσης έχουμε τη χρονική μεταβλητή w_{ik} που δείχνει τη χρονική στιγμή που ξεκινάει η εξυπηρέτηση στον πελάτη i από το όχημα k και τη μεταβλητή

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

Η αντικειμενική συνάρτηση γίνεται:

$$Z = \min \sum_{ij} c_{ij} \sum_{k} x_{ijk}$$

Υπό

$$\sum_{k \in K} \sum_{j \in V} x_{ijk} = 1 \qquad \forall i \in N \qquad (1)$$

$$\sum_{i \in V - \{0\}} x_{0jk} = 1 \qquad \forall j \in N,\ k \in K \qquad (2)$$

$\sum_{i\in V-\{j\}} x_{ijk} - \sum_{i\in V-\{j\}} x_{ijk} = 0$ $\forall j \in N,\ k \in K$ (3)

$\sum_{i\in V-\{n+1\}} x_{in+1k} = 1$ $\forall k \in K$ (4)

$x_{ijk}\left(w_{ik} + s_i + t_{ij} - w_{jk}\right) \leq 0$ $\forall k \in K (i,j) \in A$ (5)

$a_i \sum_{j\in V} x_{ijk} \leq w_{ik} \leq b_i \sum_{j\in V} x_{ijk}$ $\forall i \in N, k \in K$ (6)

$E \leq w_{ik} \leq L$ $\forall i \in (0, n+1), k \in K$ (7)

$\sum_{i\in N} d_i \sum_{j\in V} x_{ijk} \leq C$ $\forall k \in K$ (8)

$x_{ijk} \geq 0$ $\forall k \in K, (i,j) \in A$ (9)

$x_{ijk} \in \{0,1\}$ $\forall k \in K, (i,j) \in A$ (10)

Ας εξηγήσουμε τις μεταβλητές, την αντικειμενική συνάρτηση και τους περιορισμούς που αναφέρθηκαν προηγούμενα. Οι κόμβοι των πελατών είναι για *i=1,.....,n* με *i=1* τον κόμβο της αφετηρίας. Το κόστος c_{ij} της διαδρομής από το i στο j και ισχύει ότι κόστος $c_{ij} = c_{j\iota}$, για κάθε $i \neq j$ και $(i,j) \in 1, \ldots\ldots\ldots, n$ ισχύει δηλαδή αρχή της συμμετρίας και η τριγωνική ανισότητα $c_{ij} \leq c_{ik} +, \mathrm{c_{kj}}, i \neq j \neq k, (i,j,k) = 1, \ldots, n.$ Επίσης όπου d_iη ζήτηση του κάθε πελάτη και C η συνολική ποσότητα προϊόντος. Η αντικειμενική συνάρτηση εκφράζει το συνολικό κόστος που θέλουμε να μειώσουμε. Ο (1) περιορίζει τον πελάτη σε ένα μόνο όχημα, οι περιορισμοί (2), (3), (4) εκφράζουν τη ροή στο μονοπάτι από το όχημα k. Οι περιορισμοί (5), (7), (8) είναι οι περιορισμοί που αφορούν τα χρονικά περιθώρια και τη χωρητικότητα και ο περιορισμός (6) δίνει την τιμή 0 στη χρονική μεταβλητή w_{ik}, εάν το όχημα δεν επισκέπτεται τους πελάτες *i,j* σε αυτή τη διαδρομή. [9]

2.2.2 Άλλα Προβλήματα Δρομολόγησης Οχημάτων

Μία από τις βασικότερες μορφές Δρομολόγησης Οχημάτων, είναι αυτή με τα χρονικά παράθυρα που αναφέρθηκε προηγούμενα. Ωστόσο υπάρχει μια πληθώρα παραλλαγών για το Πρόβλημα Δρομολόγησης Οχημάτων. Στη συνέχεια αναφέρονται οι πιο συνηθισμένες παραλλαγές του προβλήματος:

- **Δρομολόγηση Στόλου Οχημάτων με Χωρητικότητες (Capacitated Vehicle Routing Problem - CVRP)**

Είναι η άμεση επέκταση του βασικού μοντέλου. Σε αυτό το μοντέλο, τα οχήματα έχουν μία μέγιστη χωρητικότητα Q , την οποία δεν μπορούν να υπερβούν σε καμία περίπτωση.

- **Δρομολόγηση Στόλου Οχημάτων με Παράδοση και Παραλαβή (Vehicle Routing Problem with Pickup and Delivery - VRPPD)**

Σε αυτό το μοντέλο, τα οχήματα πρέπει να εξυπηρετήσουν κάποιους πελάτες και να παραλάβουν. Υπάρχει περίπτωση ένας πελάτης να επιθυμεί ταυτόχρονα και παραλαβή και παράδοση κάποιων αγαθών.

- **Δρομολόγηση Στόλου Οχημάτων με Παραλαβές (Vehicle Routing Problem with Backhauls - VRPB)**

Αυτό το μοντέλο είναι παρόμοιο με το VRPPD αλλά η διαφορά έγκειται στο γεγονός ότι πρώτα πρέπει να γίνουν όλες οι παραδόσεις και στη συνέχεια να γίνουν οι παραλαβές από τους πελάτες (εάν υπάρχουν).

- **Δρομολόγηση Στόλου Οχημάτων με Πολλαπλές Διαδρομές (Vehicle Routing Problem with Multiple Trips - VRPMT)**

Σε αυτό το μοντέλο, το ίδιο όχημα μπορεί να εκτελέσει παραπάνω από μία διαδρομές. Και εδώ τα οχήματα έχουν μία μέγιστη χωρητικότητα Q, την οποία δεν μπορούν να υπερβούν σε καμία περίπτωση.

- **Ανοιχτή Δρομολόγηση Στόλου Οχημάτων (Open Vehicle Routing Problem - OVRP)**

Σε αυτό το μοντέλο, τα οχήματα δεν είναι υποχρεωμένα να επιστρέψουν στην αποθήκη. Αυτό το σενάριο μπορεί να συμβεί αν μισθωθεί ένας στόλος οχημάτων για να κάνουν την παράδοση των αγαθών στους πελάτες.

- **Δρομολόγηση Στόλου Οχημάτων με Ξεχωριστές Παραδόσεις (Vehicle Routing Problem with Split Delivery - VRPSD)**

Σε αυτό το μοντέλο, κάθε πελάτης μπορεί να εξυπηρετηθεί από παραπάνω από ένα οχήματα. Κάθε όχημα που εξυπηρετεί έναν πελάτη μπορεί να μεταφέρει ένα ποσοστό των αγαθών που πρέπει να παραδοθούν σε αυτόν. Στο τέλος κάθε πελάτης πρέπει να λάβει την ποσότητα των αγαθών που παρήγγειλε.

- **Δυναμική Δρομολόγηση Στόλου Οχημάτων (Dynamic Vehicle Routing Problem - DVRP)**

Σε αυτό το μοντέλο, μερικές παραγγελίες είναι γνωστές εκ των προτέρων αλλά μπορεί να προκύψουν μερικές νέες παραγγελίες που πρέπει να ενσωματωθούν άμεσα στις παλιές. Από την άλλη μεριά, κάποιος πελάτης μπορεί να ακυρώσει ξαφνικά την προγραμματισμένη παραγγελία του, κάτι που θα πρέπει φυσικά να ληφθεί υπόψιν. Στην βιβλιογραφία μπορεί να αναφέρεται και με τον όρο Online Vehicle Routing Problem - OVRP.

- **Περιοδική Δρομολόγηση Στόλου Οχημάτων (Period Vehicle Routing Problem - PVRP)**

Σε αυτό το μοντέλο, πρέπει το πρόβλημα της Δρομολόγησης Στόλου Οχημάτων να επιλυθεί για ένα χρονικό διάστημα. Για παράδειγμα, οι διαδρομές που θα δημιουργηθούν προς τους πελάτες θα πρέπει να κρατάνε για μία εβδομάδα, για ένα μήνα κ.ο.κ.

- **Δρομολόγηση Στόλου Οχημάτων με Πολλαπλές Αποθήκες (Multi Depot Vehicle Routing Problem - MDVRP)**

Σε αυτό το μοντέλο λαμβάνονται υπόψιν παραπάνω από μία αποθήκες. Τα οχήματα μπορούν να ξεκινάνε και να καταλήγουν στην ίδια αποθήκη ή μπορούν να ξεκινάνε από μία αποθήκη a και να καταλήγουν σε μία αποθήκη b.

- **Δρομολόγηση Στόλου Οχημάτων με Στοχαστική Ζήτηση (Vehicle Routing Problem with Stohastic Demands - VRPSTD)**

Σε αυτό το μοντέλο, η ζήτηση σε κάθε πελάτη δεν είναι γνωστή εκ των προτέρων αλλά υποθέτουμε ότι ακολουθεί κάποια κατανομή. Το σενάριο αυτό μπορεί να συμβεί για παράδειγμα μετά από κάποια φυσική καταστροφή (σεισμός, πλημμύρα). Η βοήθεια που πρέπει να φτάσει σε διάφορες περιοχές δεν είναι γνωστή εκ των προτέρων. Επομένως αν η ζήτηση είναι μεγάλη ένα όχημα θα πρέπει να επιστρέψει στην αποθήκη χωρίς να έχει εξυπηρετήσει όλους τους πελάτες που του είχαν ανατεθεί. Θα ανεφοδιαστεί και θα συνεχίσει την εξυπηρέτηση των πελατών που μείνανε.

- **Χρονοεξαρτώμενη Δρομολόγηση Στόλου Οχημάτων (Time Dependent Vehicle Routing Problem - TDVRP**

Σε αυτό το μοντέλο ο χρόνος ταξιδιού μεταξύ δύο πελατών *i, j* δεν είναι σταθερός και εξαρτάται από την ώρα που θα ξεκινήσει το όχημα. Με αυτό τον τρόπο λαμβάνεται υπόψιν η κίνηση του οδικού δικτύου και αυξάνεται η πολυπλοκότητα του προβλήματος.

- **Δρομολόγηση Ετερογενούς Στόλου Οχημάτων (Heterogeneous Fleet Vehicle Routing Problem - HFVRP)**

Σε αυτό το μοντέλο, υποθέτουμε ότι υπάρχει διαθέσιμος ένας στόλος οχημάτων με διαφορετικά χαρακτηριστικά. Η διαφορά μπορεί να βρίσκεται στην χωρητικότητα κάθε οχήματος, στο μέγεθος των οχημάτων ή στο κόστος λειτουργίας τους (κατανάλωση καυσίμου, κόστος μίσθωσης οδηγού).

- **Δρομολόγηση Στόλου Οχημάτων με Εξισορρόπηση Διαδρομών (Vehicle Routing Problem with Route Balancing - VRPRB)**

Σε αυτό το μοντέλο, εκτός από την κύρια συνάρτηση ελαχιστοποίησης του συνολικού κόστους των διαδρομών, υπάρχει και μία δεύτερη συνάρτηση που πρέπει να ελαχιστοποιηθεί. Αυτή είναι η διαφορά μεταξύ της μεγαλύτερης σε μήκος διαδρομής μείον την μικρότερη σε μήκος διαδρομή. Αυτός ο επιπλέον περιορισμός είναι σημαντικός καθώς έχει να κάνει με ζητήματα ίσης κατανομής φόρτου εργασίας μεταξύ των οδηγών. Για παράδειγμα, δεν μπορεί ένας οδηγός να τελειώνει τη βάρδια του σε μία ώρα (έχοντας μια μικρή σε μήκος διαδρομή να εξυπηρετήσει) και ένας άλλος σε τρεις ώρες (έχοντας να καλύψει μια μεγάλη σε μήκος διαδρομή).

- **Εμπλουτισμένη Δρομολόγηση Στόλου Οχημάτων (Rich Vehicle Routing Problem - RVRP)**

Τα τελευταία χρόνια καθώς εμφανίζονται πολλές παραλλαγές που περιλαμβάνουν πολλαπλές αποθήκες, πολλαπλά παράθυρα χρόνου, ετερογενείς στόλους οχημάτων, πολλαπλές διαδρομές για κάθε όχημα, παράδοση και παραλαβή αγαθών στους πελάτες κ.ο.κ., έχει εμφανισθεί στη βιβλιογραφία το μοντέλο της Εμπλουτισμένης Δρομολόγησης Στόλου Οχημάτων που περιλαμβάνει όλους αυτούς τους περιορισμούς. [8]

2.3 Αλγόριθμοι Εύρεσης Βέλτιστης Λύσης

Σε αυτό το κεφάλαιο παρουσιάζονται και αναλύονται κάποιοι αλγόριθμοι που βοηθούν να ξεφύγει μία λύση από κάποιο τοπικό ελάχιστο. Η μέθοδος τοπικής αναζήτησης που χρησιμοποιούμε συνήθως θα συγκλίνει γύρω από κάποιο τοπικό ελάχιστο. Επιπλέον, ο αλγόριθμος είναι πολύ ευαίσθητος στην αρχική λύση. Δηλαδή αν η αρχική λύση είναι πολύ καλή και βρίσκεται πολύ κοντά σε κάποιο τοπικό ελάχιστο τότε με τη διαδικασία της τοπικής αναζήτησης θα βρεθεί πολύ εύκολα το τοπικό ελάχιστο, αλλά δεν ξέρουμε αν μετά ο αλγόριθμος παγιδευτεί σε αυτό το τοπικό ελάχιστο και δεν μπορεί να βελτιώσει άλλο τη λύση. Για να επιλυθεί αυτό το πρόβλημα έχουν προταθεί πολλοί αλγόριθμοι που βοηθούν την αναζήτηση να ξεφύγει από κάποιο τοπικό ελάχιστο. Οι αλγόριθμοι αυτοί ονομάζονται μεθευρετικοί αλγόριθμοι. 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. 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).*** [6]

Στη συνέχεια αναλύονται κάποιοι από τους αλγορίθμους που αναφέρθηκαν προηγούμενα.

2.3.1 Διαδικασία Άπληστης Τυχαιοποιημένης Προσαρμοστικής Αναζήτησης (Greedy Randomized Adaptive Search Procedure (GRASP))

Η διαδικασία άπληστης τυχαιοποιημένης προσαρμοστικής αναζήτησης είναι μια επαναληπτική διαδικασία για την εύρεση προσεγγιστικών λύσεων σε προβλήματα συνδυαστικής βελτιστοποίησης. Αυτή η τυχαιοποιημένη τεχνική παρέχει μια εφικτή λύση σε κάθε επανάληψη. Οι επαναλήψεις της διαδικασίας 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 αναζήτησης (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. Η φάση κατασκευής μπορεί να περιγραφεί ως η 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 CandidateList) για εισαγωγή στη λύση στην οποία κάθε στοιχείο κατατάσσεται βάσει μιας συνάρτησης απληστίας. Η τυχαία επιλογή του στοιχείου σημαίνει ότι δεν είναι ανάγκη να επιλεγεί το πρώτο στοιχείο στη λίστα. Ο ευρετικός αλγόριθμος είναι προσαρμοστικός με την έννοια ότι η συνάρτηση επιλογής προσαρμόζεται για να προσμετρήσει τα στοιχεία που έχουν ήδη επιλεγεί. 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. Έχουν παρουσιαστεί δύο τρόποι για την κατασκευή της λίστας περιορισμού των υποψηφίων. Στον πρώτο τρόπο η λίστα πάντα έχει ένα καθορισμένο μήκος. Στη δεύτερη περίπτωση για να μπει κάποιο στοιχείο στη λίστα πρέπει η τιμή του να είναι μικρότερη από κάποια τιμή κατωφλίου. Αν υποθέσουμε ότι με $\boldsymbol{c_{min}}$ και $\boldsymbol{c_{max}}$ συμβολίζουμε τις μέγιστες και τις ελάχιστες τιμές που θα μπορούσε να έχει ένα στοιχείο που θα συμπεριλαμβανόταν στη λύση τότε για να μπει κάποιο στοιχείο στη λίστα θα πρέπει η τιμή του να είναι ανάμεσα στο διάστημα$[c_{min}, c_{min} + \boldsymbol{\alpha} \times (c_{max} - c_{min})]$. Το **α** είναι μία παράμετρος που δίνεται από το χρήστη και είναι πάρα πολύ σημαντικό διότι αν είναι ίση με **1** τότε ο αλγόριθμος είναι καθαρά ένας τυχαιοποιημένος αλγόριθμος αφού μπορεί να επιλεγεί οποιοδήποτε από τα στοιχεία, ενώ αν είναι ίσο με το **0** τότε είναι καθαρά άπληστος αλγόριθμος γιατί μόνο το βέλτιστο στοιχείο θα εισέλθει. Εννοείται ότι όσο μικρότερο είναι το **α** τόσο λιγότερα είναι τα στοιχεία που έχουν πιθανότητες να εισέλθουν στη λύση. Υπάρχουν τρεις τρόποι να επιλεγεί το **α.** Ο πρώτος τρόπος είναι ο στατικός όπου στο ξεκίνημα του αλγορίθμου επιλέγεται η τιμή του **α**. Στο δεύτερο τρόπο η τιμή του **α** επιλέγεται δυναμικά όπου αλλάζει στο ξεκίνημα της κάθε επανάληψης. Ο τρίτος τρόπος είναι να επιλεγεί προσαρμοστικά, όπου κάθε φορά και για κάθε στοιχείο παίρνει και μία διαφορετική τιμή. Η λύση που δημιουργείται στη φάση κατασκευής δεν εγγυάται ότι είναι τοπικό ελάχιστο, και για αυτό το λόγο μια φάση τοπικής αναζήτησης εφαρμόζεται για να παράξει μια λύση που να περιέχει τοπικό ελάχιστο. Για να εφαρμοστεί η φάση τοπικής αναζήτησης, καθορίζεται μια συνάρτηση που να κάνει αναζήτηση στη γειτονιά της αρχικής λύσης. Φυσικά, διαφορετικά προβλήματα χρειάζονται διαφορετικές συναρτήσεις απληστίας, διαφορετικές διαδικασίες τοπικής αναζήτησης για την εύρεση του τοπικού ελαχίστου όπως και διαφορετικές συναρτήσεις γειτονιάς που θα γίνει η αναζήτηση. Όταν αυτά καθοριστούν τότε το μόνο που μας ενδιαφέρει για να εξετάσουμε στη διαδικασία άπληστης τυχαιοποιημένης προσαρμοστικής αναζήτησης είναι το μέγεθος της λίστας και τον αριθμό των επαναλήψεων που χρειάζεται. Πολλές φορές για να βελτιωθεί η απόδοση της Διαδικασίας Άπληστης Τυχαιοποιημένης Προσαρμοστικής Αναζήτησης μπορεί να χρησιμοποιηθεί ακόμα μία φάση όπου η καλύτερη ή κάποιες από τις καλύτερες (εκλεκτές - ελίτ) λύσεις συνδυάζονται με την καινούρια λύση με στόχο να επιτευχθούν ακόμα καλύτερες λύσεις. Ο αλγόριθμος που συνήθως

χρησιμοποιείται σε αυτή τη διαδικασία είναι ο αλγόριθμος επανασύνδεσης διαδρομών. Μια γενικευμένη μορφή του αλγορίθμου παρουσιάζεται στη συνέχεια.

Διαδικασία ΄Απληστης Τυχαιοποιημένης Προσαρμοστικής Αναζήτησης

Αρχικοποίηση

$c(s^*) = \infty$

repeat

Κατασκευή μιας αρχικής λύσης s

Εφαρμογή τοπικής αναζήτησης στην s

if $c(s) < c\ (s^*)$ *then*

$s^* = s$

endif

until για όσο το κριτήριο τερματισμού δεν ικανοποιείται

Επέστρεψε τη βέλτιστη λύση (s^*).

2.3.2 Άλλοι Αλγόριθμοι Βελτιστοποίησης

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

Ο πρώτος που πρότεινε την χρησιμοποίηση της προσομοιωμένης ανόπτησης (Simulated Annealing) ως μέθοδο για την επίλυση προβλημάτων συνδυαστικής βελτιστοποίησης ήταν ο Kirkpatrick. Το όνομα του αλγόριθμου ή καλύτερα η στρατηγική, προέρχεται από την αναλογία ανάμεσα στην προσομοίωση της ανόπτησης των υλικών και τη στρατηγική επίλυσης προβλημάτων συνδυαστικής βελτιστοποίησης. Στη συνδυαστική βελτιστοποίηση αυτό αναπαριστάται με την κίνηση στον εφικτό χώρο αναζήτησης και όταν ο αλγόριθμος σταματήσει έχουμε βρει μια εφικτή λύση. Στην προσομοιωμένη ανόπτηση, η γειτονιά *N(s)* καθορίζει καινούριες καταστάσεις που μπορεί να τις φθάσει κάποιος από την τρέχουσα κατάσταση. Από την αρχική κατάσταση s_0 με μια τυχαία διαταραχή του συστήματος γεννιέται μια καινούρια λύση s'. Το αποτέλεσμα της αλλαγής στην τιμή της αντικειμενικής συνάρτησης υπολογίζεται ως εξής:

$$\boldsymbol{\delta = f(s') - f(s_0)}$$

Εάν το $\delta<0$ η καινούρια κατάσταση είναι πάντοτε αποδεκτή. Εάν η αλλαγή δεν είναι αρνητική, δηλαδή $\delta \geq 0$, η καινούρια κατάσταση είναι αποδεκτή με κάποια πιθανότητα. Η λογική υπολογισμού της πιθανότητας προέρχεται από τους νόμους της

θερμοδυναμικής. Έτσι, έχουμε στη θερμοκρασία t, η πιθανότητα της αύξησης στην ενέργεια της ποσότητας δ δίνεται από τη σχέση:

$$\boldsymbol{p(\delta) = e^{\frac{-\delta}{kt}},}$$

όπου το k είναι η σταθερά Boltzmann (η οποία δεν έχει κανένα απολύτως νόημα στη βελτιστοποίηση και γι'αυτό το λόγο αγνοείται). Αυτό σημαίνει ότι η προσομοιωμένη ανόπτηση προσφέρει ένα μηχανισμό για την αποδοχή μικρών αυξήσεων στην τιμή της αντικειμενικής συνάρτησης, ελέγχοντας την πιθανότητα της αποδοχής *p(δ)*, της νέας αντικειμενικής τιμής δια μέσου της θερμοκρασίας. Έτσι, αναφέρεται ότι σε υψηλότερες θερμοκρασίες είναι πιο πιθανό μια λύση με χειρότερη τιμή στη συνάρτηση κόστους να είναι αποδεκτή ως η καινούρια κατάσταση του προβλήματος. Σε κάθε επανάληψη του αλγορίθμου εκτός από την τρέχουσα λύση αποθηκεύεται και η καλύτερη λύση που έχει βρεθεί ως εκείνη τη στιγμή. Ο λόγος που γίνεται αυτό είναι γιατί από τη στιγμή που δεχόμαστε χειρότερες λύσεις είναι πολύ πιθανό το βέλτιστο να χαθεί όταν χειροτερέψει μια λύση.

- **Αλγόριθμος Επανασύνδεσης Διαδρομών (Path Relinking (PR))**

Η μέθοδος της επανασύνδεσης διαδρομών δημιουργεί καινούριες λύσεις με την εξέταση διαφορετικών τροχιών που συνδέουν υψηλής ποιότητας λύσεις. Ο αλγόριθμος ξεκινάει από μία από αυτές τις λύσεις που ονομάζεται εναρκτήρια λύση (starting solution) και λύση στόχου ή τελική λύση (target solution). Η βασική ιδέα του αλγορίθμου επανασύνδεσης διαδρομών είναι να δημιουργηθούν και να εξερευνηθούν τροχιές στο χώρο λύσεων που συνδέουν μία αρχική λύση s και μία τελική λύση t. Η ιδέα είναι να μετατρέψουμε το γραμμικό συνδυασμό των σημείων στον Ευκλείδιο χώρο λύσεων σε διαδρομές ανάμεσα και πέρα από τις συγκεκριμένες λύσεις στο γειτονικό χώρο αναζήτησης. Η διαδρομή ανάμεσα στις δύο λύσεις θα οδηγήσει σε λύσεις που μοιράζονται κοινά καλά χαρακτηριστικά με τις αρχικές λύσεις. Μία ακολουθία από γειτονικές λύσεις δημιουργούνται και από αυτές ο αλγόριθμος επιστρέφει την καλύτερη. Ο βασικός στόχος είναι στο τέλος των επαναλήψεων η λύση στόχος και η λύση που δημιουργείται να είναι ίδιες δηλαδή η απόσταση μεταξύ τους *(dist(s,t))* να είναι μηδέν. Κατά τη διάρκεια του σχεδιασμού των διαδρομών πρέπει να λάβουμε υπόψη πως θα συνδεθούν οι δύο λύσεις. Άρα, πρέπει να βρεθεί ένας κατάλληλος αλγόριθμος που να μπορεί να συνδέσει τις λύσεις και να μπορεί να μειώσει την απόσταση *(dist(s,t))* (η απόσταση ανάμεσα στις δύο λύσεις εξαρτάται από το πρόβλημα που θέλουμε να επιλύσουμε). Ο ευρετικός αλγόριθμος που θα χρησιμοποιηθεί για να συνδέσει τις δύο διαδρομές θα πρέπει να μην χρειάζεται πολύ μεγάλο χρόνο εκτέλεσης ώστε να μη χειροτερέψει υπολογιστικά ο αλγόριθμος.

- **Μέθοδος Αποδοχής Κατωφλίου (Threshold Accepted (TA))**

Η μέθοδος της αποδοχής κατωφλίου ξεφεύγει από το τοπικό ελάχιστο χρησιμοποιώντας ένα κατώφλι αποδοχής T ως πάνω όριο στην επιτρεπόμενη αύξηση

της τιμής της αντικειμενικής συνάρτησης από τη μία κίνηση στην επόμενη. Έτσι, στη μέθοδο αποδοχής κατωφλίου, η γειτονιά *N(s)* καθορίζει καινούριες καταστάσεις που μπορεί να τις φθάσει κάποιος από την τρέχουσα κατάσταση s. Από την αρχική κατάσταση s_0 με μια τυχαία διαταραχή του συστήματος γεννιέται μια καινούρια λύση s'. Το αποτέλεσμα της αλλαγής στην τιμή της αντικειμενικής συνάρτησης υπολογίζεται ως εξής:

$$\boldsymbol{\delta = f(s') - f(s_0)}$$

Εάν το $\delta<Ti$ η καινούρια κατάσταση είναι πάντοτε αποδεκτή, όπου το *Ti* είναι το κατώφλι αποδοχής στην επανάληψη i. Το κατώφλι αποδοχής μπορεί να είναι σταθερό σε όλες τις επαναλήψεις ή να μειώνεται κατά τη διάρκεια των επαναλήψεων.

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

Ο μεθευρετικός αλγόριθμος της περιορισμένης αναζήτησης, είναι ίσως ο πιο γνωστός μεθευρετικός αλγόριθμος και αρχικά παρουσιάστηκε από τον Glover. Η περιορισμένη αναζήτηση χρησιμοποιεί ένα ευρετικό αλγόριθμο για να μετακινηθεί από τη μια λύση στην άλλη. Όμως, όπως και στους άλλους μεθευρετικούς αλγορίθμους, υπάρχει η πιθανότητα η λύση να παγιδευτεί σε τοπικό ελάχιστο. Η σημαντική διαφορά της περιορισμένης αναζήτησης από τη στρατηγική της προσομοιωμένης ανόπτησης είναι ότι για να ξεφύγει η λύση από το τοπικό ελάχιστο χρησιμοποιείται μια συγκεκριμένη στρατηγική για την επιλογή της επόμενης λύσης και δεν γίνεται τυχαία επιλογή της επόμενης λύσης. Η στρατηγική αυτή χρησιμοποιεί μνήμη (αντί για πιθανότητα). Όταν αναφέρεται μνήμη εννοούμε μνήμη από τις προηγούμενες κινήσεις που έχουν πραγματοποιηθεί. Για να αποφευχθούν οι συνεχείς επιστροφές σε προηγούμενες λύσεις (δηλαδή για να αποφευχθούν οι επαναλαμβανόμενοι κύκλοι γύρω από μια ομάδα λύσεων) οι τελευταίες κινήσεις καταγράφονται σε μια λίστα, η οποία ονομάζεται λίστα περιορισμένων κινήσεων (tabu list) και οι συγκεκριμένες κινήσεις απαγορεύεται να επιστρέψουν στη λύση για ένα συγκεκριμένο αριθμό επαναλήψεων.

- **Γενετικοί Αλγόριθμοι (Genetic Algorithms)**

Οι Γενετικοί Αλγόριθμοι αποτελούν μία μέθοδο αναζήτησης λύσεων και είναι χρήσιμοι σε προβλήματα που περιέχουν πολλές παραμέτρους ή διαστάσεις και δεν υπάρχει αναλυτική μέθοδος που να μπορεί να βρει το βέλτιστο συνδυασμό. Οι γενετικοί αλγόριθμοι βασίζονται σε μια μίμηση της βιολογικής διαδικασίας στην οποία αναπτύσσονται νέοι πληθυσμοί μεταξύ διαφορετικών ειδών κατά τη διάρκεια της εξέλιξης, μέσω της διασταύρωσης, της γενετικής μετάλλαξης και της φυσικής επιλογής. Ένας γενετικός αλγόριθμος είναι μία στοχαστική επαναληπτική διαδικασία κατά τη διάρκεια των οποίων διατηρεί το μέγεθος του πληθυσμού σταθερό και ονομάζεται γενιά (generation). Κατά τη διάρκεια της αναζήτησης βέλτιστης λύσης χρησιμοποιούν πληροφορίες από έναν αριθμό λύσεων που ονομάζονται άτομα. Για να δημιουργηθεί η νέα λύση εφαρμόζονται ένας δυαδικός τελεστής που ονομάζεται διασταύρωση (crossover) και ένας μοναδιαίος που ονομάζεται μετάλλαξη (mutation). Η διασταύρωση παίρνει δύο άτομα που ονομάζονται γονείς (parents) και παράγει δύο

νέα που ονομάζονται απόγονοι (offspring). Οι λύσεις που βρίσκονται κοντά στο επιθυμητό αποτέλεσμα, αναπαράγονται σε επόμενη γενιά λύσεων όπου θα δεχτούν μια τυχαία μετάλλαξη. Μετά από έναν αριθμό επαναλήψεων οι τυχαίες μεταλλάξεις σε συνδυασμό με την επιβίωση ή αναπαραγωγή των λύσεων που είναι κοντά στο επιθυμητό αποτέλεσμα, θα παράξουν το τελικό βέλτιστο αποτέλεσμα. Ο γενετικός κώδικας συνήθως αναπαριστάται με δυαδικά στοιχεία.

- **Αλγόριθμος Βελτιστοποίησης Αποικίας Μυρμηγκιών**

Η Βελτιστοποίηση Αποικίας Μυρμηγκιών (Ant Colony Optimization(ACO)) είναι μια σχετικά νέα στρατηγική για την επίλυση προβλημάτων συνδυαστικής βελτιστοποίησης που πρωτοπαρουσιάστηκε από τους Dorigo, Maniezzo και Colorni. Η Βελτιστοποίηση Αποικίας Μυρμηγκιών είναι ένα σύστημα που μιμείται τη συμπεριφορά των πραγματικών μυρμηγκιών κατά τη διαδικασία της εύρεσης της τροφής τους. Τα μυρμήγκια αναπτύσσουν μια τεχνική για να βρουν τη συντομότερη διαδρομή από τη φωλιά τους προς την πηγή της τροφής τους και αντιθέτως. Τα μυρμήγκια ξεκινούν την αναζήτηση της τροφής γύρω από την πηγή με τυχαίο τρόπο και καθώς κινούνται αφήνουν πίσω τους μια ποσότητα μιας ουσίας που ονομάζεται φερομόνη και με αυτό τον τρόπο μαρκάρουν το μονοπάτι που έχουν διανύσει. Η ποσότητα της φερομόνης στο κάθε μονοπάτι εξαρτάται από την απόσταση, την ποιότητα και την ποσότητα της τροφής που βρέθηκε. Το επόμενο μυρμήγκι που θα φύγει από τη φωλιά του είναι πολύ πιθανό να ακολουθήσει τη φερομόνη που θα υπάρχει σε κάποιο μονοπάτι, αφήνοντας μια ποσότητα φερομόνης στο ίδιο μονοπάτι. Καθώς η ποσότητα φερομόνης στο συγκεκριμένο μονοπάτι όλο και αυξάνεται όλο και περισσότερα μυρμήγκια ακολουθούν αυτό το μονοπάτι. Όμως καθώς η ώρα περνάει η φερομόνη, ιδιαίτερα από τα μονοπάτια που δεν πηγαίνουν πολλά μυρμήγκια, ελαττώνεται. Τελικά, από όλα τα υπόλοιπα μονοπάτια η φερομόνη εξαφανίζεται και όλα τα μυρμήγκια ακολουθούν τελικά το ίδιο μονοπάτι, που είναι και η βέλτιστη ή η σχεδόν βέλτιστη λύση. [6]

Κεφάλαιο 3
Μοντελοποίηση και Επίλυση του Προβλήματος Δρομολόγησης Οχημάτων

3.1 Εισαγωγή

Στο Κεφάλαιο αυτό περιγράφουμε τα δεδομένα του προβλήματος δρομολόγησης οχημάτων προς επίλυση. Πιο συγκεκριμένα, παρουσιάζονται τα επιμέρους χαρακτηριστικά οχημάτων και πελατών καθώς και οι δοθέντες περιορισμοί. Με βάση τα δεδομένα αυτά, περιγράφεται η μοντελοποίηση του προβλήματος και αναπαράσταση των δεδομένων με τη χρήση προγραμματιστικού περιβάλλοντος MATLAB. Στη συνέχεια, αναλύονται τα επιμέρους βήματα της μεθόδου **GRASP**: (i) Το αρχικό στάδιο εύρεσης αρχικής λύσης με έναν απλό αλγόριθμο απληστίας και (ii) το τελικό στάδιο εύρεσης βέλτιστης λύσης με εφαρμογή τοπικής αναζήτησης. Ανά στάδιο, αναπτύσσονται οι χρησιμοποιημένες δομές δεδομένων, οι υλοποιημένες υπορουτίνες και οι αντίστοιχοι ψευδοκώδικες.

3.2 Μοντελοποίηση του Προβλήματος

3.2.1 Περιγραφή Δεδομένων Προβλήματος

Το πρόβλημα προς επίλυση περιγράφεται από τα επιμέρους σενάρια των προβλημάτων του Solomon[7], όπως αυτά δίνονται στα αντίστοιχα αρχεία κειμένου. Ανά αρχείο υπάρχουν τα αντίστοιχα δεδομένα που αφορούν την περιγραφή (i) της αποθήκης, (ii) των οχημάτων και (iii) των πελατών. Με βάση την ύπαρξη χρονικών παραθύρων, η σημαντικότερη ιδιαιτερότητα του προβλήματος είναι ότι κάθε σημείο (πελάτες και αποθήκη) έχει ένα συγκεκριμένο χρονικό παράθυρο που μπορεί να δεχθεί εξυπηρέτηση, ενώ δεν μπορούν οποιαδήποτε άλλη χρονική στιγμή. Αναλυτικότερα ανά περίπτωση, έχουμε τα εξής χαρακτηριστικά:

i. **Αποθήκη**: Όπως έχουμε ήδη αναφέρει, στην περίπτωσή μας θεωρούμε ότι διαθέτουμε μία αποθήκη. Τα δεδομένα που σχετίζονται με την αποθήκη είναι:
   - Χρονικό Παράθυρο: Το παράθυρο της αποθήκης είναι ουσιαστικά ο συνολικός χρόνος λειτουργίας της. Συνεπώς ως νωρίτερο χρόνο (READY TIME) θεωρούμε την έναρξη λειτουργίας της, δηλαδή τη χρονική στιγμή *t=0* όπου και ξεκινούν ταυτόχρονα τα φορτηγά. Επιπλέον, ως έσχατο χρόνο (DUE TIME) θεωρούμε τη στιγμή κλεισίματος της, όταν και τα φορτηγά θα πρέπει να έχουν επιστρέψει εγκαίρως
   - Συντεταγμένες Αποθήκης: Το σημείο συντεταγμένων (XCOORD., YCOORD.) που αντιστοιχεί στην τοποθεσία της αποθήκης

ii. **Οχήματα**: Έχουμε στη διάθεσή μας έναν πεπερασμένο αριθμό οχημάτων για την εξυπηρέτηση όλων των πελατών. Κάθε όχημα περιγράφεται από τα εξής χαρακτηριστικά:
   - Αριθμός Οχήματος: Επιλέγουμε απλή αρίθμηση των διαθέσιμων οχημάτων, όπου *i=1* το πρώτο φορτηγό, *i=2* το δεύτερο κ.ο.κ.

- Χωρητικότητα: Η συνολική διαθέσιμη χωρητικότητα κάθε οχήματος

iii. **Πελάτες**: Όπως αναφέραμε, βασικό χαρακτηριστικό ανά πελάτη είναι η ύπαρξη χρονικών παραθύρων. Αναλυτικότερα, για τους πελάτες έχουμε τα εξής δεδομένα προβλήματος:
   - Αριθμός Πελάτη: Με βάση τα διαθέσιμα αρχεία κειμένου των προβλημάτων του Solomon, έχουμε την αρίθμηση των πελατών, με το 0 να αντιστοιχεί στην αποθήκη. Παρόλα αυτά, το περιβάλλον της MATLAB δε χρησιμοποιεί ως τιμή μεταβλητών indexes το 0, ξεκινάμε την αρίθμηση από το 1, με το 1 να αντιστοιχεί στην αποθήκη, το 2 στον 1ο πελάτη, το 3 στο 2ο πελάτη κ.ο.κ
   - Συντεταγμένες: Όπως και για την αποθήκη, παρέχονται τα σημεία συντεταγμένων (XCOORD.,YCOORD.) που αντιστοιχούν στην τοποθεσία του κάθε πελάτη
   - Ζήτηση: Η ζήτηση (DEMAND) ανά πελάτη
   - Χρονικό παράθυρο: Οι χρονικές τιμές για την νωρίτερη και έσχατη εξυπηρέτηση κάθε πελάτη (READY και DUE TIME αντίστοιχα)
   - Χρόνος Εξυπηρέτησης: Ο χρόνος που απαιτείται να παραμείνει στο πελάτη ένα όχημα, ώστε να ολοκληρωθεί η εξυπηρέτησή του

Πρέπει τώρα να εξετάσουμε ποιες μεταβλητές έχουμε να αρχικοποιήσουμε και ποιες χρειάζεται να δημιουργήσουμε στη συνέχεια. Με βάση τα δεδομένα που είναι διαθέσιμα στο αρχείο κειμένου του κάθε σεναρίου, μπορούμε άμεσα να διαβάσουμε το αρχείο μέσω MATLAB και να τα αποθηκεύσουμε στις αντίστοιχες μεταβλητές. Με βάση αυτές δημιουργούμε τις επιπλέον δομές δεδομένων που είναι απαραίτητες. Αναλυτικά οι αρχικές και επιπλέον μεταβλητές περιγράφονται στην επόμενη και μεθεπόμενη υποενότητα αντίστοιχα.

3.2.2 Αρχικοποιημένες Μεταβλητές από Δεδομένα Προβλήματος

Τα αρχικά δεδομένα διαβάζονται από το αρχείο κειμένου (αποθηκευμένο σε μορφή notepad) με την εντολή data=dlmread('data.txt'), όπου "data" το εκάστοτε όνομα του αρχείου. Με την εντολή αυτή οι στήλες του αρχείου αποθηκεύονται στη δομή "data". Για τα δεδομένα λοιπόν που παρουσιάστηκαν στην προηγούμενη υποενότητα, χρησιμοποιούνται οι εξής μεταβλητές:

- Cust_Num: Ο αριθμός των πελατών (στον οποίων συμπεριλαμβάνεται και η αποθήκη ως σημείο 1, με βάση την αρίθμηση που περιγράφηκε προηγουμένως)
- Ready_Time: Μονοδιάστατος πίνακας ελάχιστου χρονικού παραθύρου για κάθε πελάτη
- Due_Time: Μονοδιάστατος πίνακας μέγιστου χρονικού παραθύρου για κάθε πελάτη
- Cust_Coords: Δισδιάστατος πίνακας των συντεταγμένων x και y του κάθε πελάτη

- Cust_Demand: Μονοδιάστατος πίνακας ζήτησης του κάθε πελάτη
- Time_Service: Χρόνος εξυπηρέτησης του κάθε πελάτη
- Time_Max: Μέγιστος χρόνος που μπορεί να διαρκέσει μια διαδρομή. Συνεπώς, ο μέγιστος χρόνος υποδηλώνεται από το βραδύτερο χρόνο της αποθήκης, δηλαδή Time_Max = Due_Time(1)
- Veh_Cap: Χωρητικότητα κάθε οχήματος
- Veh_Num: Ο μέγιστος διαθέσιμος αριθμός οχημάτων

3.2.3 Επιπλέον Μεταβλητές για την Επίλυση του Προβλήματος

Στη συνέχεια δημιουργούμε επιπλέον δομές δεδομένων απαραίτητες για τη διαδικασία επίλυσης. Με δεδομένο ότι ο μεθευρετικός αλγόριθμος που θα χρησιμοποιήσουμε βασίζεται σε γειτονιά αναζήτησης, σημαντικό στοιχείο είναι ο πίνακας που θα περιλαμβάνει τις αποστάσεις μεταξύ όλων των κόμβων. Ο πίνακας αυτός συμβολίζεται ως "Cost", όπου Cost(i,j) η απόσταση (το κόστος μετακίνησης) από το σημείο *i* στο σημείο *j*. Προφανώς, η απόσταση που διανύουμε για να πάμε από έναν κόμβο *i* σε έναν επόμενο κόμβο *j*, ισούται με την απόσταση για να πάμε από τον κόμβο *j* στον κόμβο *i*. Κατά συνέπεια, ο πίνακας "Cost" είναι συμμετρικός. Επίσης, είναι προφανές ότι ο πίνακας Cost εμφανίζει όλες τις αποστάσεις μεταξύ των κόμβων, θα είναι λοιπόν διαστάσεων [Cust_Num x Cust_Num]. Ο πίνακας κατασκευάστηκε μέσω της συνάρτησης "create_cost_tables" με τα εξής χαρακτηρικιστικά:

Ορίσματα:

- Cust_Coords: Οι συντεταγμένες x και y όλων των κόμβων
- Cust_Num: Ο αριθμός των κόμβων

Αποτελέσματα:

- Cost: Οι αποστάσεις μεταξύ όλων τον κόμβων
- Cost_Temp: Αντίγραφο του Cost για μετέπειτα τροποποιήση

Οι αποστάσεις υπολογίζονται με βάση τη σχέση:

$$\text{Cost(i,j)}= \sqrt{(\text{Cust\_Coords}(i,1) - \text{Cust\_Coords}(j,1))^2 + (\text{Cust\_Coords}(i,2) - \text{Cust\_Coords}(j,2))^2}$$

Επόμενη σημαντική πληροφορία που πρέπει να διατηρούμε είναι η ιχνηθέτηση (μαρκάρισμα) των κόμβων που έχουν ήδη εξυπηρετηθεί. Για μεγαλύτερη απόδοση της υλοποίησης ενσωματώνουμε την πληροφορία αυτή στον πίνακα Cost_Temp, που είναι το αντίγραφο του αρχικού Cost. Συγκεκριμένα, όταν ένας κόμβος *j* έχει εξυπηρετηθεί ήδη, οι αποστάσεις προς τον κόμβο αυτό (με άλλα λόγια η στήλη

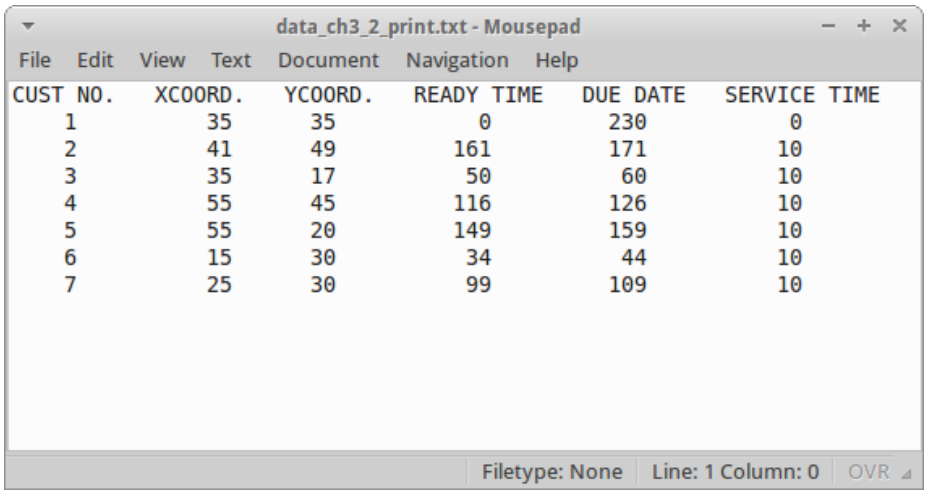
Cost(:,j)) απειρίζεται. Συνεπώς, από κάθε άλλο κόμβο *i* η απόσταση προς τον *j* "φαίνεται" ως άπειρη και δεν λαμβάνεται υπ'όψιν στην επίλυση του προβλήματος. Επίσης, αφού όλα τα φορτηγά τοποθετούνται στην αποθήκη αρχικά, τα στοιχεία της πρώτης στήλης είναι όλα Cost(:,1)=Inf, ώστε να ληφθούν υπ'όψιν οι λοιποί πελάτες εκτός της αποθήκης ως επόμενο δυνατό βήμα. Με την ίδια λογική τα στοιχεία της διαγωνίου θέτονται και αυτά ίσα με Inf, καθώς ένα όχημα που ήδη βρίσκεται στον κόμβο *i* δε θα πρέπει να έχει ως δυνατή επιλογή την "μετακίνηση" στον εαυτό του, δηλαδή ξάνα στον κόμβο *i*. Συνεπώς Cost(i,i) = Inf, για κάθε $i \epsilon$ [1, Cust_Num].

Εξίσου σημαντική είναι η διατήρηση του εκάστοτε αποτελέσματος από την επίλυση του προβλήματος. Πιο συγκεκριμένα, θα πρέπει να έχουμε τη διαδρομή που ακολουθεί κάθε όχημα. Για το σκοπό αυτό δημιουργούμε ένα δισδιάστατο πίνακα, όπου κάθε γραμμή αντιστοιχεί σε κάθε φορτηγό (άρα αριθμός γραμμών = Veh_Num). Ως σταθερό αριθμό στηλών θέτουμε τον αριθμό των πελατών, για το corner case όπου ένα φορτηγό εξυπηρετεί όλους τους πελάτες. Ο πίνακας αρχικοποιείται με μηδενικά στοιχεία, που δηλώνουν μη πραγματοποιημένη κίνηση. Έπειτα, όλα τα φορτηγά τοποθετούνται στην αποθήκη αρχικά, με άλλα λογια τα στοιχεία της πρώτης στήλης είναι όλα route(:,1)=1. Όταν το 1ο φορτηγό (1η γραμμή) κινείται στον επόμενο κόμβο X, το διπλανό σημείο της 1ης γραμμής ορίζεται ίσο με route(1,2)=X, το επόμενο route(1,3)=Y, κ.ο.κ. Με τον τρόπο αυτό οι στήλες με τα μη μηδενικά στοιχεία της γραμμής i αντιστοιχούν στην αλληλουχία των κόμβων από τους οποίους περνά το όχημα i. Προφανώς ο αριθμός βημάτων δεν είναι απαραίτητα ίδιος για όλα το φορτηγα αλλά σε όλες τις γραμμές το τελευταίο μη μηδενικό στοιχείο είναι ίσο με 1, καθώς όλα επιστρέφουν στην αποθήκη. Ανά σειρά i, μπορούμε να "απομονώσουμε" τη διαδρομή του αντίστοιχου οχήματος με μια απλή απομόνωση των μη μηδενικών στοιχείων (άμμεσα με εκτέλεση της εντολής nnz για έρευση των non-zero elements). Με τον τρόπο αυτό εξασφαλίζουμε μια ευέλικτη δομή για τη διατήρηση του αποτελέσματος, εξαλείφοντας την ανάγκη για δυναμική δέσμευση μνήμης σε πραγματικό χρόνο εκτέλεσης. Έτσι μειώνουμε δραστικά την πολυπλοκότητα του κώδικα.

Ορίζουμε δομές για να παρακολουθούνται τα βήματα κάθε οχήματος και για να ελέγχονται οι περιορισμοί ανά βήμα. Πιο συγκεκριμένα, ορίζουμε τα διανύσματα με διαστάσεις ίσες με τον αριθμό οχημάτων Veh_Num:

- Traveling_Time: Ο συνολικός χρόνος που ανά πάσα στιγμή έχει διανύσει κάθε φορτηγό, ώστε να ελέγχεται η δυνατότητα εξυπηρέτησης νέου πελάτη χωρίς τον παραβιασμό του χρόνου κλεισίματος της αποθήκης
- Arrival_Time: Με βάση και το τρέχοντα χρόνο ανά όχημα, η χρονική στιγμή που κάποιο όχημα θα έχει φτάσει στον επόμενο πελάτη. Με τον τρόπο αυτό ελέγχουμε αργότερα αν μπορούμε να ικανοποιήσουμε τον περιορισμό του DUE TIME του πελάτη
- cap_per_veh: Η χωρητικότητα κάθε οχήματος ανά πάσα στιγμή, για τον έλεγχο του περιορισμού της μέγιστης χωρητικότητας

Για παράδειγμα ας θεωρήσουμε μία απλή περίπτωση με 6 πελάτες προς εξυπηρέτηση και 2 διαθέσιμα οχήματα. Τα στοιχεία των πελατών και της αποθήκης παρουσιάζονται στο Σχήμα 3.2.3 παρακάτω. Παρατηρούμε στην αρίθμηση των κόμβων περιλαμβάνεται και η αποθήκη, συνεπώς Cust_Num = 7.



Σχήμα 3.2.3: *Δεδομένα απλού παραδείγματος με 6 πελάτες για την αρχική επίλυση του προβλήματος*

Με την εκτέλεση της συνάρτησης "create_cost_tables" κατασκευάζουμε τον ακόλουθο πίνακα:

Cost Temp

Inf	15.2315	18.0000	22.3607	25.0000	20.6155	11.1803
Inf	Inf	32.5576	14.5602	32.2025	32.2025	24.8395
Inf	32.5576	Inf	34.4093	20.2237	23.8537	16.4012
Inf	14.5602	34.4093	Inf	25.0000	42.7200	33.5410
Inf	32.2025	20.2237	25.0000	Inf	41.2311	31.6228
Inf	32.2025	23.8537	42.7200	41.2311	Inf	10.0000
Inf	24.8395	16.4012	33.5410	31.6228	10.0000	Inf

Πίνακας 3.2.3.1: *Αποτελέσματα χρήσης συνάρτησης "create_cost_tables"*

Όπως είναι αναμενόμενο, ο πίνακας έχει διαστάσεις Cust_Num x Cust_Num. Παρατηρούμε ότι η πρώτη στήλη που αντιστοιχεί σε μετακινήσεις προς την αποθήκη είναι απειρισμένη. Με εξαίρεση λοιπόν την πρώτη στήλη, ο υπο-πίνακας Cost_Temp[2:Cust_Num,2:Cust_Num] είναι συμμετρικός. Τέλος και τα στοιχεία της διαγωνίου περιέχουν το Inf. Κατ' επέκταση ο πίνακας route αρχικοποιείται ως εξής:

route							
1	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0

Πίνακας 3.2.3.2: *Αποτελέσματα πίνακα route*

Το ενδεικτικό αυτό παράδειγμα και οι προαναφερθείσες δομές θα χρησιμοποιηθούν στη συνέχεια για την επεξήγηση των υλοποιημένων υπορουτινών στα επιμέρους σημεία του αλγορίθμου.

3.3 Εύρεση Αρχικής Εφικτής Λύσης με τον Αλγόριθμο Απληστίας

Στην ενότητα αυτή θα παρουσιάσουμε την υλοποίηση για εύρεση αρχικής εφικτής λύσης με βάση έναν απλό αλγόριθμο απληστίας. Η συνολική διαδικασία περιγράφεται με τον παρακάτω ψευδοκώδικα στον Αλγόριθμο 3.1. Ο βασικός επαναληπτικός βρόχος εκτελείται μέχρι να εξυπηρετηθούν όλοι οι πελάτες. Για κάθε όχημα επιλέγεται η αλληλουχία πελατών προς εξυπηρέτηση με εκτέλεση του αλγόριθμου απληστίας σε κάθε βήμα του οχήματος. Ο αλγόριθμός ελέγχει σε κάθε βήμα αν το όχημα μπορεί να εξυπηρετήσει νέο πελάτη χωρίς να παραβιάσει τους χρονικούς περιορισμούς. Αν οι περιορισμοί παραβιάζονται, η ρουτίνα επιστρέφει 1, υποδεικνύοντας την επιστροφή του οχήματος στην αποθήκη. Στην περίπτωση αυτή, επιλέγεται επόμενο όχημα προς δρομολόγηση των εναπομεινάντων πελατών, και έπειτα τρίτο, κ.ο.κ. Με τον τρόπο αυτό η εκτέλεση συνεχίζεται μέχρι να εξυπηρετηθούν όλοι, ενώ διακόπτεται σε περίπτωση που παραβιαστεί ο περιορισμός μέγιστου αριθμού οχημάτων.

Αλγόριθμος 3.1 : Εύρεση Αρχικής Λύσης

visited ← Αρχικοποίησε αριθμό εξυπηρετημένων πελατών ίσο με 1

current_vehicle ← Αρχικοποίησε αριθμό χρησιμοποιημένων οχηματων ίσο με 1

Μέχρι να εξυπηρετηθούν όλοι οι πελάτες

current_pos_of_truck_i_step_Ki ← Θέση τρέχοντος οχήματος i στο βήμα K(i) από πίνακα route

Distances_from_Current_pnt ← Διάνυσμα αποστάσεων από τρέχουσα θέση προς άλλους κόμβους

next_point ← Εκτέλεσε ευρετικό αλγόριθμο για εύρεση επόμενου σημείου του τρέχοντος οχήματος

Προσέθεσε το next_point στη διαδρομή του τρέχοντος οχήματος

Ανανέωσε τη χωρητικότητα και τους χρόνους μετακίνησης/ άφιξης του οχήματος

Αν next_point ΔΕΝ είναι 1 (άρα πας σε νέο πελάτη και όχι πίσω στην αποθήκη)

visited + 1 ← Αύξησε τον αριθμό εξυπηρετημένων πελατών κατά 1

Αλλιώς

current_vehicle + 1 ← Πάρε νέο όχημα και αύξησε τα χρησιμοποιημένα οχήματα κατά 1

Τέλος Αν

Αν χρησιμοποιημένα οχήματα είναι περισσότερα από μέγιστο αριθμό οχημάτων

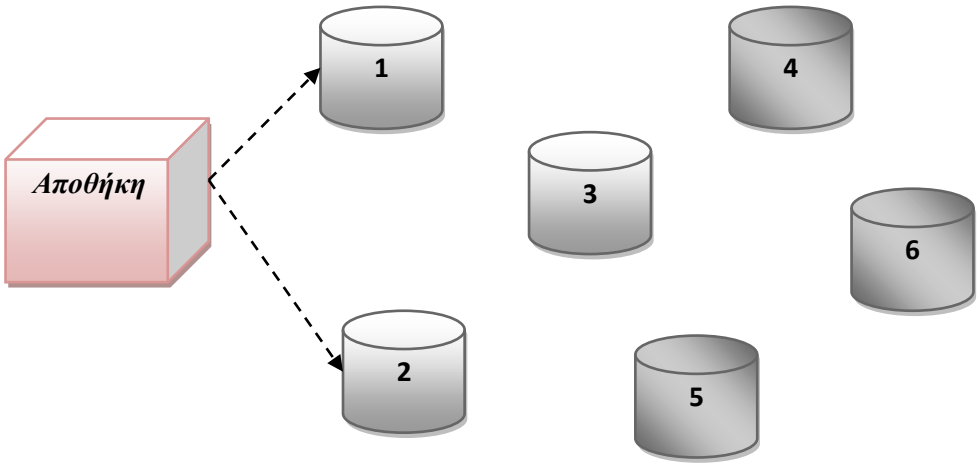
Επέστρεψε μήνυμα λάθους

Διέκοψε την επίλυση

Τέλος Αν

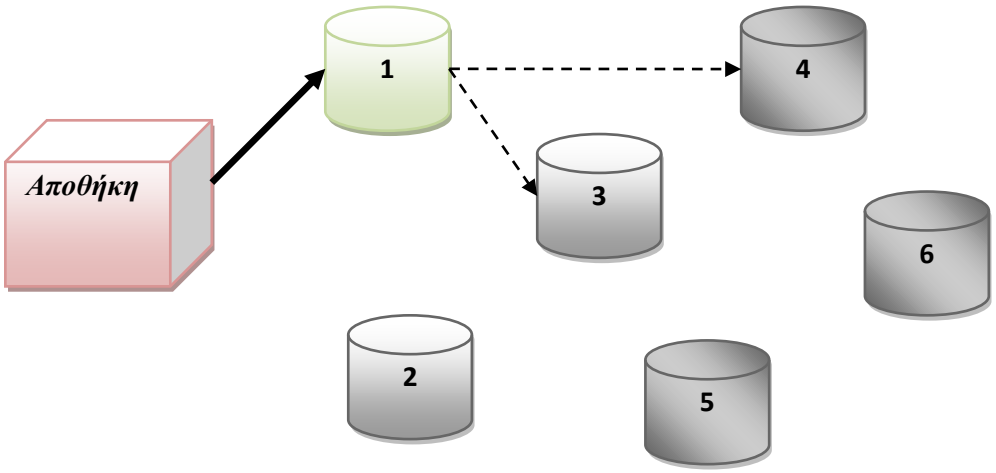
Τέλος Μέχρι

Στο επόμενο σχήμα παρουσιάζεται ένα απλό παράδειγμα για την εύρεση της αρχικής μας λύσης, όπου ξεκινώντας από την αποθήκη υπολογίζουμε τον πλησιέστερο πελάτη σε αυτήν. Έτσι αρχική μας επιλογή είναι ο πελάτης 1. Στη συνέχεια θα πρέπει να ελέγξουμε αν ικανοποιούνται οι περιορισμοί και αν αυτό συμβαίνει, τότε ο πελάτης επιλέγεται.



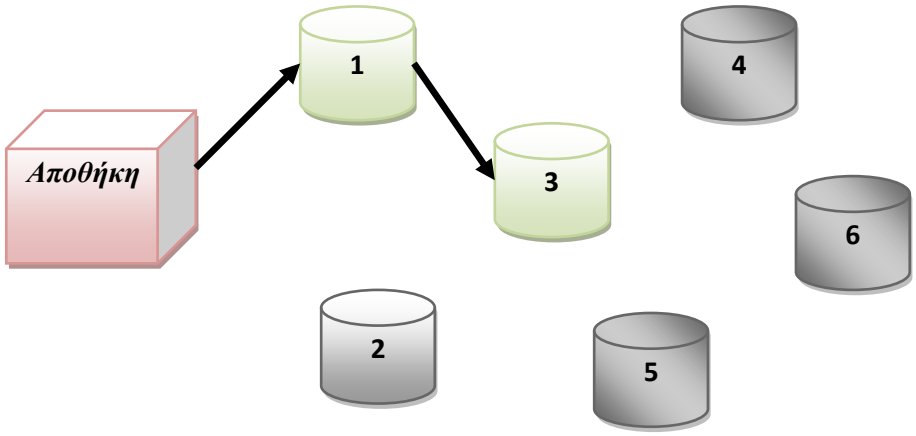
Σχήμα 3.3.1: *Εύρεση Αρχικής Εφικτής Λύσης με τον Αλγόριθμο Απληστίας πρώτο στάδιο*

Στη συνέχεια από τον πελάτη 1, θα πρέπει να επιλέξουμε τον επόμενο και κατά επέκταση πλησιέστερο σε αυτόν. Έτσι θα ελεγχθούν οι πελάτες 3 και 4.



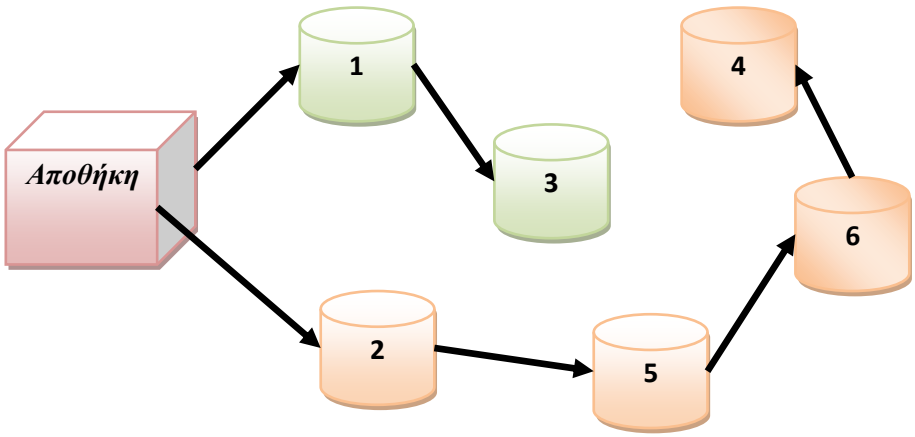
Σχήμα 3.3.2: *Εύρεση Αρχικής Εφικτής Λύσης με τον Αλγόριθμο Απληστίας πρώτο στάδιο*

Πλησιέστερος είναι ο πελάτης 3 και θα επιλεχθεί μόνο αν ικανουποιούνται οι περιορισμοί του προβλήματος. Έτσι η διαδρομή μας στο επόμενο βήμα θα έχει την εξής μορφή:



Σχήμα 3.3.3: *Εύρεση Αρχικής Εφικτής Λύσης με τον Αλγόριθμο Απληστίας δεύτερο στάδιο*

Αν στον επόμενο κοντινότερο του πελάτη 3, δεν ικανοποιείται κάποιος περιορισμός, η διαδικασία σταματάει και ένα νέο όχημα ξεκινά από την αποθήκη. Η διαδικασία επαναλαμβάνεται μέχρι να εξυπηρετηθούν όλοι οι πελάτες. Τελικά θα προκύψουν οι διαδρομές Αποθήκη - 1 - 3 - Αποθήκη και Αποθήκη - 2 - 5 - 6 - 4 - Αποθήκη. Σχηματικά δηλαδή θα έχουμε:



Σχήμα 3.3.4: *Εύρεση Αρχικής Εφικτής Λύσης με τον Αλγόριθμο Απληστίας τελικό στάδιο*

Το αρχικό τμήμα του παραπάνω αλγορίθμου αντιστοιχεί σε απλή αρχικοποίηση μεταβλητών. Ας θεωρήσουμε ξανά το απλό παράδειγμα που αναφέρθηκε στην προηγούμενη ενότητα. Από τον πίνακα:

route							
1	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0

Πίνακας 3.3.1: *Αποτελέσματα πίνακα route*

έχουμε ότι το πρώτο φορτηγό "current_vehicle=1" βρίσκεται κατά το βήμα K=1 στο σημείο route(current_vehicle,K(current_vehicle))=route(1,K(1))=route(1,1)=1, δηλαδή στην αποθήκη. Άρα έχουμε ότι current_pos_of_truck_i_step_Ki =1. Γνωρίζουμε ότι η αποστάση από το σημείο αυτό προς ένα άλλο σημείο j δίνεται από τον πίνακα Cost_Temp, και συγκεκριμένα αντιστοιχεί στην τιμή Cost_Temp(current_pos_of_truck_i_step_Ki,j). Άρα, οι αποστάσεις από το τρέχον σημείο αντιστοιχούν στην current_pos_of_truck_i_step_Ki γραμμή του πίνακα Cost_Temp. Δηλαδή:

Distances_from_Current_pnt = Cost_Temp(current_pos_of_truck_i_step_Ki, :) =						
Inf	15.2315	18.0000	22.3607	25.0000	20.6155	11.1803

Πίνακας 3.3.2: *Αποτελέσματα πίνακα Cost_Temp*

Στο σημείο λοιπόν αυτό, πρώτο σημαντικό βήμα είναι η εκτέλεση ενός ευρετικού αλγορίθμου για την εύρεση του επόμενου σημείου "next_point" που πρόκειται να μεταβεί το τρέχον όχημα. Άρα πρώτο σημαντικό βήμα της υλοποίησής μας είναι η κατασκευή της αντίστοιχης υπορουτίνας. Με βάση την περιγραφή της μεθόδου GRASP από προηγούμενο κεφάλαιο, γνωρίζουμε ότι μια τυχαιοποιημένη συνάρτηση απληστίας χρησιμοποιείται για να κατασκευαστεί μια αρχική λύση. Πιο συγκεκριμένα, το επόμενο βήμα επιλέγεται τυχαία από μία λίστα υποψηφίων, στην οποία κάθε στοιχείο κατατάσσεται βάσει μιας συνάρτησης απληστίας. Πρωτού λοιπόν κατασκευάσουμε τη διαδικασία αυτή, θα πρέπει να ξεκινήσουμε με την υλοποίηση της συνάρτησης απληστίας. Στα πλαίσια της εργασίας μας επιλέγεται η μέθοδος του κοντινότερου γείτονα, όπου οι πιθανοί προορισμοί κατατάσσονται με βάση τη γεωμετρική του απόσταση από το τρέχον σημείο. Βέβαια, η μέθοδος του πλησιέστερου γείτονα, στην απλή της μορφή δεν επαρκεί πάντα για την εύρεση εφικτής λύσης για την περίπτωση του προβλήματος με χρονικά παράθυρα. Θα πρέπει λοιπόν να επεκταθεί ανάλογα, ώστε να κατασκευαστεί τελικά η ζητούμενη τυχαιοποιημένη διαδικασία. Στις επόμενες υποενότητες παρουσιάζεται αναλυτικά η διαδικασία αυτή. Αρχικά καταδεικνύουμε ότι στην απλή της μορφή η μέθοδος του

απλού γείτονα μπορεί να μην επιστρέψει πάντα εφικτή λύση. Στη συνέχεια προτείνουμε σχετική τροποποίηση και τέλος παρουσιάζουμε συνολικά τη διαδικασία κατασκευής αρχικής λύσης.

3.3.1 Υπορουτίνα Αλγόριθμου Απληστίας Πλησιέστερου Γείτονα (NearestNeighbor)

Στην περίπτωση του απλού γείτονα επιλέγεται το πρώτο στοιχείο από μια ταξινομημένη λίστα. Το διάνυσμα ταξινόμησης προκύπτει με βάση τις αποστάσεις από το τρέχον σημείο. Θα πρέπει λοιπόν αρχικά να ταξινομηθεί το διάνυσμα αποστάσεων Distances_from_Current_pnt με τη χρήση της απλής ρουτίνας της MATLAB "sort(Distances_from_Current_pnt)". Ως αποτέλεσμα έχουμε το ταξινομημένο διάνυσμα :

candidates_sorted						
11.1803	15.2315	18.0000	20.6155	22.3607	25.0000	Inf

Πίνακας 3.3.1.1: *Αποτελέσματα ταξινομημένου διανύσματος candidates_sorted*

όπου πράγματι τα στοιχεία με μικρότερες τιμές βρίσκονται στις πρώτες θέσεις και τα μεγαλύτερα στις τελευταίες. Με τον τρόπο αυτό γνωρίζουμε τον καλύτερο υποψήφιο προς επίσκεψη, δεν γνωρίζουμε όμως αν είναι όντως δυνατή η επίσκεψη αυτή ώστε να ικανοποιούνται ταυτόχρονα όλοι οι περιορισμοί. Θα πρέπει λοιπόν να εξετάσουμε τον πρώτο υποψήφιο και να ελέγξουμε τις ανάλογες συνθήκες με βάση τους περιορισμούς. Αν ικανοποιούνται, ο υποψήφιος επιλέγεται ως το επόμενο σημείο route(current_vehicle,K(current_vehicle)+1) της διαδρομής του τρέχοντος current_vehicle. Διαφορετικά, θα πρέπει να ελέγξουμε το δεύτερο κατά σειρά υποψήφιο, το τρίτο κ.ο.κ..

Στο παράδειγμά μας ως πρώτος υποψήφιος προς εξέταση έχει τοποθετηθεί ο κόμβος 7, με απόσταση 11.1803. Παρατηρούμε ότι στη συνέχεια της εκτέλεσης θα χρειαστεί να κάνουμε πρόσβαση σε δομές που αντιστοιχούν στον πελάτη αυτό (για παράδειγμα Due_Time(7)). Ταυτόχρονα λοιπόν με την ταξινόμηση του διανύσματος Distances_from_Current_pnt, θα πρέπει να ταξινομήσουμε και τις λοιπές δομές που κατασκευάστηκαν προηγουμένως, ώστε να γνωρίζουμε σε αύξουσα σειρά τα στοιχεία των πιο οφέλιμων πελατών. Για το σκοπό αυτό, θα είναι σημαντικό να αποθηκεύσουμε και τους δείκτες εκτός από το αποτέλεσμά της. Γνωρίζουμε ότι το δεύτερο αποτέλεσμα της "sort" είναι το διάνυσμα index_of_points, που περιέχει τις αρχικές θέσεις των ταξινομημένων στοιχείων:

index_of_points						
7	2	3	6	4	5	1

Πίνακας 3.3.1.2: *Αποτελέσματα διανύσματος index_of_points*

Με βάση τη σύνταξη της MATLAB, η αναφορά σε στοιχείο ενός διανύσματος A με σύνταξη Due_Time(index_of_points(1)) = Due_Time(7) είναι ισοδύναμο με αναφορά σε διάνυσμα B με σύνταξη Due_Time_sorted(1), αν στο διάνυσμα B έχει επιβληθεί η ταξινόμηση index_of_points:

$$Due\_Time\_sorted = Due\_Time(index\_of\_points); \quad (1)$$

Πράγματι λοιπόν, στη δομή Due_Time_sorted έχουμε με αύξουσα σειρά τους βραδύτερους χρόνους των πελατών προς εξέταση, καθώς από τη σύνταξη της MATLAB έχουμε:

$$Due\_Time\_sorted(1) = Due\_Time(index\_of\_points(1)) = Due\_Time \quad (7)$$

Στο πρώτο λοιπόν κομμάτι της υπορουτίνας NearestNeighbor εφαρμόζουμε την συνάρτηση sort και με βάση το αποτέλεσμά της ταξινομούμε τις βασικές δομές δεδομένων που θα χρειαστούμε στη συνέχεια για τον έλεγχο των περιορισμών. Είπαμε ότι η διαδικασία αυτή θα επαναληφθεί για κάθε υποψήφιο candidate. Οι περιορισμοί που πρέπει να ελεγχθούν είναι:

- Ο πελάτης δεν πρέπει να έχει εξυπηρετηθεί, δηλαδή:
  $$Distances\_sorted(candidate) \sim= Inf \quad (2)$$

- Αν ο χρόνος που θα φτάσουμε στον πελάτη είναι μικρότερος από το βραδύτερο χρόνο που μπορεί να δεχτεί επίσκεψη ο πελάτης, δηλαδή:

  $$Arrival\_Time\_New <= Due\_Time\_sorted(candidate) \quad (3)$$

- Η ποσότητα προϊόντος θα πρέπει να ικανοποιεί τη ζήτηση του πελάτη, δηλαδή:
  $$Cap\_New <= Veh\_Cap \quad (4)$$

- Το φορτηγό στη συνέχεια θα πρέπει να μπορεί να επιστρέψει στην αποθήκη χωρίς να υπερβεί το βραδύτερο χρόνο της αποθήκης, δηλαδή:
  $$Traveling\_Time\_New + Cost(index\_of\_points(candidate),1) <= Time\_Max \quad (5)$$

Οι περιορισμοί που ελέγχουμε προκύπτουν από τις παρακάτω πράξεις που πραγματοποιούνται μέσα στην αλγοριθμική επαναληπτική διαδικασία:

- Cap_New = cap_per_veh(current_vehicle) - Cust_Demand_sorted(candidate) (6)
- Arrival_Time_New = Arrival_Time(current_vehicle) + Distances_sorted(candidate) (7)
- Traveling_Time_New = Traveling_Time(current_vehicle + Distances_sorted(candidate) + Time_Service + waiting_time (8)

Παρατηρούμε ότι το next_point (η λύση που επιστρέφει δηλαδή η συνάρτηση) αρχικοποιείται στο 1. Αυτό σημαίνει ότι εάν κανένας υποψήφιος δεν ικανοποιεί τους περιορισμούς, η συνάρτηση επιστρέφει next_point == 1, δηλαδή το όχημα επιστρέφει πίσω στην αποθήκη και χρησιμοποιούμε νέο όχημα τη χρονική στιγμή *t=0*, με νέο διαθέσιμο φορτίο. Τα προαναφερθέντα βήματα συγκεντρώνονται και περιγράφονται στον ακόλουθο Αλγόριθμο 3.2.

Αλγόριθμος 3.2 : Υπορουτίνα μεθόδου κοντινότερου γείτονα NearestNeighbor

candidates_list ← Αρχικοποίησε λίστα υποψηφίων με βάση αποστάσεις Distances_from_Current_pnt

candidates_sorted ← Ταξινόμησε λίστα candidates_list και κράτα σειρά ταξινόμησης index_of_points

Ταξινόμησε τις λοιπές δομές (όπως Cust_Demand, Due_Time κλπ) με αύξουσα σειρά index_of_points

next_point ← Αρχικοποίησε ως επόμενο σημείο την αποθήκη

Για όλους τους υποψηφίους της ταξινομημένης λίστας candidates_sorted

Αν Ο πελάτης δεν πρέπει να έχει εξυπηρετηθεί (απόσταση προς αυτόν δεν είναι Inf)

Υπολόγισε τρέχουσες τιμές περιορισμών (Εξισώσεις 3-5)

Αν οι περιορισμοί ικανοποιούνται

Εξυπηρέτησε τον πελάτη

Ανανέωσε κατάλληλα μεταβλητές (όπως χωρητικότητα οχήματος κλπ)

next_point ← Επέστρεψε ως επόμενο σημείο τον συγκεκριμένο πελάτη

Διέκοψε την διαδικασία (έξοδος από βρόχο, υπορουτίνα)

Τέλος Αν

Τέλος Αν

Τέλος Για

Αν δε βρέθηκε λύση (next_point==1)

Ανανέωσε κατάλληλα μεταβλητές (όπως χωρητικότητα οχήματος κλπ)

next_point ← Επέστρεψε το όχημα στην αποθήκη

Τέλος Αν

Αν λοιπόν εκτελέσουμε τον Αλγόριθμο 3.1 με χρήση του Αλγορίθμου 3.2 ως ευρηματική μέθοδο, παρατηρούμε ότι η μέθοδος αδυνατεί να βρει εφικτή λύση. Συγκεκριμένα η εκτέλεση της NearestNeighbor επιστρέφει και τα 2 διαθέσιμα φορτηγά στην αποθήκη μετά την εξυπηρέτηση μόλις 2 πελατών ανά όχημα. Πιο συγκεκριμένα:

route							
1	7	2	1	0	0	0	0
1	3	5	1	0	0	0	0

Πίνακας 3.3.1.3: *Αποτελέσματα πίνακα route*

Με βάση τον Αλγόριθμο 3.1, αφού έχουν χρησιμοποιηθεί και τα 2 φορτηγά ενώ οι εξυπηρετημένοι πελάτες είναι visited ~= Cust_Num, η υλοποίησή μας επιστρέφει μήνυμα λάθους. Για την εύρεση των μη εξυπηρετημένων πελατών, μπορούμε να αξιοποιήσουμε την ιδιότητα της δομής Cost και Cost_Temp ότι πελάτες στους οποίους δεν έχει πραγματοποιηθεί επίσκεψη θα έχουν στήλη αποστάσεων διάφορη του απείρου. Πράγματι, βλέπουμε ότι δεν έχουν εξυπηρετηθεί οι πελάτες 4 και 6.

Cost_Temp						
Inf	Inf	Inf	22.3607	Inf	20.6155	Inf
Inf	Inf	Inf	14.5602	Inf	32.2025	Inf
Inf	Inf	Inf	34.4093	Inf	23.8537	Inf
Inf	Inf	Inf	Inf	Inf	42.7200	Inf
Inf	Inf	Inf	25.0000	Inf	41.2311	Inf
Inf	Inf	Inf	42.7200	Inf	Inf	Inf
Inf	Inf	Inf	33.5410	Inf	10.0000	Inf

Πίνακας 3.3.1.4: *Αποτελέσματα πίνακα Cost_Temp*

Οι δείκτες των πελατών αυτών μπορούν εύκολα να βρεθούν αναζητώντας τις θέσεις σημείων μη άπειρων με χρήση της συνάρτησης MATLAB. Εκτελούμε λοιπόν no_served_clients = find(Cost_Temp(1,:)<10000). Το μήνυμα λάθους κατάλληλα επιστρέψει:

Have not served 2 no_served_clients = 4 6

Ο λόγος που συμβαίνει αυτό είναι ότι η απλή υλοποίηση του πλησιέστερου γείτονα δεν περιέχει πληροφορία για τα χρονικά παράθυρα. Και για τα 2 οχήματα επιλέγονται οι 2 κοντινότεροι πελάτες και εξυπηρετούνται, ενώ οι εναπομείναντες έχουν ήδη "χάσει" τις προθεσμίες τους για βραδύτερους χρόνους. Για το σκοπό αυτό θα επεκτείνουμε κατάλληλα τη μέθοδο. Η υπορουτίνα που προκύπτει είναι η NearestNeighbor_TimeWindows και περιγράφεται στην επόμενη υποενότητα.

3.3.2 Υπορουτίνα Αλγορίθμου Απληστίας Πλησιέστερου Γείτονα με Χρονικά Παράθυρα (NearestNeighbor_TimeWindows)

Ο τρόπος για αντιμετώπιση του προβλήματος χρονικών παραθύρων βασίζεται στην απλή παρατήρηση ότι πελάτες που επείγει να εξυπηρετηθούν θα πρέπει να τοποθετηθούν πρώτοι στην ταξινόμηση. Με άλλα λόγια πελάτες με μικρότερους νωρίτερους χρόνους READY_TIME θα πρέπει να τοποθετηθούν στην αρχή της ταξινομημένης λίστας υποψηφίων. Αυτό μπορεί εύκολα να υλοποιηθεί με την εισαγωγή βαρών (συντελεστών) τον συνυπολογισμό των χρόνων στις τιμές των στοιχείων προς ταξινόμηση, δηλαδή:

*candidates_list = W1 * Distances_from_Current_pnt + W2 * (Ready_Time');* (9)

όπου W1 και W2 τα βάρη για κάθε ένα από τους όρους. Τα βάρη αυτά έχουν επιλεχθεί κατάλληλα ύστερα από επαναλήψεις και δοκιμές. Μπορούμε λοιπόν εύκολα να παρατηρήσουμε ότι ένας πελάτης Α που επείγει να εξυπηρετηθεί θα έχει μικρό Ready_Time. Αυτό σημαίνει ότι σε σχέση με κάποιον άλλο πελάτη Β που ίσως βρίσκεται κοντινότερα από το τρέχον σημείο, η συνολική τιμή του candidates_list θα είναι μικρότερη για τον Α. Κατά συνέπεια, ο περιορισμός λόγω παραθύρου θα ληφθεί κατάλληλα υπόψη και ο Α θα επιλεγεί ως ο επόμενος πελάτης. Παρατηρούμε επίσης ότι θα μπορούσε ισοδύναμα να χρησιμοποιηθούν οι βραδύτεροι χρόνοι DUE_TIME καθώς ισοδυναμούν στους νωρίτερους χρόνους με τη προσθήκη της σταθεράς SERVICE_TIME.

Η βελτίωση με την εισαγωγή της τροποποίησης αυτής είναι ορατή με την επανεκτέλεση του ενδεικτικού παραδείγματος των 7 κόμβων που παρουσιάστηκε προηγουμένως. Όπως είπαμε, κατά το πρώτο βήμα του πρώτου φορτηγού, οι αποστάσεις από την αποθήκη είναι:

Distances_from_Current_pnt						
Inf	15.2315	18.0000	22.3607	25.0000	20.6155	11.1803

Πίνακας 3.3.2.1: *Αποτελέσματα πίνακα Dinstances_from_Current_pnt*

Με βάση το παραπάνω διάνυσμα ο Αλγόριθμος 3.2 θα επιλέξει τον κόμβο 7 ως επόμενο σημείο, καθώς έχει τη μικρότερη απόσταση ίση με 11.1803. Με βάση όμως την Εξίσωση (9), η λίστα υποψηφίων γίνεται:

candidates_list						
Inf	146.4232	46.8000	106.6361	136.6000	32.6616	90.2180

Πίνακας 3.3.2.2: *Αποτελέσματα πίνακα candidates_list*

όπου το 6ο στοιχείο έχει μικρότερη τιμή, γεγονός αναμενόμενο καθώς έχει τιμή νωρίτερου χρόνου Due_Time(6) = 44, πολύ μικρότερη συγκριτικά σε σχέση με τους άλλους πελάτες. Άρα αυτό το σημείο θα είναι και το πρώτο ύστερα από ταξινόμηση:

candidates_sorted						
32.6616	46.8000	90.2180	106.6361	136.6000	146.4232	Inf

Πίνακας 3.3.2.3: *Αποτελέσματα πίνακα candidates_sorted*

Άρα επιλέγεται αυτό το σημείο προς έλεγχο των περιορισμών. Αν δεν τους ικανοποιεί ελέγχεται το επόμενο, κ.ο.κ.. Κατά συνέπεια λοιπόν το υπόλοιπο κομμάτι της υπορουτίνας, με εξαίρεση το αρχικό κομμάτι εκτέλεσης της Εξίσωσης (8), δε διαφέρει σε σχέση με την απλή μέθοδο. Συνεπώς, έχουμε τον παρακάτω Αλγόριθμο 3.3.

Αλγόριθμος 3.2: Κοντινότερος γείτονας με χρονικά παράθυρα NearestNeighbor_TimeWindows

candidates_list ← Αρχικοποίησε λίστα υποψηφίων με βάση Εξίσωση (9)

candidates_sorted ← Ταξινόμησε λίστα candidates_list και κράτα σειρά ταξινόμησης index_of_points

Ταξινόμησε τις λοιπές δομές (όπως Cust_Demand, Due_Time κλπ) με αύξουσα σειρά index_of_points

next_point ← Αρχικοποίησε ως επόμενο σημείο την αποθήκη

Για όλους τους υποψηφίους της ταξινομημένης λίστας candidates_sorted

Αν Ο πελάτης δεν πρέπει να έχει εξυπηρετηθεί (απόσταση προς αυτόν δεν είναι Inf)

Υπολόγισε τρέχουσες τιμές περιορισμών (Εξισώσεις 2-4)

Αν οι περιορισμοί ικανοποιούνται

Εξυπηρέτησε τον πελάτη

Ανανέωσε κατάλληλα μεταβλητές (όπως χωρητικότητα οχήματος κλπ)

next_point ← Επέστρεψε ως επόμενο σημείο τον συγκεκριμένο πελάτη

Διέκοψε την διαδικασία (έξοδος από βρόχο, υπορουτίνα)

Τέλος Αν

Τέλος Αν

Τέλος Για

Αν δε βρέθηκε λύση (next_point==1)

Ανανέωσε κατάλληλα μεταβλητές (όπως χωρητικότητα οχήματος κλπ)

next_point ← Επέστρεψε το όχημα στην αποθήκη

Τέλος Αν

Πράγματι, με εκτέλεση της εν λόγω υπορουτίνας έχουμε αποτέλεσμα:

route							
1	6	7	5	1	0	0	0
1	3	4	2	1	0	0	0

Πίνακας 3.3.2.4: *Αποτελέσματα πίνακα route*

άρα όλοι οι πελάτες εξυπηρετούνται κατάλληλα.

3.3.3 Υπορουτίνα Τυχαιοποιημένης Συνάρτησης Απληστίας Πλησιέστερου Γείτονα με Χρονικά Παράθυρα (Random_TimeWindows)

Με βάση την τροποποίηση που περιγράφηκε στην προηγούμενη υποενότητα, η μέθοδος μπορεί έγκυρα να χειριστεί χρονικά παράθυρα. Αφού λοιπόν η συνάρτηση απληστίας είναι ολοκληρωμένη στην απλή της μορφή, μπορούμε πολύ εύκολα να την επεκτείνουμε στην τυχαιοποιημένη περίπτωση. Στην περίπτωση αυτή οι υποψήφιοι προς έλεγχο των περιορισμών επιλέγονται τυχαία από τη λίστα υποψηφίων. Από τη θεωρία γνωρίζουμε ότι 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. Γνωρίζουμε ότι υπάρχουν δύο βασικοί τρόποι για την κατασκευή λίστας. Στην πρώτη περίπτωση η λίστα έχει πάντα σταθερό, προκαθορισμένο μήκος. Βασικό μειονέκτημα της μεθόδου αυτής είναι ότι μία μικρή

σχετικά λίστα που είναι υποσύνολο της αρχικής ταξινόμησης θα μπορούσε να περιορίσει την δυνατότητα της υπορουτίνας να επιστρέψει λύση. Για παράδειγμα, έστω ότι το πέμπτο κατά σειρά στοιχείο της ταξινομημένης λίστας υποψηφίων είναι αυτό που ικανοποιεί τις συνθήκες και θα επιλεγόταν από τον Αλγόριθμο 3.2 ως η λύση next_point. Αν έχουμε ορίσει το σταθερό μήκος ίσο με 4, και η τυχαία επιλογή γίνει μεταξύ των 4 πρώτων στοιχείων της λίστας, σημαίνει ότι η τυχαιοποιημένη περίπτωση θα αδυνατεί να επιστρέψει λύση. Με βάση το δεύτερο τρόπο, χρησιμοποιείται κατώφλι, όπου για να μπει κάποιο στοιχείο στη λίστα θα πρέπει η τιμή του να είναι ανάμεσα στο διάστημα$\mathbf{[}[\boldsymbol{c_{min}}, \boldsymbol{c_{min}} + \boldsymbol{a} \times (\boldsymbol{c_{max}} - \boldsymbol{c_{min}})]$. Στην δεύτερη περίπτωση βασικό μειονέκτημα είναι ότι η αποδοτικότητα της μεθόδου εξαρτάται σε τεράστιο βαθμό από την επιλογή του α. Για μεγάλη τιμή $\alpha \rightarrow 1$, το σύνολο περιέχει όλα τα σημεία, ενώ για $\alpha \rightarrow 0$ η λίστα περιορίζεται αρκετά και αντιμετωπίζει το ίδιο πρόβλημα του πρώτου τρόπου.

Για την αντιμετώπιση των παραπάνω προβλημάτων προτείνεται μία υβριδική λύση, που αποτελεί τη βασική συνεισφορά της παρούσας εργασίας. Η αρχική ταξινομημένη λίστα υποψηφίων τυχαιοποιείται ανα ομάδες (blocks) υποψηφίων. Με τον τρόπο αυτό ο αριθμός των υποψηφίων δε μειώνεται, συνεπώς αντιμετωπίζονται προβλήματα μη εύρεσης λύσης λόγω αυστηρών περιορισμών. Ταυτόχρονα, θέλουμε να αντιμετωπίσουμε την απόλυτα τυχαία επιλογή που είναι πρόβλημα της χρήσης κατωφλίου με $a \rightarrow 1$. Έτσι λοιπόν, επιλέγοντας τυχαία ένα από όλα τα στοιχεία, ίσως οδηγήσει σε επιλογή του μη βέλτιστου (για παράδειγμα τελευταίου της ταξινόμησης με μεγαλύτερη συνάρτηση κόστους). Για την αποφυγή αυτού του προβλήματος, προτείνεται τυχαιοποίηση των how_many στοιχείων, μετά των επόμενων how_many, κ.ο.κ..

Κατά συνέπεια, αντιμετωπίζονται τα 2 corner cases της πλήρους τυχαιοποιημένης και πλήρους άπληστης συνάρτησης με $a \rightarrow 1$ και $a \rightarrow 0$ αντίστοιχα. Με άλλα λόγια, έχουμε υβριδική λύση εν μέρει άπληστη (καθώς οι how_many επιλογές εξετάζονται πρώτες) αλλά ταυτόχρονα εξ ίσου τυχαιοποιημένη, καθώς τυχαία επιλέγεται ένα απο τα how_many στοιχεία. Επίσης η υπολοποίηση αυτή αυξάνει την ευελιξία του κώδικα, καθώς απαιτείται η τυχαιοποίηση μόνο των δεικτών ανά ομάδες των how_many στοιχείων. Μετά από την επιβολή της ταξινόμησης αυτής τις δομές δεδομένων όπως στις προηγούμενες υποενότητες, η συνέχεια της υλοποίησης είναι ακριβώς ίδια όπως στους Αλγορίθμους 3.2 και 3.3. Μπορούμε λοιπόν πολύ εύκολα να δημιουργήσουμε μία τυχαιοποιημένη σειρά δεικτών indexed ομαδοποιημένων σε how_many-άδες. Για το σκοπό αυτό χρησιμοποιούμε την συνάρτηση της MATLAB 'randperm(how_many)', που επιστρέφει τυχαία αρίθμηση από το 1 έως το how_many. Δηλαδή για εκτέλεση randperm(5) έχουμε μία τυχαία σειρά των αριθμών από 1 έως 5, δηλαδή:

ans				
5	3	1	4	2

Έστω ότι έχουμε λίστα με 10 υποψηφίους και θέλουμε να τους τυχαιοποιήσουμε σε 5-άδες, άρα 2 blocks. Το 1ο block (m=1) θα έχει δείκτες από 1 έως 5, δηλαδή το αποτέλεσμα της randperm(5). Το 2ο block (m=2) θα έχει τυχαία σειρά των αριθμών από 6 έως 10, άρα είναι ισοδύναμο στο αποτέλεσμα της εκτέλεσης:

$$(2 - 1)*5 + \text{randperm}(5) = 5 + \text{randperm}(5) =$$

$$5 + 2 \quad 5 + 5 \quad 5 + 3 \quad 5 + 4 \quad 5 + 1 = 7 \quad 10 \quad 8 \quad 9 \quad 6$$

Συνεπώς αλγοριθμικά αρκεί να βρούμε τον αριθμό των blocks με την εντολή:

```
% how many groupings of K-elements do you have??
groups = floor(Cust_Num./how_many);
```

και για κάθε how_many-άδα m να βρούμε τυχαία σειρά δεικτών:

```
randomize_indexes = (m-1)*how_many + randperm(how_many);
```
(10)

Με τον τρόπο αυτό κατασκευάζουμε τυχαιοποιημένη σειρά δεικτών, που "επιβάλλεται" στη λίστα που είναι ταξινομημένη ως προς τη συνάρτηση κόστους (στην περίπτωσή μας απλός κοντινότερος γείτονας). Χαρακτηριστικά λοιπόν, στο απλό παράδειγμα που ήδη έχουμε εξετάσει, επιλέγουμε μικρή τιμή how_many=2. Από την εκτέλεση του sort έχουμε αρχικά ταξινόμηση των δεικτών:

index_of_points_not_random						
6	3	7	4	5	2	1

Πίνακας 3.3.3.1: *Αποτελέσματα διανύσματος index_of_points_not_random*

Αντί λοιπόν να ταξινομήσουμε τις δομές δεδομένων (π.χ. Due_Time) με βάση το αποτέλεσμα της ταξινόμησης (τους δείκτες index_of_points_not_random με βάση

την ιδιότητα από Εξίσωση 1), τυχαιοποιούμε αυτούς τους δείκτες ανά 2-άδες. Για μια εκτέλεση της διαδικασίας, έχουμε:

index_of_points						
3	6	7	4	2	5	1

Πίνακας 3.3.3.2: *Αποτελέσματα διανύσματος index_of_points*

Βλέπουμε λοιπόν ότι από την πρώτη 2άδα του index_of_points_not_random με στοιχεία 6 και 3, τυχαία επιλέγεται το 3 πρώτα (άρα μετά το 6), από τη δεύτερη 2-άδα τυχαίνει και πρώτα επιλέγεται το ήδη πρώτο στοιχείο από τα δύο (τα 7 και 4), κ.ο.κ.. Μπορούμε λοιπόν την τυχαιοποιημένη ανά 2-άδες ταξινόμηση να την εφαρμόσουμε και στις δομές. Ενώ ο Αλγόριθμος 3.2 επέλεξε συνεπώς το στοιχείο 6, η τυχαιοποιημένη περίπτωση έδωσε το 3 ως αρχικό υποψήφιο. Αφού λοιπόν ελέγχεται κατά τα γνωστά ότι ικανοποιεί τις συνθήκες, επιλέγεται ως το επόμενο βήμα του 1ου φορτηγού. Η χαρακτηριστική διαφορά φαίνεται στην παρακάτω συνολική λύση, με τον Αλγόριθμο 3.3 να δίνεται στο τέλος:

route (Αλγόριθμος 3.2)								**route (Αλγόριθμος 3.3)**							
1	6	7	5	1	0	0	0	1	3	7	5	1	0	0	0
1	3	4	2	0	0	0	0	1	6	4	2	0	0	0	0

Πίνακας 3.3.3.3: *Αποτελέσματα route μέσω αλγορίθμων 3.2 και 3.3*

Αλγόριθμος 3.3 : Τυχαιοποιημένη συνάρτηση απληστίας Random_TimeWindows

candidates_list ← Αρχικοποίησε λίστα υποψηφίων με βάση Εξίσωση (9)

candidates_sorted ← Ταξινόμησε λίστα candidates_list

Κράτα σειρά ταξινόμησης index_of_points_not_random

index_of_points ← Τυχαιοποίησε δείκτες ανά how_many στοιχεία με βάση Εξίσωση 10

Ταξινόμησε τις λοιπές δομές (όπως Cust_Demand, Due_Time κλπ) με αύξουσα σειρά index_of_points

next_point ← Αρχικοποίησε ως επόμενο σημείο την αποθήκη

Για όλους τους υποψηφίους της ταξινομημένης λίστας candidates_sorted

Αν Ο πελάτης δεν πρέπει να έχει εξυπηρετηθεί (απόσταση προς αυτόν δεν είναι Inf)

Υπολόγισε τρέχουσες τιμές περιορισμών (Εξισώσεις 2-4)

Αν οι περιορισμοί ικανοποιούνται

Εξυπηρέτησε τον πελάτη

Ανανέωσε κατάλληλα μεταβλητές (όπως χωρητικότητα οχήματος κλπ)

next_point ← Επέστρεψε ως επόμενο σημείο τον συγκεκριμένο πελάτη

Διέκοψε την διαδικασία (έξοδος από βρόχο, υπορουτίνα)

Τέλος Αν

Τέλος Αν

Τέλος Για

Αν δε βρέθηκε λύση (next_point==1)

Ανανέωσε κατάλληλα μεταβλητές (όπως χωρητικότητα οχήματος κλπ)

next_point ← Επέστρεψε το όχημα στην αποθήκη

Τέλος Αν

3.4 Υλοποίηση Τοπικής Αναζήτησης

Η λύση που δημιουργείται στη φάση κατασκευής δεν εγγυάται ότι είναι τοπικό ελάχιστο, και για αυτό το λόγο μια φάση τοπικής αναζήτησης εφαρμόζεται για να παράξει μια λύση που να περιέχει τοπικό ελάχιστο. Για να εφαρμοστεί η φάση τοπικής αναζήτησης, καθορίζεται μια συνάρτηση που να κάνει αναζήτηση στη γειτονιά της αρχικής λύσης, όπως ανταλλαγές κόμβων μεταξύ διαφορετικών διαδρομών, είτε κόμβων της ίδιας διαδρομής. Στα πλαίσια της παρούσας εργασίας χρησιμοποιούμε και τις 3 μεθόδους για βελτίωση της αρχικής λύσης. Ανά περίπτωση, η υλοποίησή μας περιγράφεται στις επόμενες υποενότητες.

3.4.1 Πραγματοποίηση της Μεθόδου 2-opt

Σε αυτή την περίπτωση της μεθόδου 2-opt έχουμε δημιουργήσει μια αρχική λύση και πάνω σε αυτή θα εφαρμόσουμε τον αλγόριθμο τοπικής αναζήτησης. Οι τρόποι που δημιουργούνται οι λύσεις είναι είτε μέσω κάποιου αλγορίθμου απληστίας, είτε με τη χρήση κάποιας τυχαίας αρχικής λύσης. Στην περίπτωσή μας, η λύση έχει βρεθεί μέσω του αλγορίθμου απληστίας. Η μέθοδος αποτελείται γενικά από τη διαγραφή δύο ακμών και την επανασύνδεση δύο μονοπατιών με διαφορετικό τρόπο για να καθορίσουμε μια νέα διαδρομή. Έστω ότι έχουμε μια αρχική λύση με διαδρομή:

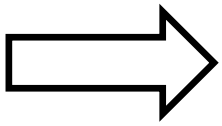
1 - 7 - 4 - 9 - 5 - 8 - 3 - 10 - 2 - 6 - 1 και συνολικό κόστος λύσης 275.29. Έστω ότι διαγράφονται τα τόξα **5 - 8** και **7 - 1**. Σε αυτή την περίπτωση έχουμε δημιουργήσει δύο διαφορετικούς κύκλους, τον **7 - 4 - 9 - 5 - 7** και τον κύκλο **1 - 6 - 2 -10 - 3 - 8 - 1.** Ο πιο εφικτός τρόπος υπολογισμού του νέου κόστους είναι ο υπολογισμός της διαφοράς του κόστους των τόξων που εξέρχονται με το κόστος που εισέρχονται. Αν το κόστος είναι θετικό τότε δεχόμαστε τη λύση, διαφορετικά την απορρίπτουμε. Έτσι στο παράδειγμα μας που διαγράφονται τα τόξα 5 - 8 και 7 - 1 και προστίθενται τα τόξα 7 - 8 και 1 - 5 θα έχουμε: $c_{58} + c_{71} - c_{78} - c_{15}$ και υπολογίζουμε το νέο κόστος. Η διαδικασία επαναλαμβάνεται μέχρι να ελεγχθούν όλοι οι δυνατοί συνδυασμοί.

3.4.2 Ανταλλαγή Μεταξύ Κόμβων Διαφορετικών Διαδρομών

Στο πρώτο μισό του κώδικα τοπικής αναζήτησης, εξετάζουμε αν μπορεί να βελτιωθεί το συνολικό κόστος της διαδρομής με την ταυτόχρονη ανταλλαγή 2 κόμβων διαφορετικών διαδρομών. Πιο συγκεκριμένα, επιλέγονται 2 κόμβοι, πραγματοποιείται ταυτόχρονη ανταλλαγή, και η νέα συνολική λύση που προκύπτει ελέγχεται ως προς την ικανοποίηση των περιορισμών. Σε περίπτωση που κανένας περιορισμός δεν παραβιάζεται, η νέα λύση ελέγχεται σε σύγκριση με την υπάρχουσα για να διαπιστώσουμε αν μειώνεται το συνολικό κόστος.

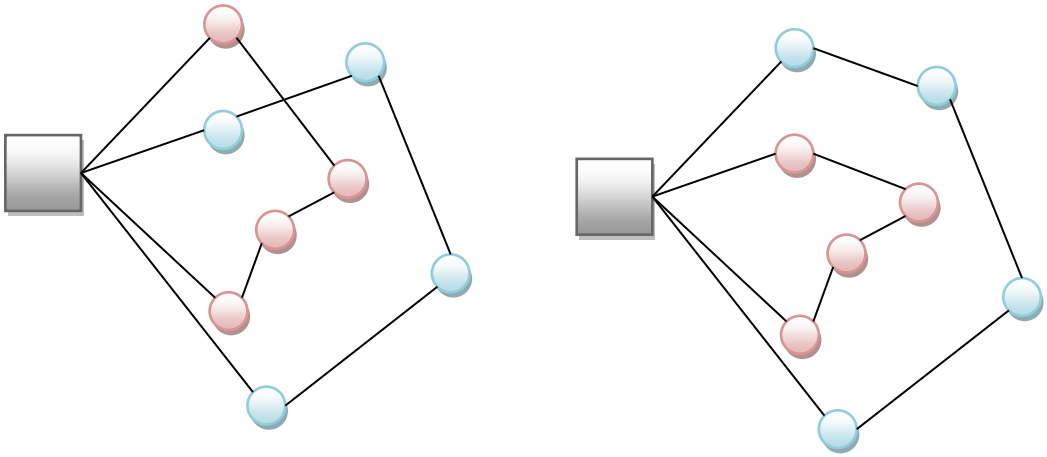
Η βασική ιδέα είναι ότι αν αλλάξει η σειρά εξυπηρέτησης ανάμεσα σε 2 πελάτες, χωρίς την παραβίαση των περιορισμών, υπάρχει η πιθανότητα να μειωθεί η συνολική απόσταση και η λύση μας να καταλήξει σε τοπικό ελάχιστο. Υπάρχουσες υλοποιήσεις της μεθόδου περιορίζουν την εξέταση πιθανών ανταλλαγών σε μία γειτονία λύσεων, με την τυχαία επιλογή κάποιου κόμβου. Στην παρούσα εργασία, πραγματοποιείται εξαντλητική αναζήτηση όλων των πιθανών συνδυασμών. Πιο συγκεκριμένα, για κάθε γραμμή Α εξετάζουμε την πιθανή ανταλλαγή των στοιχείων της Α με όλα τα στοιχεία κάθε άλλης γραμμής Β. Η διαδικασία αυτή περιγράφεται στον Αλγόριθμο 3.4 παρακάτω και παρουσιάζεται στα παρακάτω σχήματα.

1	**15**	**8**	**22**	**1**
1	**2**	**7**	**13**	**1**



1	**2**	**8**	**22**	**1**
1	**15**	**7**	**13**	**1**

Σχήμα 3.4.2.1: *Αποτελέσματα ανταλλαγής μεταξύ κόμβων διαφορετικών διαδρομών*



Σχήμα 3.4.2.2: *Αποτελέσματα ανταλλαγής μεταξύ κόμβων διαφορετικών διαδρομών*

Αλγόριθμος 3.4: Υπορουτίνα 1-1 ανταλλαγής κόμβων μεταξύ διαφορετικών διαδρομών

route ← Αρχική εφικτή λύση από πρώτο κομμάτι της GRASP

Για κάθε γραμμή Α του πίνακα route

route_to_check_A ← Διαδρομή που αντιστοιχεί στην γραμμή Α

Για κάθε πελάτη Χ της διαδρομής Α

Για κάθε γραμμή Β του πίνακα route

Σχόλιο: Θέλουμε ανταλλαγή μεταξύ διαφορετικών διαδρομών

Αν η γραμμή Β δεν είναι ίδια με τη γραμμή Α

Για κάθε πελάτη Υ της διαδρομής Β

Αντικατέστησε πελάτη Χ της Α με πελάτη Υ της Β

Ανανέωσε τον πίνακα της λύσης route

valid_route_A ← 'Ελεγξε αν η νέα λύση για τη διαδρομή Α ικανοποιεί τους περιορισμούς

valid_route_B ← 'Ελεγξε αν η νέα λύση για τη διαδρομή Β ικανοποιεί τους περιορισμούς

Αν γραμμή Α και γραμμή Β έγκυρες

total_cost_new ← Υπολόγισε συνολικό κόστος νέας λύσης

Αν το νέο κόστος total_cost_new μικρότερο από το αρχικό

Κράτα τη νέα λύση

Αλλιώς

Σχόλιο: Η ανταλλαγή δεν βελτιώνει, επιστροφή στην προηγούμενη (πριν την ανταλλαγή)

Αντικατέστησε πελάτη Χ της Α με πελάτη Υ της Β

Ανανέωσε τον πίνακα της λύσης route

Τέλος Αν

Αλλιώς

Σχόλιο: Η λύση δεν είναι έγκυρη, επιστροφή στην προηγούμενη (πριν την ανταλλαγή)

Αντικατέστησε πελάτη Χ της Α με πελάτη Υ της Β

Ανανέωσε τον πίνακα της λύσης route

Τέλος Αν

Τέλος Για

Τέλος Αν

Τέλος Για

Τέλος Για

Τέλος Για

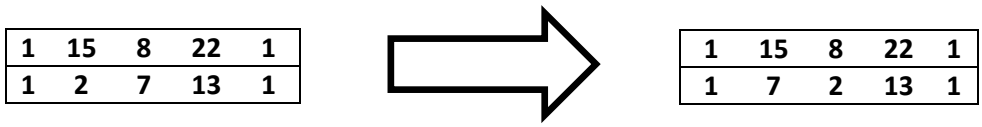
Με βάση τον παραπάνω αλγόριθμο, παρατηρούμε ότι η βέλτιστη λύση δεν μπορεί να πάρει τη λύση που προκύπτει από μια ανταλλαγή που παραβιάζει τους περιορισμούς. Συνεπώς, θα πρέπει η εν λόγω ανταλλαγή να αναιρεθεί. Για την αναίρεση της ανταλλαγής, να σημειωθεί ότι ο πίνακας route επιστρέφει στην προηγούμενη του μορφή και όχι στην αρχική του μορφή. Ο λόγος είναι ότι μέχρι εκείνο το σημείο οι προηγούμενες επαναλήψεις ίσως έχουν δώσει ήδη μια καλύτερη λύση. Διατηρώντας λοιπόν τα επιμέρους βήματα που οδηγούν σε καλύτερες λύσεις, καταλήγουμε στην τελική γειτονιά με το "τελικό" τοπικό ελάχιστο. Επίσης, παρατηρούμε ότι οι εσωτερικοί βρόχοι αφορούν όλους τους πελάτες Χ και Υ των διαδρομών Α και Β αντίστοιχα, μη λαμβάνοντας υπόψη το αρχικό σημείο εκκίνησης από την αποθήκη και το τελικό σημείο επιστροφής σε αυτή. Συνεπώς, οι εσωτερικοί βρόχοι πραγματοποιούν ανταλλαγές στοιχείων από την δεύτερη έως την προτελευταία στήλη ανά διαδρομή.

Παρόλο που μια εξαντλητική εξέταση όλων των συνδυασμών συνιστά πιθανή αύξηση της υπολογιστικής πολυπλοκότητας, στα πλαίσια του προβλήματός μας δεν παρατηρείται τέτοιο πρόβλημα. Ο λόγος είναι ότι υπάρχει πεπερασμένος αριθμός γραμμών της δομής route, καθώς έχουμε πεπερασμένο αριθμό οχημάτων. Συνεπώς έχουμε μικρό αριθμό επαναλήψεων του εξωτερικού επαναληπτικού βρόχου. Το ίδιο ισχύει και για τον εσωτερικό βρόχο, καθώς ο αριθμός των πελατών, και κατά συνέπεια των στηλών του πίνακα route, είναι πεπερασμένος.

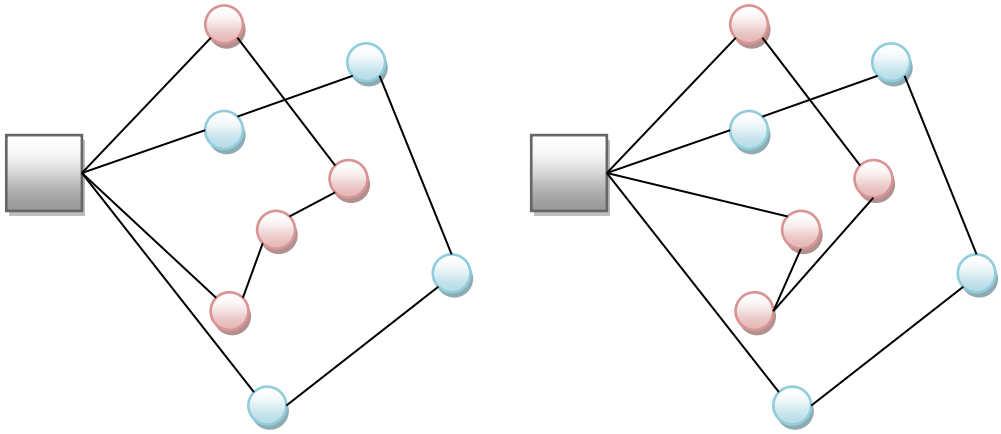
3.4.3 Ανταλλαγή Κόμβων Ίδιας Διαδρομής

Το προηγούμενο τμήμα ανταλλαγής κόμβων μεταξύ διαφορετικών διαδρομών μας επιστρέφει το "συνολικό" τοπικό ελάχιστο, δηλαδή μια coarse-grained γειτονία αναζήτησης, που είναι βέλτιστη ως προς όλες τις διαδρομές. Η λύση αυτή μπορεί να βελτιωθεί περαιτέρω, με την εύρεση fine-grained ελαχίστων ανά διαδρομή. Για το σκοπό αυτό πραγματοποιούμε ανταλλαγές μεταξύ όλων των στοιχείων κάθε επιμέρους διαδρομής. Η διαδικασία είναι παρόμοια με αυτή που περιγράφηκε στην προηγούμενη υποενότητα, καθώς ανά ανταλλαγή ελέγχουμε αν η νέα λύση ικανοποιεί τους περιορισμούς. Αν αυτό συμβαίνει, ελέγχουμε αν αντιστοιχεί σε λύση με λιγότερο κόστος και η τελική λύση ανανεώνεται κατάλληλα. Οι ανταλλαγές πραγματοποιούνται διαδοχικά για όλες τις σειρές και για όλα τα στοιχεία κάθε σειράς.

Μία απλή υλοποίηση συνιστά τη χρήση 3 βρόχων, ενός εξωτερικού για την εξέταση όλων των διαδρομών και 2 εσωτερικών για την ανταλλαγή ενός στοιχείου X με ένα στοιχείο Y. Μπορούμε στο σημείο αυτό να κάνουμε μία ενδιαφέρουσα παρατήρηση. Έστω ότι ένας πελάτης X ανταλλαγεί με τον επόμενο πελάτη στη διαδρομή και η τελική λύση δεν είναι εφικτή διότι ο πελάτης "χάνει" το χρονικό παράθυρο στο οποίο πρέπει να εξυπηρετηθεί. Εύκολα λοιπόν συμπεραίνουμε ότι η οποιαδήποτε άλλη τοποθέτηση αργότερα στη διαδρομή του πελάτη αυτού X θα επιστρέψει επίσης μη εφικτή λύση. Με άλλα λόγια, μπορούμε να εξετάσουμε την ανταλλαγή μόνο με τον επόμενο πελάτη στη διαδρομή και να μειώσουμε δραστικά τον αριθμό των επαναλήψεων. Η διαδικασία αυτή περιγράφεται στα παρακάτω σχήματα και στον αλγόριθμο που ακολουθούν.



Σχήμα 3.4.3.1: *Αποτελέσματα ανταλλαγής μεταξύ κόμβων ίδιων διαδρομών*



Σχήμα 3.4.3.2: *Αποτελέσματα ανταλλαγής μεταξύ κόμβων ίδιων διαδρομών*

Αλγόριθμος 3.4 : Υπορουτίνα 1-1 ανταλλαγής κόμβων της ίδιας διαδρομής

route ← "Βελτιωμένη" λύση από την εκτέλεση του Αλγορίθμου 3.4

Για κάθε γραμμή Α του πίνακα route

route_to_check_A ← Διαδρομή που αντιστοιχεί στην γραμμή Α

Για κάθε πελάτη Χ της διαδρομής Α

Αντικατέστησε πελάτη Χ με τον επόμενο πελάτη Χ+1 της διαδρομής Α

Ανανέωσε τον πίνακα της λύσης route

valid_route ← 'Ελεγξε αν η νέα λύση για τη διαδρομή Α ικανοποιεί τους περιορισμούς

Αν λύση είναι έγκυρη

total_cost_new ← Υπολόγισε συνολικό κόστος νέας λύσης

Αν το νέο κόστος total_cost_new μικρότερο από το αρχικό

Κράτα τη νέα λύση

Αλλιώς

Σχόλιο: Η ανταλλαγή δεν βελτιώνει, επιστροφή στην προηγούμενη (πριν την ανταλλαγή)

Αντικατέστησε πελάτη Χ της Α με πελάτη Χ+1

Ανανέωσε τον πίνακα της λύσης route

Τέλος Αν

Αλλιώς

Σχόλιο: Η λύση δεν είναι έγκυρη, επιστροφή στην προηγούμενη (πριν την ανταλλαγή)

Αντικατέστησε πελάτη Χ της Α με πελάτη Χ+1

Ανανέωσε τον πίνακα της λύσης route

Τέλος Αν

Τέλος Για

Τέλος Για

Και σε αυτήν την περίπτωση δεν επιλέγουμε τυχαία κάποιο στοιχείο της διαδρομής Α, αλλά εξετάζονται όλοι οι πελάτες όλων των διαδρομών. Όπως εξηγήσαμε παραπάνω, με βάση τις πεπερασμένες διαστάσεις της λύσης του προβλήματος, η εξαντλητική υλοποίηση δεν αυξάνει την υπολογιστική πολυπλοκότητα.

ΚΕΝΗ ΣΕΛΙΔΑ

Κεφάλαιο 4
Περιγραφή και Ανάλυση Αποτελεσμάτων

4.1 Γενική Περιγραφή των Αποτελεσμάτων

Για την αποτελεσματικότητα και τη σωστή λειτουργία του αλγορίθμου εκτελέστηκαν μια σειρά από προβλήματα, τα οποία είναι γνωστά χρησιμοποιούμενα προβλήματα **Solomon[7]**. Από τα προβλήματα αυτά τα δεδομένα μας ήταν ο αριθμός των πελατών, η μέγιστη χωρητικότητα των οχημάτων, οι τετμημένες και τεταγμένες των πελατών, τα χρονικά παράθυρα, η ζήτηση, ο χρόνος εξυπηρέτησης, ο μέγιστος διαθέσιμος αριθμός οχημάτων και ο χρόνος που αντιστοιχεί με την απόσταση μεταξύ των πελατών. Εξετάσαμε συνολικά 56 προβλήματα και σε όλα μας τα προβλήματα σταθερά ήταν ο αριθμός των πελατών, που ήταν ίσος με 100, η χωρητικότητα του κάθε οχήματος που ισούται με 200 και ο αριθμός των διαθέσιμων οχημάτων που ήταν ίσος με 25.

Τα 56 προβλήματα του Solomon χωρίζονται σε 6 επιμέρους κατηγορίες προβλημάτων. Ο διαχωρισμός τους γίνεται ανάλογα με τη χωρητικότητα, το μέγιστο χρονικό παράθυρο της αποθήκης, το χρόνο εξυπηρέτησης, τη διασπορά των πελατών και το εύρος των χρονικών παραθύρων. Οι κατηγορίες των προβλημάτων είναι **C1, C2, R1,R2, RC1, RC2** με την κάθε κατηγορία να περιέχει από 8 έως 12 προβλήματα.

Στις κατηγορίες **C1, C2** οι πελάτες είναι τοποθετημένοι ομοιόμορφα. Η κατηγορία **C1** περιέχει αυστηρά χρονικά παράθυρα και μέγιστη χωρητικότητα των οχημάτων ίση με 200. Ο χρόνος εξυπηρέτησης των πελατών ισούται με 90 μονάδες. Στα προβλήματα **C2** τα παράθυρα είναι πιο ανοιχτά, η μέγιστη χωρητικότητα ισούται με 700 μονάδες και ο χρόνος εξυπηρέτησης ισούται με 90 μονάδες. Στην κατηγορία αυτών των προβλημάτων αυξάνεται και ο χρόνος κλεισίματος της αποθήκης. Στις κατηγορίες προβλημάτων **R1** και **R2** οι πελάτες είναι κατανεμημένοι τυχαία και ανομοιόμορφα. Ο χρόνος εξυπηρέτησης είναι κοινός και ισούται με 10 μονάδες. Η χωρητικότητα στα προβλήματα **R1** είναι 200 μονάδες και τα χρονικά παράθυρα είναι κλειστά, ενώ στα προβλήματα **R2** είναι 1000 μονάδες και τα χρονικά παράθυρα είναι χαλαρά**.** Στα προβλήματα **RC** υπάρχει συνδυασμός των δύο προηγούμενων κατηγοριών. Κάποιοι από τους πελάτες είναι τοποθετημένοι σε ομάδες και άλλοι τυχαία. Τα προβλήματα της κατηγορίας **RC1** έχουν σχετικά κλειστά παράθυρα και τα προβλήματα της κατηγορίας **RC2** έχουν κάποιους πελάτες με πιο κλειστά παράθυρα και κάποιους με πιο χαλαρά. Ο χρόνος εξυπηρέτησης είναι 10 μονάδες, ενώ η μέγιστη χωρητικότητα για τα προβλήματα **RC1** είναι 200 μονάδες και 1000 μονάδες για τα προβλήματα **RC2**. Ο βραδύτερος χρόνος στα προβλήματα **RC2** είναι πολύ μεγαλύτερος από τα προβλήματα **RC1.**

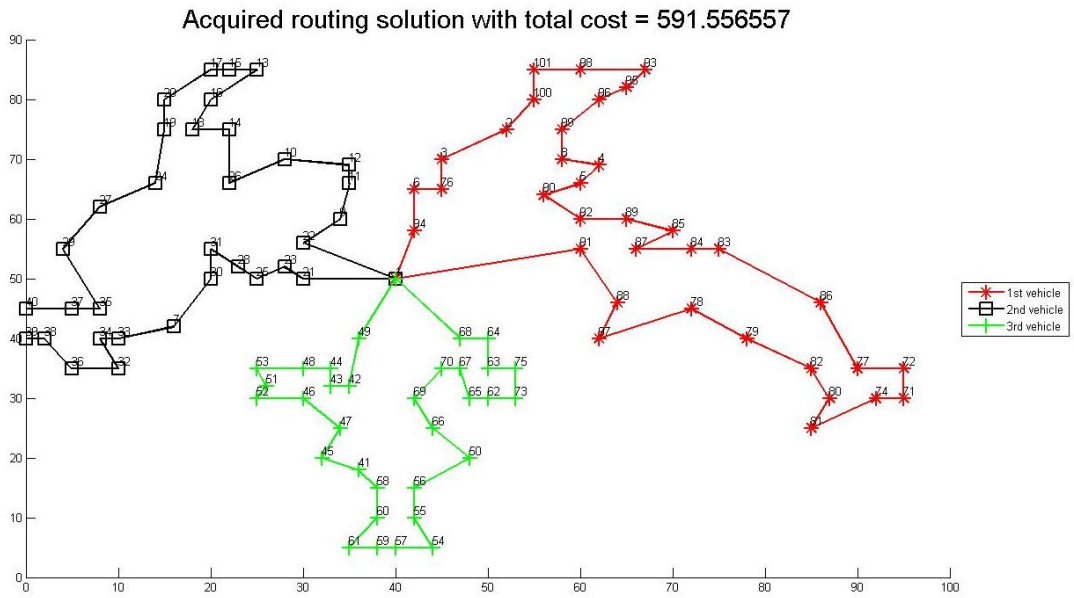
Το κάθε πρόβλημα εκτελέστηκε για 1.000.000 επαναλήψεις, 7-10 φορές το καθένα ξεχωριστά. Από αυτές κρατήσαμε το καλύτερο αποτέλεσμα και την καλύτερη λύση

που προέκυψε. Αναλυτικά το βέλτιστο κόστος και η βέλτιστη διαδρομή για κάθε κατηγορία προβλήματος παρουσιάζονται στο παράρτημα.

4.2 Παρουσίαση Αποτελεσμάτων των Προβλημάτων

Σε αυτή την ενότητα, γίνεται μια παρουσίαση ορισμένων αποτελεσμάτων από τα γνωστά χρησιμοποιούμενα προβλήματα **Solomon[7]** που εκτελέστηκαν.

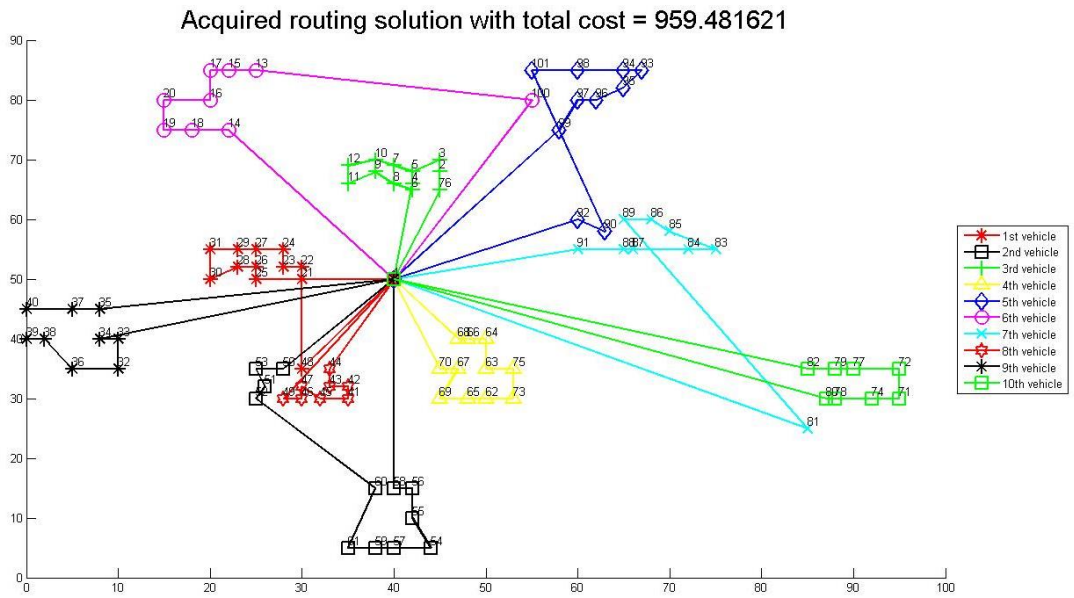
Αποτελέσματα προβλήματος C201



Σχήμα 4.2.Α: *Δρομολόγηση οχημάτων C201*

Από το παραπάνω σχήμα για το πρόβλημα **C201**, όπου βρήκαμε βέλτιστη λύση, παρατηρούμε μια ομαλή αλληλουχία στις διαδρομές μας. Τα οχήματα που χρησιμοποιήθηκαν σε αυτό το πρόβλημα ακολουθούν μια ομαλή πορεία, χωρίς να παρεκκλίνουν από τη γειτονία πελατών που εξυπηρετούν. Έτσι η εξυπηρέτηση των πελατών γίνεται από λιγότερα οχήματα και η τελική πορεία που ακολουθούν γίνεται εύκολα αντιληπτή.

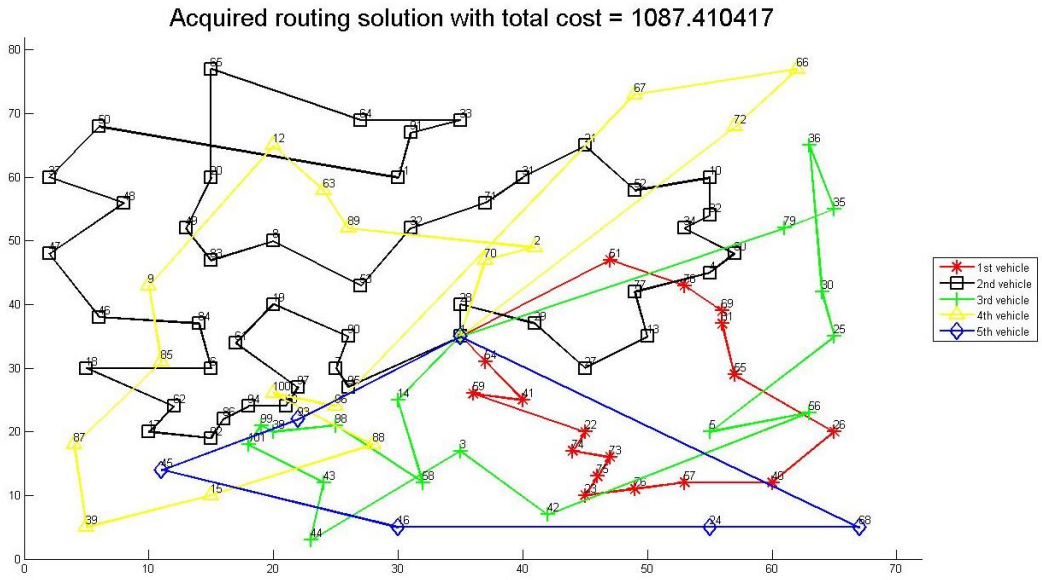
Αποτελέσματα προβλήματος C105



Σχήμα 4.2.Β: *Δρομολόγηση οχημάτων C105*

Στο σχήμα 4.2.Β που παρουσιάζονται τα αποτελέσματα του προβλήματος **C105**, παρατηρούμε μία λύση που δεν είναι βέλτιστη, αλλά αποκλίνει από αυτή σε μικρό ποσοστό, της τάξης του 15.74%. Από το γράφημα, αυτό είναι εύκολα αντιληπτό καθώς αν και τα οχήματα που χρησιμοποιούνται για να εξυπηρετήσουν τους πελάτες ακολουθούν μια ομαλή πορεία και εξυπηρετούν ένα πλήθος γειτονικών πελατών, ωστόσο σε ορισμένες περιπτώσεις παρατηρούμε μια έντονη απόκλιση από τη γειτονία. Αυτό έχει σαν αποτέλεσμα να καθιστά τη λύση μας λιγότερο αποτελεσματική, άρα και μη βέλτιστη, δίνοντας μας και τελικά μία λύση με μεγαλύτερο κόστος.

Αποτελέσματα προβλήματος R208



Σχήμα 4.2.C: *Δρομολόγηση οχημάτων R208*

Σε ότι αφορά το σχήμα 4.2.C του προβλήματος κατηγορίας **R2**, γίνεται εύκολα αντιληπτό πως η λύση που επιτύχαμε με τον αλγοριθμό μας έχει μεγάλη απόκλιση από το βέλτιστο που επιδιώκαμε. Το κάθε όχημα που αντιστοιχεί σε μία διαδρομή πελατών δεν εξυπηρετεί μια γειτονία πελατών. Αντίθετα τα οχήματα εξυπηρετούν πελάτες που βρίσκονται μακριά ο ένας από τον άλλον, με αποτέλεσμα η μετάβαση να μην είναι ομαλή και βέβαια να απεικονίζεται αυτό σχηματικά. Επίσης τα οχήματα διανύουν μεγαλύτερες αποστάσεις και αυτό έχει σαν αποτέλεσμα μεγαλύτερο συνολικό κόστος.

Όπως αναφέραμε και προηγούμενα όμως καθε κατηγορία προβλήματος έχει διαφορετικά χαρακτηριστικά. Δηλαδή στις κατηγορίες **C1, C2** οι πελάτες είναι τοποθετημένοι ομοιόμορφα, ενώ στις κατηγορίες προβλημάτων **R1** και **R2** οι πελάτες είναι κατανεμημένοι τυχαία και ανομοιόμορφα. Αυτό σε μεγάλο βαθμό επηρρεάζει και την τελική μας λύση.

Αναλυτικά τα αποτελέσματα και για τα 56 προβλήματα **Solomon[7]**, παρουσιάζονται στο παράρτημα στο τέλος της διπλωματικής. Σε αυτά γίνεται αναλυτικά τόσο η σχηματική απεικόνιση του κάθε προβλήματος, η παρουσιάση σε πίνακες των πελατών που εξυπηρετεί κάθε όχημα καθώς και η αναφορά του συνολικού κόστους κάθε προβλήματος.

4.3 Πίνακες Αποτελεσμάτων

Παραπάνω παρουσιάστηκαν τα γραφήματα και οι πίνακες διαδρομών από ένα σύνολο προβλημάτων. Στη συνέχεια συνοψίζονται όλα τα αποτελέσματα στους παρακάτω πίνακες που περιλαμβάνουν την αρχική εφικτή λύση, τη βέλτιστη λύση που βρέθηκε από το συνδυασμό της 2-opt και της GRASP και η παγκόσμια βέλτιστη λύση.

Προβλήματα	Αλγόρθμος Απληστίας	Οχήματα Αλγορίθμου Απληστίας	Grasp & 2-opt	Οχήματα Grasp & 2-opt	Παγκόσμια βέλτιστη λύση	Οχήματα Παγκόσμιας βέλτιστης
C101	1870.69	21	895.88	10	828.94	10
C102	1289.35	13	1219.36	13	828.24	10
C103	1748.41	17	1479.91	13	828.06	10
C104	1888.28	20	964.07	10	824.78	10
C105	1888.38	20	959.48	10	828.94	10
C106	1913.08	19	971.09	10	828.94	10
C107	2131.53	18	919.95	10	828.94	10
C108	2103.40	17	970.62	10	828.24	10
C109	2013.48	17	969.36	10	828.24	10
C201	1880.46	15	591.56	3	591.56	3
C202	1603.09	14	1047.76	4	591.56	3
C203	1583.70	11	1118.43	5	591.17	3
C204	1446.73	9	1113.53	5	590.60	3
C205	1485.09	9	589.72	3	588.88	3
C206	1267.88	8	629.59	3	588.49	3
C207	1292.97	8	666.84	4	588.29	3
C208	1145.22	6	629.12	3	588.32	3
R101	1972.04	25	1702.38	19	1650.80	19
R102	1519.34	18	1497.95	18	1486.12	17
R103	1647.94	15	1457.32	13	1292.68	13
R104	1505.36	16	1389.12	15	1007.31	9
R105	1748.41	17	1465.49	14	1377.11	14
R106	2152.40	25	1730.55	17	1252.03	12
R107	1749.90	19	1485.74	14	1104.66	10
R108	1505.36	16	1386.42	15	960.88	9
R109	1853.10	15	1252.09	13	1194.73	11
R110	1564.93	14	1285.47	13	1118.84	10
R111	1522.47	14	1285.47	13	1096.72	10
R112	1364.93	14	1247.78	13	982.14	9
R201	1984.95	15	1603.33	7	1252.37	4
R202	1812.99	14	1584.39	6	1191.70	3
R203	1532.72	11	1437.46	6	939.50	3
R204	1603.09	14	1084.19	4	825.52	2
R205	1446.73	9	1142.30	5	994.42	3
R206	1603.09	14	1095.01	4	906.14	3
R207	1723.02	14	1095.33	4	890.61	2
R208	1104.46	5	1087.41	5	726.82	2
R209	1404.35	5	1062.11	5	909.16	3

R210	1568.64	5	1114.57	3	939.37	3
R211	1368.64	5	1114.57	3	885.71	2
RC101	2685.08	25	1956.30	18	1696.94	14
RC102	2596.10	25	2025.14	19	1554.75	12
RC103	2025.36	21	1412.85	15	1261.67	11
RC104	1743.86	16	1536.42	14	1135.48	10
RC105	2119.21	20	1705.88	14	1629.44	13
RC106	2061.53	20	1775.67	14	1424.73	11
RC107	1952.32	18	1797.67	16	1230.48	11
RC108	2103.40	17	1218.67	11	1139.82	10
RC201	2468.24	15	1736.94	6	1406.94	4
RC202	2114.34	13	1741.11	6	1365.65	3
RC203	1361.20	7	1223.56	6	1049.62	3
RC204	1267.88	8	888.92	4	798.46	3
RC205	2073.99	13	1440.93	6	1297.65	4
RC206	1583.70	11	1342.21	5	1146.32	3
RC207	1725.90	5	1084.19	4	1061.14	3
RC208	1395.02	5	1252.89	4	828.14	3

Πίνακας 4.3.1: *Συγκεντρωτικά αποτελέσματα*

Κεφάλαιο 5
Υπολογιστικά Αποτελέσματα

5.1 Γενική Περιγραφή των Αποτελεσμάτων

Στο προηγούμενο κεφάλαιο παρουσιάστηκαν τα αποτελέσματα των αλγορίθμων απληστίας και της τοπικής αναζήτησης. Στο κεφάλαιο αυτό θα γίνει μια αξιολόγηση αυτών των αποτελεσμάτων. Στη συνέχεια παρουσιάζονται σε πίνακες ανά κατηγορία προβλημάτων τα συγκριτικά αποτελέσματα ανάμεσα στην αρχική λύση και τη βέλτιστη λύση της και ανάμεσα στη βέλτιστη και την παγκοσμίως βέλτιστη. Η σύγκριση των λύσεων αναγράφεται σε ποσοστιαία (%) απόκλιση.

Προβλήματα κατηγορίας C1

Προβλήματα	Αλγόρθμος Απληστίας	Grasp & 2-opt	Βελτίωση Λύσης (%)	Βελτίωση οχημάτων	Παγκόσμια βέλτιστη λύση	Απόκλιση (%)
C101	1870.69	895.88	52.13%	21→10	828.94	8.07%
C102	1289.35	1219.36	5.46%	13→13	828.24	47.22%
C103	1748.41	1479.91	15.38%	17→13	828.06	78.72%
C104	1888.28	964.07	48.96%	20→10	824.78	16.88%
C105	1888.38	959.48	49.12%	20→10	828.94	15.74%
C106	1913.08	971.09	49.28%	19→10	828.94	17.14%
C107	2131.53	919.95	56.81%	18→10	828.94	10.97%
C108	2103.40	970.62	53.85%	17→10	828.24	17.15%
C109	2013.48	969.36	51.82%	17→10	828.24	17.03%

Πίνακας 5.1.1: *Σύγκριση αποτελεσμάτων προβλημάτων C1*

Σε αυτή την κατηγορία προβλημάτων, όπου έχουμε αυστηρά χρονικά παράθυρα, παρατηρούμε μια αρκετά ικανοποιητική λύση σε σχέση με το παγκόσμιο βέλτιστο. Και στα εννιά διαφορετικά προβλήματα καταφέραμε και μειώσαμε ή διατηρήσαμε ίδιο τον αριθμό των οχημάτων με μέσο όρο μείωσης 9.78.

Προβλήματα κατηγορίας C2

Προβλήματα	Αλγόρθμος Απληστίας	Grasp & 2-opt	Βελτίωση Λύσης (%)	Βελτίωση οχημάτων	Παγκόσμια βέλτιστη λύση	Απόκλιση (%)
C201	1880.46	591.56	68.57%	15→3	591.56	0.0%
C202	1603.09	1047.76	34.62%	14→4	591.56	67.11%
C203	1583.70	1118.43	29.33%	11→5	591.17	75.72%
C204	1446.73	1113.53	23.19%	9→5	590.60	77.88%
C205	1485.09	589.72	60.30%	9→3	588.88	0.14%
C206	1267.88	629.59	50.36%	8→3	588.49	6.98%
C207	1292.97	666.84	48.47%	8→4	588.29	13.35%
C208	1145.22	629.12	45.18%	6→3	588.32	6.93%

Πίνακας 5.1.2: *Σύγκριση αποτελεσμάτων προβλημάτων C2*

Σε αυτή την κατηγορία προβλημάτων, παρατηρούμε μια μεγαλύτερη διακύμανση των λυσεών μας. Συγκεκριμένα στο πρόβλημα **C201** καταφέραμε και πετύχαμε τη βέλτιστη λύση, ενώ σε προβλήματα όπως τα **C204** είχαμε απόκλιση που αγγίζει το 77.88%. Ωστόσο και σε αυτά τα προβλήματα καταφέραμε να βελτιώσουμε σε μεγάλο ποσοστό την αρχική μας λύση μέσω των μεθόδων **Grasp & 2-opt** ,με ποσοστό της τάξης του 68.57%. Τέλος και στα οκτώ διαφορετικά προβλήματα αυτής της κατηγορίας είχαμε σημαντική μείωση των οχημάτων με μέσο όρο μείωσης 8.13.

Προβλήματα κατηγορίας R1

Προβλήματα	Αλγόρθμος Απληστίας	Grasp & 2-opt	Βελτίωση Λύσης (%)	Βελτίωση οχημάτων	Παγκόσμια βέλτιστη λύση	Απόκλιση (%)
R101	1972.04	1702.38	13.67%	25→19	1650.80	3.12%
R102	1519.34	1497.95	1.40%	18→18	1486.12	0.79%
R103	1647.94	1457.32	11.56%	15→13	1292.68	12.73%
R104	1505.36	1389.12	7.72%	16→15	1007.31	37.9%
R105	1748.41	1465.49	16.18%	17→14	1377.11	6.41%
R106	2152.40	1730.55	19.59%	25→17	1252.03	38.21%
R107	1749.90	1485.74	15.1%	19→14	1104.66	34.49%
R108	1505.36	1386.42	7.9%	16→15	960.88	44.2%
R109	1853.10	1252.09	32.43%	15→13	1194.73	4.8%
R110	1564.93	1285.47	17.85%	14→13	1118.84	14.89%
R111	1522.47	1285.47	15.56%	14→13	1096.72	17.21%
R112	1364.93	1247.78	8.58%	14→13	982.14	27.04%

Πίνακας 5.1.3: *Σύγκριση αποτελεσμάτων προβλημάτων R1*

Σε ότι αφορά τα προβλήματα **R1**, παρατηρούμε μια πιο ισορροπημένη διαφοροποίηση στη λύση των μεθόδων **Grasp & 2-opt** σε σχέση με το παγκόσμιο βέλτιστο για τα διάφορα προβλήματα αυτής της κατηγορίας. Βλέπουμε δηλαδή μικρότερη απόκλιση 0.79% και μεγαλύτερη 44.2%. Επίσης σε όλα τα προβλήματα ο κώδικας μας μείωσε τον αριθμό των οχημάτων και τέλος μεγάλο είναι και το ποσοστό βελτίωσης της αρχικής μας λύσης.

Προβλήματα κατηγορίας R2

Προβλήματα	Αλγόρθμος Απληστίας	Grasp & 2-opt	Βελτίωση Λύσης (%)	Βελτίωση οχημάτων	Παγκόσμια βέλτιστη λύση	Απόκλιση (%)
R201	1984.95	1603.33	19.22%	15→7	1252.37	28.02%
R202	1812.99	1584.39	12.6%	14→6	1191.70	32.95%
R203	1532.72	1437.46	6.21%	11→6	939.50	53%
R204	1603.09	1084.19	32.36%	14→4	825.52	31.33%
R205	1446.73	1142.30	21.04%	9→5	994.42	14.87%
R206	1603.09	1095.01	31.69%	14→4	906.14	20.84%
R207	1723.02	1095.33	36.42%	14→4	890.61	22.98%
R208	1104.46	1087.41	1.54%	5→5	726.82	49.61%
R209	1404.35	1062.11	24.36%	5→5	909.16	16.82%
R210	1568.64	1114.57	32.29%	5→3	939.37	18.65%
R211	1368.64	1114.57	18.56%	5→3	885.71	25.83%

Πίνακας 5.1.4: *Σύγκριση αποτελεσμάτων προβλημάτων R2*

Στα **R2** προβλήματα παρατηρούμε για τον κώδικά μας ότι δεν είχαμε σε κανένα από τα έντεκα προβλήματα αυτής της κατηγορίας ικανοποιητική λύση και γι'αυτό παρατηρούμε μεγάλη απόκλιση στη λύση μας και στο παγκοσμίως βέλτιστο. Μια απόκλιση που κυμαίνεται από 16.82% μέχρι το 49.61%. Αυτό οφείλεται σε μεγάλο βαθμό στα αυστηρά χρονικά περιθώρια και στην αυξημένη τυχαιοποίηση των πελατών που συναντάμε σε αυτή την κατηγορία προβλημάτων.

Προβλήματα κατηγορίας RC1

Προβλήματα	Αλγόρθμος Απληστίας	Grasp & 2-opt	Βελτίωση Λύσης (%)	Βελτίωση οχημάτων	Παγκόσμια βέλτιστη λύση	Απόκλιση (%)
RC101	2685.08	1956.30	27.14%	25→18	1696.94	15.28%
RC102	2596.10	2025.14	21.99%	25→19	1554.75	30.25%
RC103	2025.36	1412.85	30.24%	21→15	1261.67	11.98%
RC104	1743.86	1536.42	11.89%	16→14	1135.48	35.31%
RC105	2119.21	1705.88	19.5%	20→14	1629.44	4.69%
RC106	2061.53	1775.67	13.86%	20→14	1424.73	24.63%
RC107	1952.32	1797.67	7.92%	18→16	1230.48	46.09%
RC108	2103.40	1218.67	42.06%	17→11	1139.82	6.91%

Πίνακας 5.1.5: *Σύγκριση αποτελεσμάτων προβλημάτων RC1*

Και σε αυτή την κατηγορία προβλημάτων επιτύχαμε σημαντική μείωση της αρχικής μας λύσης σε μέγιστο ποσοστό της τάξης του 42.06%, όπως και μείωση των οχημάτων μας με μέσο όρο 6.24. Σε ότι αφορά την απόκλισή μας από το βέλτιστο, παρατηρείται μια ικανοποιητική απόκλιση, λόγω της πολυπλοκότητας αυτών των προβλημάτων, καθώς έχουν κλειστά παράθυρα οι πελάτες και είναι τοποθετημένοι άλλοι σε σειρά και άλλοι τυχαία.

Προβλήματα κατηγορίας RC2

Προβλήματα	Αλγόρθμος Απληστίας	Grasp & 2-opt	Βελτίωση Λύσης (%)	Βελτίωση οχημάτων	Παγκόσμια βέλτιστη λύση	Απόκλιση (%)
RC201	2468.24	1736.94	29.62%	15→6	1406.94	23.45%
RC202	2114.34	1741.11	17.65%	13→6	1365.65	27.49%
RC203	1361.20	1223.56	10.11%	7→6	1049.62	16.57%
RC204	1267.88	888.92	29.88%	8→4	798.46	11.32%
RC205	2073.99	1440.93	30.52%	13→6	1297.65	11.04%
RC206	1583.70	1342.21	15.24%	11→5	1146.32	17.08%
RC207	1725.90	1084.19	37.18%	5→4	1061.14	2.17%
RC208	1395.02	1252.89	10.18%	5→4	828.14	51.28%

Πίνακας 5.1.6: *Σύγκριση αποτελεσμάτων προβλημάτων RC2*

Τέλος και για την κατηγορία **RC2** παρατηρούμε μεγάλο ποσοστό βελτίωσης της αρχικής μας λύσης και της μείωσης των οχημάτων μας. Επίσης η λύση μας συγκριτικά με την παγκοσμίως βέλτιστη είναι ικανοποιητική, λόγω όπως αναφέραμε και προηγούμενα της πολυπλοκότητας αυτών των προβλημάτων.

5.2 Ανάλυση Αποτελεσμάτων και Συμπεράσματα

Από τους πίνακες της προηγούμενης ενότητας, μπορούμε να κάνουμε κάποιες παρατηρήσεις βάσει των οποίων μπορεί να εκτιμηθεί η ποιότητα των αποτελεσμάτων και εν συνεχεία των αλγορίθμων. Οι κυριότερες παρατηρήσεις που προκύπτουν από τους πίνακες συγκριτικών αποτελεσμάτων είναι:

1. Ο μέσος όρος απόκλισης από τη παγκοσμίως βέλτιστη λύση είναι 23.15%. Το εύρος απόκλισης κυμαίνεται από 0% το ελάχιστο και 78.72% το μέγιστο.
2. Σε ότι αφορά τα οχήματα χρησιμοποιούμε κατά μέσο όρο 2.58 παραπάνω σε σχέση με το παγκοσμίως βέλτιστο. Το εύρος είναι 0 ο ελάχιστος αριθμός οχημάτων και 6 οχήματα ο μέγιστος.
3. Σημαντική παρατήρηση στην επίλυση μας είναι ο μέσος όρος βελτίωσης της αρχικής μας λύσης. Το ποσοστό της βελτίωσης ανέρχεται για όλες τις κατηγορίες των προβλημάτων στο 41.03%. Το εύρος βελτίωσης της λύσης μας κυμαίνεται ανάμεσα στο 1.40% και φτάνει μέχρι το 68.57%.
4. Μέσω του αλγορίθμου Grasp & 2-opt, βελτιώσαμε σημαντικά τον αριθμό των χρησιμοποιούμενων οχημάτων. Αυτό αριθμητικά αντιστοιχεί σε 6.78% λιγότερα οχήματα συγκριτικά με την αρχική μας λύση. Γενικά παρατηρούμε μια διακύμανση της μείωσης των οχημάτων και σε κάποια προβλήματα μειώνουμε μέχρι και 12 οχήματα ενώ σε κάποια δε βελτιώνεται καθόλου.

Συνολικά, μπορούμε να καταλήξουμε ότι ο αλγόριθμος βελτιστοποίησης με τη χρήση της μεθόδου Grasp & 2-opt είναι αρκετά αποτελεσματικός. Σε ότι αφορά την απόκλιση από το παγκοσμίως βέλτιστο είναι σχετικά μικρή. Σε πολλά προβλήματα

είναι μικρότερη του 10%. Τις μεγαλύτερες αποκλίσεις τις παρατηρούμε στα προβλήματα που εκτελούνται λίγες διαδρομές.

Σε ότι αφορά τον αλγόριθμο Grasp και της μεθόδου 2-opt, είχαμε ικανοποιητικά αποτελέσματα στη βελτίωση των αρχικών εφικτών λύσεων. Ο αλγόριθμος βελτίωσε τη λύση σε όλα τα προβλήματα κατά ένα μεγάλο ποσοστό που πολλές φορές ξεπέρασε και το 60%. Εκεί που ο αλγόριθμος ήταν και αρκετά αποτελεσματικός ήταν στη μείωση των οχημάτων. Αυτό είχε σαν αποτέλεσμα σε πολλές κατηγορίες προβλημάτων να είμαστε κοντά ή και ακριβείς σε σχέση με το παγκοσμίως βέλτιστο.

Αν και σε ένα μεγάλο ποσοστό τα αποτελέσματα που πήραμε ήταν ικανοποιητικά, παρατηρήσαμε πως σε κάποιες περιπτώσεις είχαμε αρκετά μεγάλη απόκλιση από το βέλτιστο. Είναι σίγουρο πως ο κώδικας επιδέχεται βελτίωση και αλλαγές για την αποτελεσματικότερη και καλύτερη χρήση του. Μερικές διορθωτικές αλλαγές θα μπορούσαν να είναι:

1. Αύξηση του αριθμού των επαναλήψεων. Ο αλγόριθμος είδαμε πως βασίζεται στην τυχαιότητα, γι'αυτό και σε κάθε τρέξιμο βρίσκουμε διαφορετικά αποτελέσματα. Άρα οπως είναι φυσιολογικό, με περισσότερες επαναλήψεις μπορούμε να επιτύχουμε επίλυση περισσότερων συνδυασμών λύσεων που θα μας βελτιώσουν το αποτέλεσμα μας.
2. Εισαγωγή επιπλέον στρατηγικών στον αλγόριθμο Grasp. Είδαμε και στη δική μας περίπτωση πως η εισαγωγή περαιτέρω διερεύνσης με τη μέθοδο της 2-opt, έδωσε πολύ καλύτερη λύση από την αρχική.
3. Μείωση της τυχαιότητας στην αναζήτηση γειτονίας. Στο κομμάτι που εκτελούνται οι ανταλλαγές επιλέγεται τυχαία ο επόμενος πελάτης. Αν αντί αυτό, επιλεγόταν να γίνει μία αλλαγή από το τόξο με το μεγαλύτερο κόστος θα βελτιώναμε ακόμα περισσότερο τη λύση μας.

Βιβλιογραφία

[1] Μαρινάκης, Ι., Μυγδαλάς, Α., 2008, "Σχεδιασμός και Βελτιστοποίηση της Εφοδιαστικής Αλυσίδας", Εκδόσεις 'σοφία'.

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

[3] https://www.dhl-discoverlogistics.com/cms/en/course/origin/historical_development.jsp

[4] http://www.logistics.org.gr/4/27/136/

[5] http://neo.lcc.uma.es/dynamic/vrp.html)

[6] Μαρινάκης, Ι., Μαρινάκη, Μ., 2010, "Εξελικτικοί Αλγόριθμοι και Βελτιστοποίηση Συστημάτων Μεγάλης Κλίμακας",

[7] https://www.sintef.no/projectweb/top/vrptw/solomon-benchmark/100-customers/

[8] Toth, P., Vigo, D., 2000, "An overview of Vehicle Routing Problems: Vehicle Routing Problems", Philadelphia, PA, USA, Society for industrial and Applied Mathematics

[9] Küçükoğlu , I., Öztürk,N., 2012, "Computers & Industrial Engineering, An advanced hybrid meta-heuristic algorithm for the vehicle routing problem with backhauls and time windows"

[10] Ombuki B., Ross B., Hanshar F., 2006, "Multi-Objective Genetic Algorithms for Vehicle Routing Problem with Time Windows", Department of Computer Science, Brock Univercity St Catharines.

[11] Shaw P., 1998, "Using Constraint Programming and Local Search Methods to Solve Vehicle Routing Problems", Department of computer science, Univercity of Strathclyde, Glasgow

[12] Zygiaris, S., 2000, Supply Chain Management, INNOREGIO: dissemination of innovation and knowledge management techniques, EC funded project

[13] Larsen A., 2001, "The Dynamic Vehicle Routing Problem", Lyngby

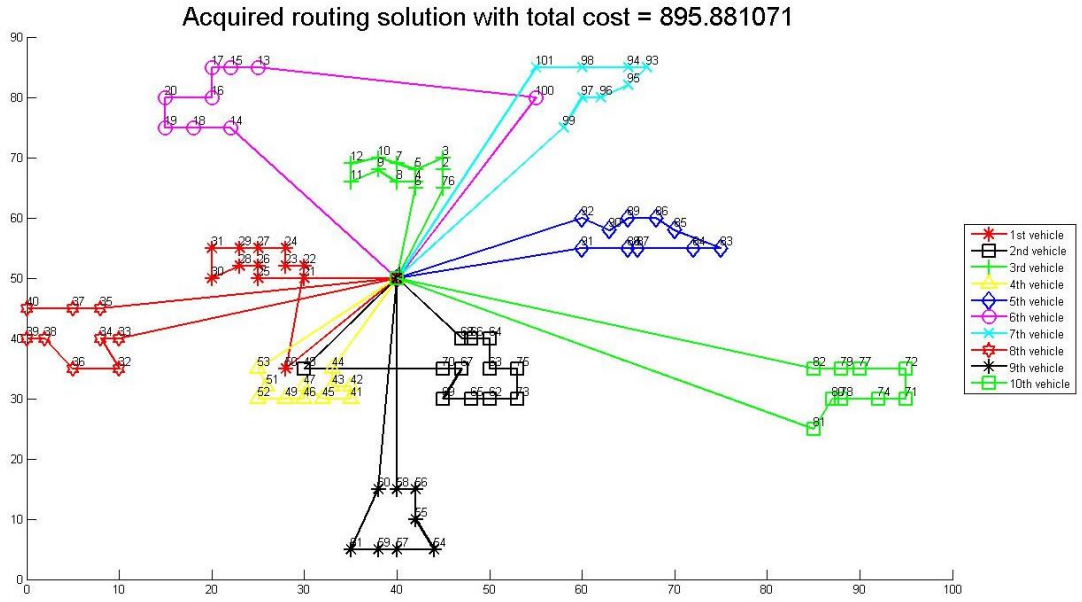
[14] Douglas, M., Martha C. Cooper, Janus D. Pagh, 1998, "Supply Chain Management: Implementation Issues and Research Opportunities", The Inernational journal of Logistics Management, vol. 9No. 2, pp.2

[15] Solomon M.M., 1987, "Algorithms for the vehicle routing and scheduling problems with time windows constraints",Operations Research, Vol. 35, No. 2, p. 254-265

Παράρτημα

Προβλήματα κατηγορίας C1

1. Αποτελέσματα προβλήματος C101

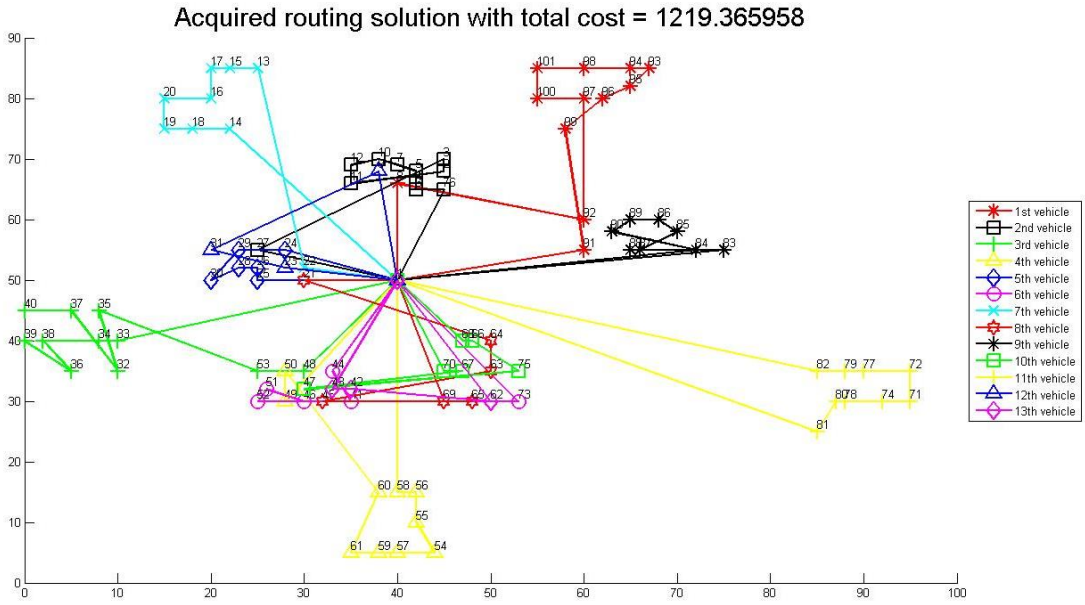


Σχήμα 1: *Δρομολόγηση οχημάτων C101*

ΠΕΛΑΤΕΣ ΑΝΑ ΟΧΗΜΑ													
1	21	25	26	28	30	31	29	27	24	23	22	50	1
1	68	66	64	63	75	73	62	65	69	67	70	48	1
1	6	4	8	9	11	12	10	7	5	3	2	76	1
1	44	43	42	41	45	47	46	49	52	51	53	1	0
1	91	88	87	84	83	85	86	89	90	92	1	0	0
1	14	18	19	20	16	17	15	13	100	1	0	0	0
1	99	97	96	95	93	94	98	101	1	0	0	0	0
1	33	34	32	36	38	39	40	37	35	1	0	0	0
1	58	56	55	54	57	59	61	60	1	0	0	0	0
1	82	79	77	72	71	74	78	80	81	1	0	0	0

Πίνακας 1: *Δρομολόγηση οχημάτων C101*

2. Αποτελέσματα προβλήματος C102

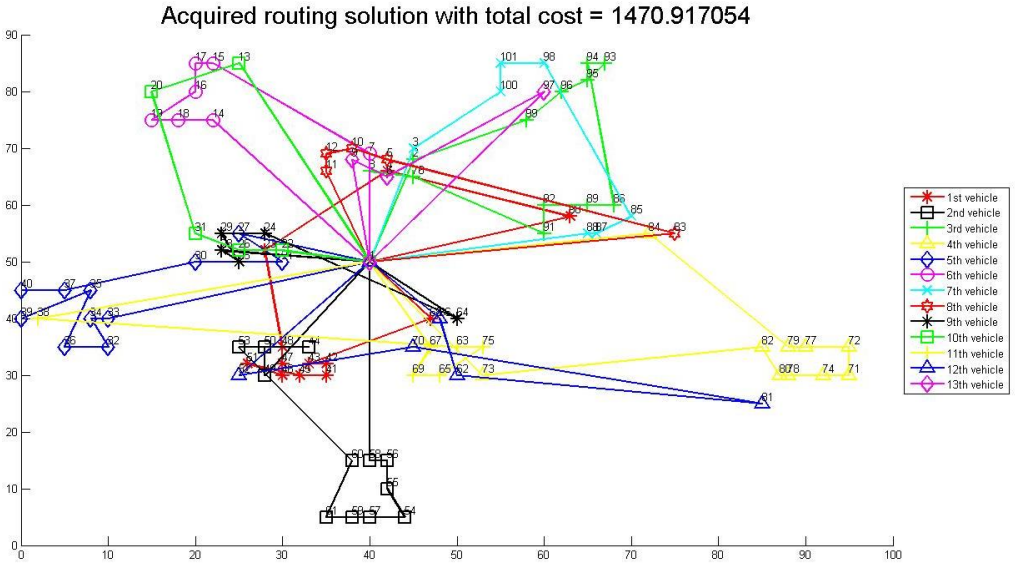


Σχήμα 2: *Δρομολόγηση οχημάτων C102*

ΠΕΛΑΤΕΣ ΑΝΑ ΟΧΗΜΑ													
1	91	88	92	96	93	94	98	101	100	97	95	99	1
1	44	43	83	84	85	86	89	90	6	2	76	1	0
1	68	64	41	45	47	46	52	49	51	50	48	1	0
1	58	56	55	54	57	59	61	60	67	70	1	0	0
1	66	63	73	62	65	69	1	0	0	0	0	0	0
1	9	11	8	7	24	23	1	0	0	0	0	0	0
1	14	18	19	20	16	17	15	13	4	1	0	0	0
1	33	34	38	3	10	5	1	0	0	0	0	0	0
1	21	26	36	39	40	37	35	22	1	0	0	0	0
1	25	42	32	30	31	29	1	0	0	0	0	0	0
1	82	79	77	72	71	74	78	80	81	1	0	0	0
1	53	28	12	1	0	0	0	0	0	0	0	0	0
1	27	87	75	1	0	0	0	0	0	0	0	0	0

Πίνακας 2: *Δρομολόγηση οχημάτων C102*

3.Αποτελέσματα προβλήματος C103

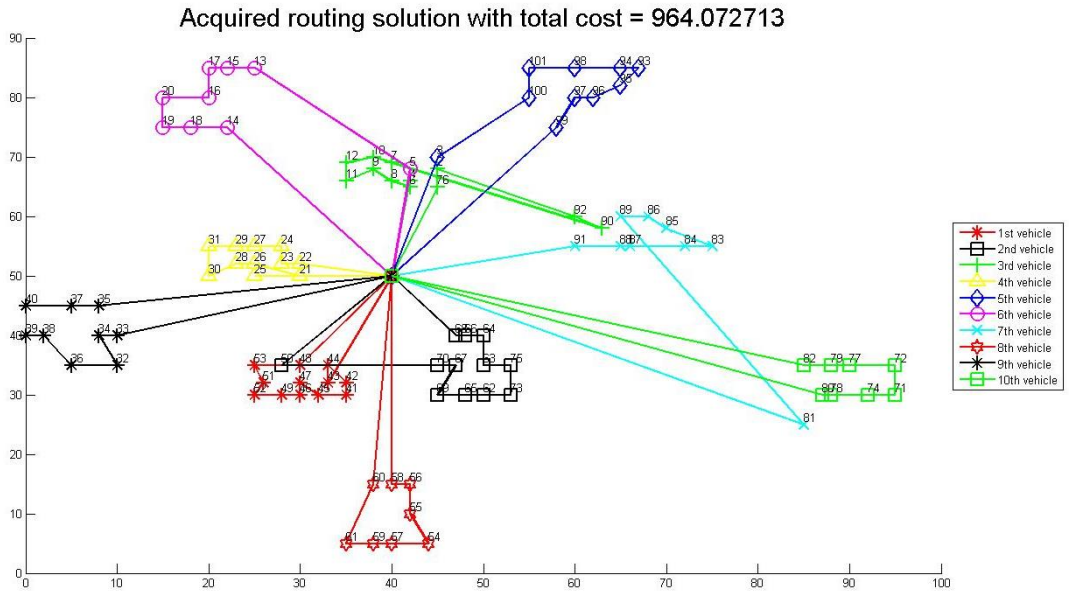


Σχήμα 3: *Δρομολόγηση οχημάτων C103*

ΠΕΛΑΤΕΣ ΑΝΑ ΟΧΗΜΑ													
1	92	88	87	86	83	85	90	89	84	70	50	51	1
1	68	66	64	54	57	59	55	60	61	53	49	48	1
1	6	76	98	96	93	94	95	101	100	97	3	2	1
1	79	77	72	71	74	78	80	81	82	91	1	0	0
1	43	35	36	38	39	40	17	15	13	20	4	1	0
1	42	45	33	30	44	52	47	1	0	0	0	0	0
1	32	34	41	27	24	21	22	1	0	0	0	0	0
1	18	19	16	8	23	1	0	0	0	0	0	0	0
1	63	75	46	73	67	1	0	0	0	0	0	0	0
1	14	28	31	29	37	1	0	0	0	0	0	0	0
1	99	9	12	10	7	5	1	0	0	0	0	0	0
1	26	25	11	65	58	1	0	0	0	0	0	0	0
1	62	56	69	1	0	0	0	0	0	0	0	0	0

Πίνακας 3: *Δρομολόγηση οχημάτων C103*

4. Αποτελέσματα προβλήματος C104

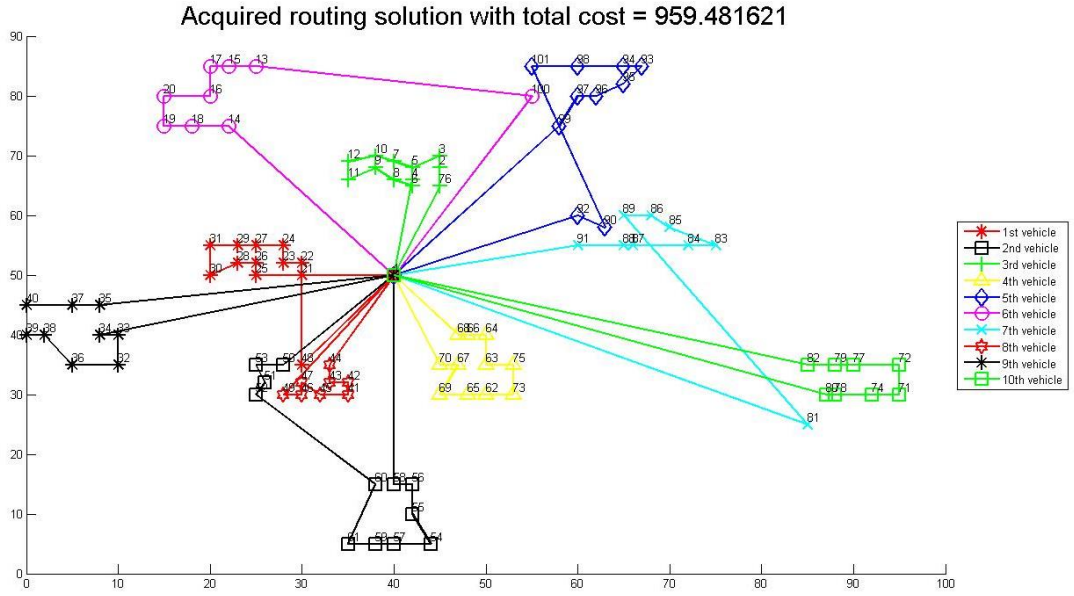


Σχήμα 4: *Δρομολόγηση οχημάτων C104*

ΠΕΛΑΤΕΣ ΑΝΑ ΟΧΗΜΑ													
1	43	44	42	41	45	47	46	49	52	51	53	48	1
1	68	66	64	63	75	73	62	65	69	67	70	50	1
1	4	6	8	9	11	12	10	7	90	92	2	76	1
1	25	21	26	28	30	31	29	27	24	23	22	1	0
1	99	97	96	95	93	94	98	101	100	3	1	0	0
1	14	18	19	20	16	17	15	13	5	1	0	0	0
1	91	88	87	84	83	85	86	89	81	1	0	0	0
1	58	56	55	54	57	59	61	60	1	0	0	0	0
1	33	34	32	36	38	39	40	37	35	1	0	0	0
1	82	79	77	72	71	74	78	80	1	0	0	0	0

Πίνακας 4: *Δρομολόγηση οχημάτων C104*

5. Αποτελέσματα προβλήματος C105

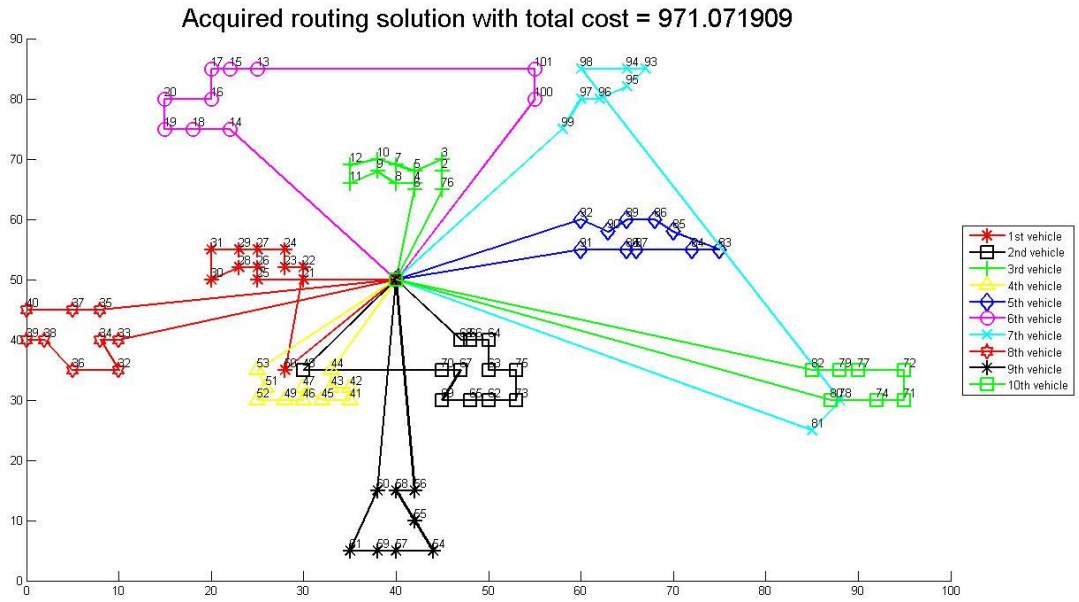


Σχήμα 5: *Δρομολόγηση οχημάτων C105*

ΠΕΛΑΤΕΣ ΑΝΑ ΟΧΗΜΑ													
1	6	4	8	9	11	12	10	7	5	3	2	76	1
1	44	43	42	41	75	73	62	65	69	67	70	48	1
1	21	25	26	28	30	31	29	27	24	23	22	50	1
1	66	68	64	63	45	47	46	49	52	51	53	1	0
1	99	97	96	95	93	94	98	101	90	92	1	0	0
1	14	18	19	20	16	17	15	13	100	1	0	0	0
1	91	88	87	84	83	85	86	89	81	1	0	0	0
1	58	56	55	54	57	59	61	60	1	0	0	0	0
1	33	34	32	36	38	39	40	37	35	1	0	0	0
1	82	79	77	72	71	74	78	80	1	0	0	0	0

Πίνακας 5: *Δρομολόγηση οχημάτων C105*

6. Αποτελέσματα προβλήματος C106

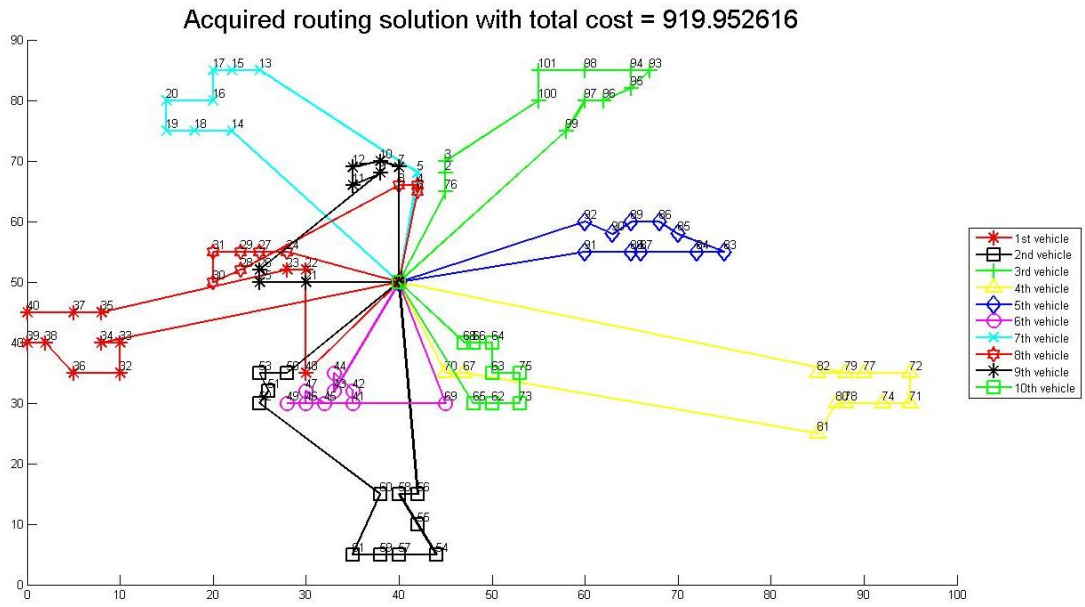


Σχήμα 6: *Δρομολόγηση οχημάτων C106*

ΠΕΛΑΤΕΣ ΑΝΑ ΟΧΗΜΑ												
1	21	25	26	28	30	31	29	27	24	23	22	50	1
1	68	66	64	63	75	73	62	65	69	67	70	48	1
1	6	4	8	9	11	12	10	7	5	3	2	76	1
1	44	43	42	41	45	47	46	49	52	51	53	1	0
1	91	88	87	83	84	85	86	89	90	92	1	0	0
1	14	18	19	20	16	17	15	13	101	100	1	0	0
1	99	97	96	95	93	94	98	78	81	1	0	0	0
1	33	34	32	36	38	39	40	37	35	1	0	0	0
1	56	58	55	54	57	59	61	60	1	0	0	0	0
1	82	79	77	72	71	74	80	1	0	0	0	0	0

Πίνακας 6: *Δρομολόγηση οχημάτων C106*

7. Αποτελέσματα προβλήματος C107

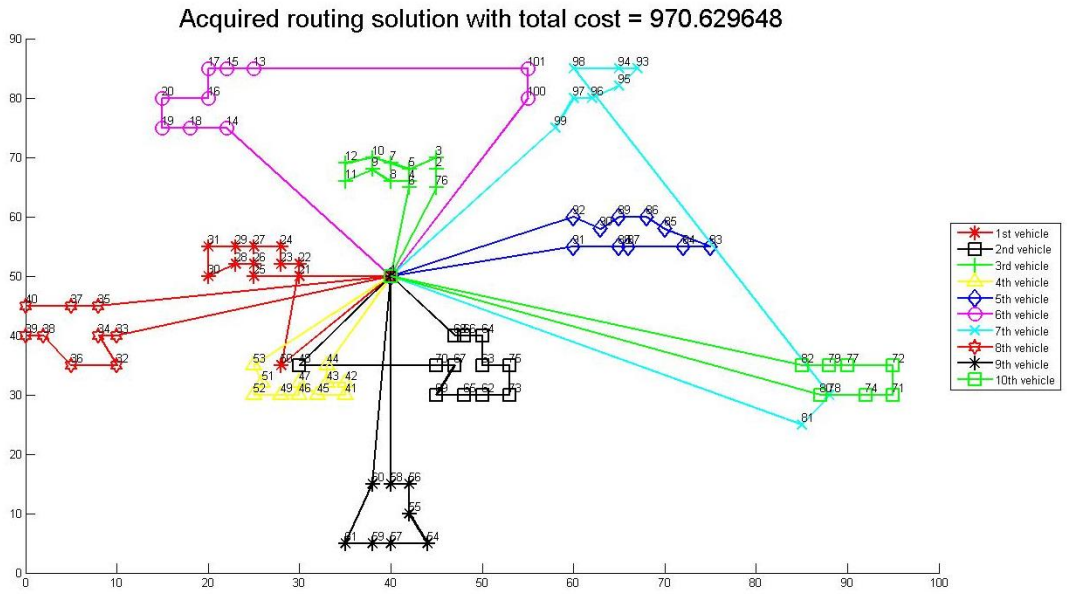


Σχήμα 7: *Δρομολόγηση οχημάτων C107*

ΠΕΛΑΤΕΣ ΑΝΑ ΟΧΗΜΑ													
1	6	43	55	54	57	59	61	60	69	67	70	48	1
1	91	4	8	9	11	12	10	7	5	3	2	76	1
1	66	44	42	41	45	47	46	49	52	51	53	50	1
1	33	34	32	36	38	39	40	37	35	23	22	1	0
1	58	88	87	84	83	85	86	89	90	92	1	0	0
1	82	79	77	72	71	74	78	80	81	1	0	0	0
1	21	25	26	28	30	31	29	27	24	1	0	0	0
1	97	99	96	95	93	94	98	101	100	1	0	0	0
1	18	14	19	20	16	17	15	13	1	0	0	0	0
1	56	68	64	63	75	73	62	65	1	0	0	0	0

Πίνακας 7: *Δρομολόγηση οχημάτων C107*

8. Αποτελέσματα προβλήματος C108

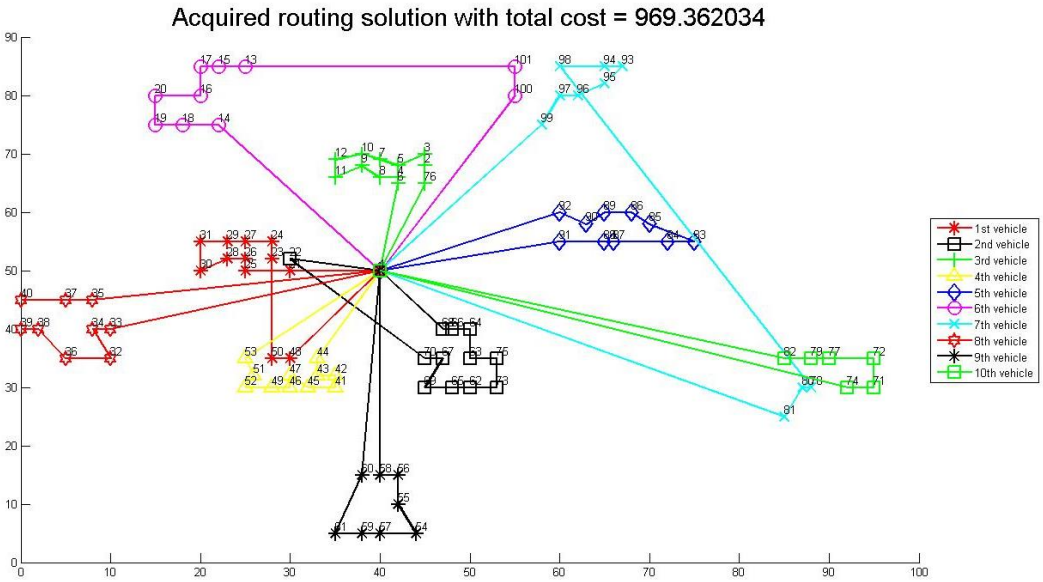


Σχήμα 8: *Δρομολόγηση οχημάτων C108*

ΠΕΛΑΤΕΣ ΑΝΑ ΟΧΗΜΑ												
1	21	25	26	28	30	31	29	27	24	23	22	50	1
1	68	66	64	63	75	73	62	65	69	67	70	48	1
1	6	4	8	9	11	12	10	7	5	3	2	76	1
1	44	43	42	41	45	47	46	49	52	51	53	1	0
1	91	88	87	84	83	85	86	89	90	92	1	0	0
1	14	18	19	20	16	17	15	13	101	100	1	0	0
1	99	97	96	95	93	94	98	78	81	1	0	0	0
1	33	34	32	36	38	39	40	37	35	1	0	0	0
1	56	58	55	54	57	59	61	60	1	0	0	0	0
1	82	79	77	72	71	74	80	1	0	0	0	0	0

Πίνακας 8: *Δρομολόγηση οχημάτων C108*

9. Αποτελέσματα προβλήματος C109



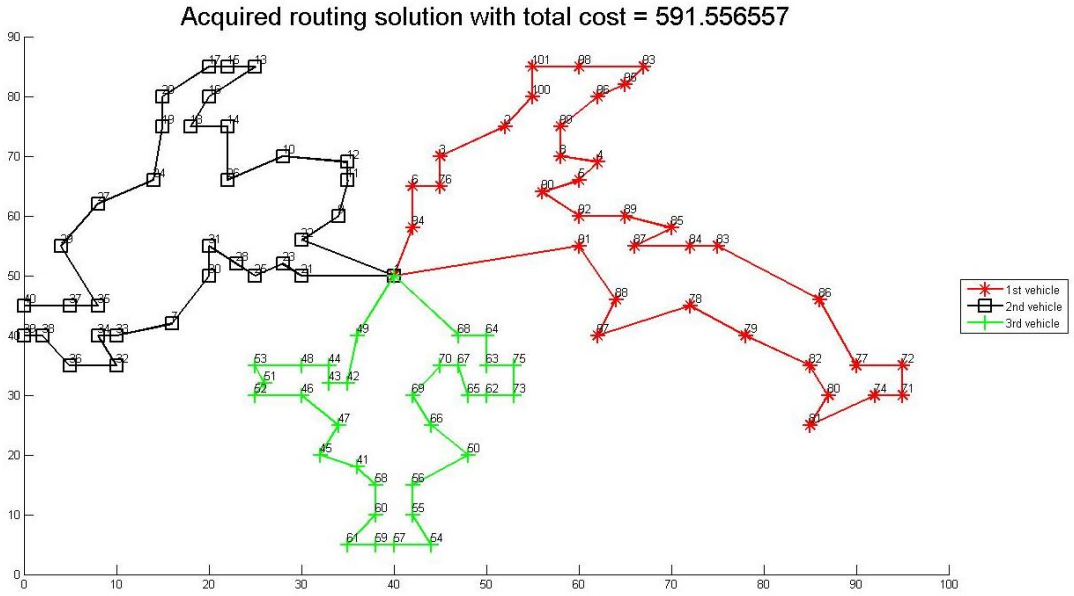
Σχήμα 9: *Δρομολόγηση οχημάτων C109*

ΠΕΛΑΤΕΣ ΑΝΑ ΟΧΗΜΑ													
1	21	25	26	28	30	31	29	27	24	23	50	48	1
1	68	66	64	63	75	73	62	65	69	70	67	22	1
1	6	4	8	9	11	12	10	7	5	3	2	76	1
1	44	42	41	43	45	46	47	49	52	51	53	1	0
1	91	88	87	84	83	85	86	89	90	92	1	0	0
1	14	18	19	20	16	17	15	13	101	100	1	0	0
1	99	97	96	95	93	94	98	78	80	81	1	0	0
1	33	34	32	36	38	39	40	37	35	1	0	0	0
1	56	58	55	54	57	59	61	60	1	0	0	0	0
1	82	79	77	72	71	74	1	0	0	0	0	0	0

Πίνακας 9: *Δρομολόγηση οχημάτων C109*

Προβλήματα κατηγορίας C2

10. Αποτελέσματα προβλήματος C201

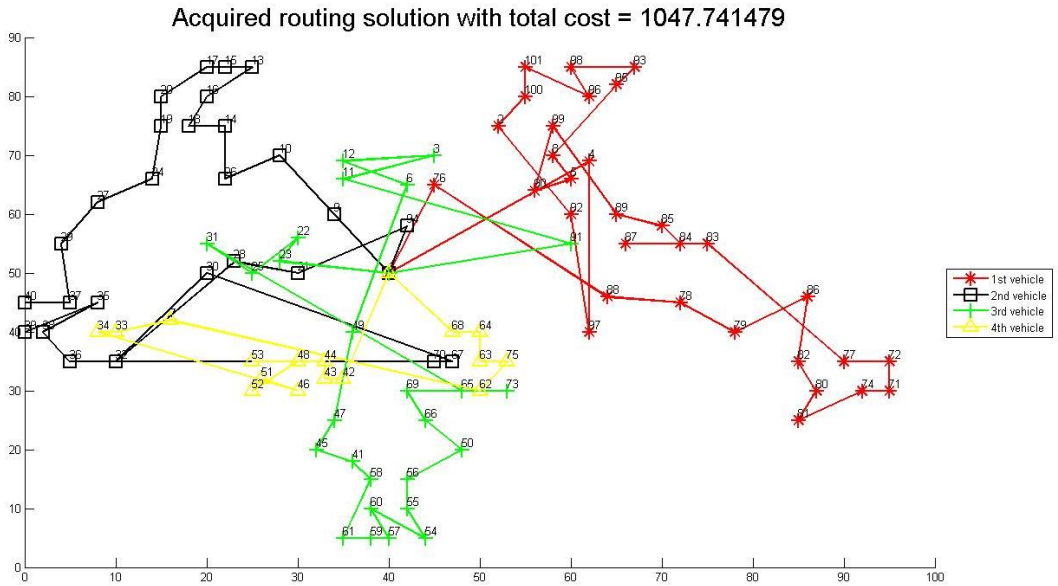


Σχήμα 10: *Δρομολόγηση οχημάτων C201*

ΠΕΛΑΤΕΣ ΑΝΑ ΟΧΗΜΑ												
1	94	6	76	3	2	100	101	98	93	95	96	99
8	4	5	90	92	89	85	87	84	83	86	77	72
71	74	81	80	82	79	78	97	88	91	1		
1	21	23	25	28	31	30	7	33	34	32	36	38
39	40	37	35	29	27	24	19	20	17	15	13	16
18	14	26	10	12	11	9	22	1	0	0		
1	68	64	63	75	73	62	65	67	70	69	66	50
56	55	54	57	59	61	60	58	41	45	47	46	52
51	53	48	44	43	42	49	1	0	0	0		

Πίνακας 10: *Δρομολόγηση οχημάτων C201*

11. Αποτελέσματα προβλήματος C202

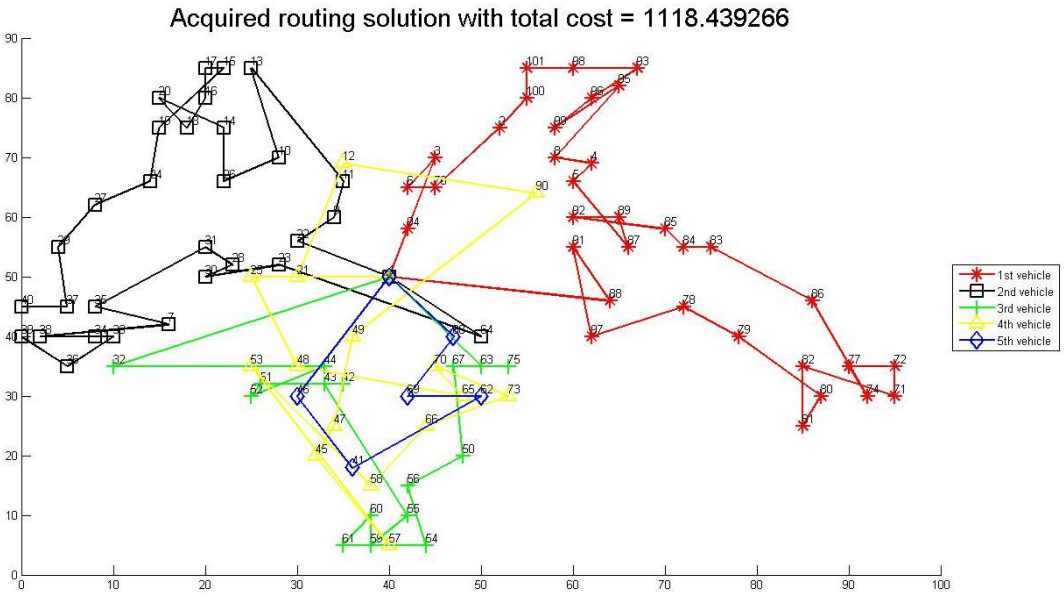


Σχήμα 11: *Δρομολόγηση οχημάτων C202*

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

Πίνακας 11: *Δρομολόγηση οχημάτων C202*

12. Αποτελέσματα προβλήματος C203

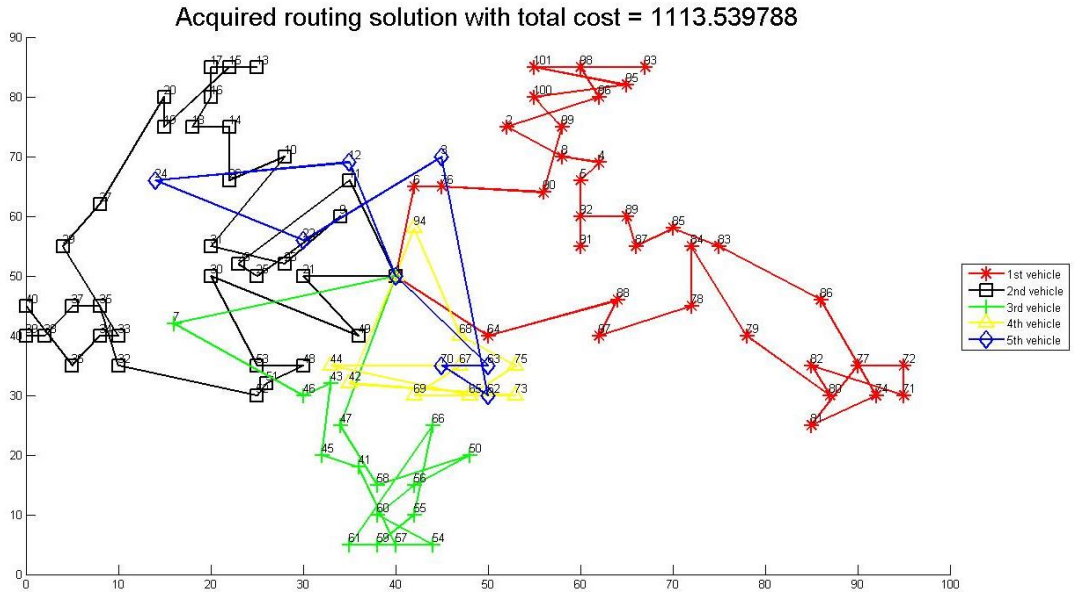


Σχήμα 12: *Δρομολόγηση οχημάτων C203*

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

Πίνακας 12: *Δρομολόγηση οχημάτων C203*

13. Αποτελέσματα προβλήματος C204

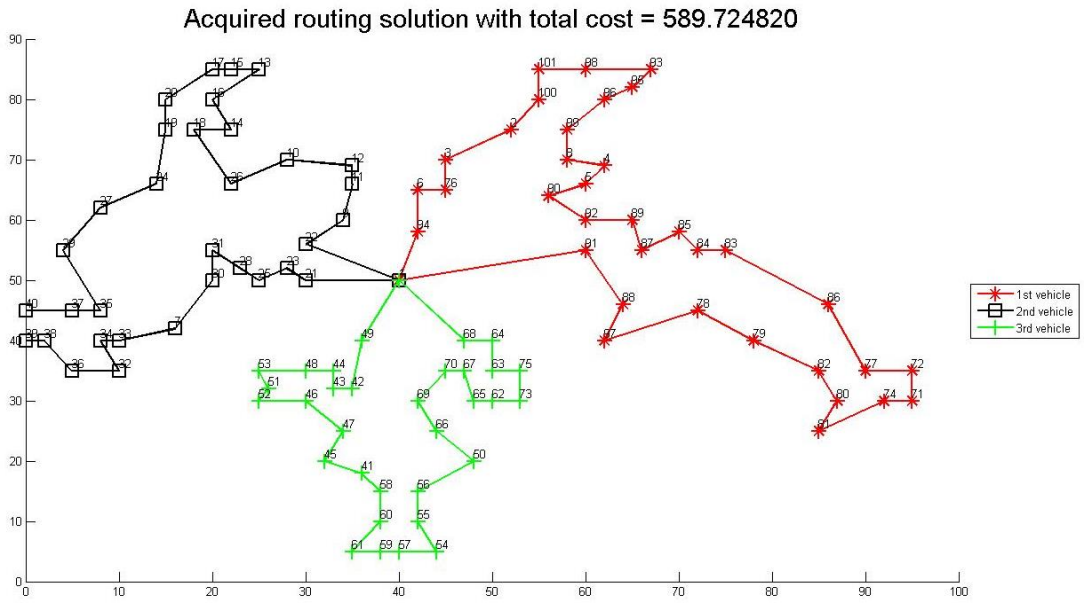


Σχήμα 13: *Δρομολόγηση οχημάτων C204*

ΠΕΛΑΤΕΣ ΑΝΑ ΟΧΗΜΑ											
1	6	76	90	99	100	95	101	93	98	96	2
8	4	5	91	92	89	87	85	83	86	74	81
77	72	71	82	80	79	84	78	97	88	64	1
1	21	49	30	53	48	51	52	32	35	37	38
39	40	36	34	33	29	27	20	19	15	13	17
16	18	14	26	10	31	23	9	25	28	11	1
1	47	58	50	56	60	54	59	55	66	61	57
41	45	43	46	7	1	0	0	0	0	0	0
1	94	68	75	65	44	67	69	73	42	1	0
1	63	70	62	3	22	24	12	1	0	0	0

Πίνακας 13: *Δρομολόγηση οχημάτων C204*

14. Αποτελέσματα προβλήματος C205

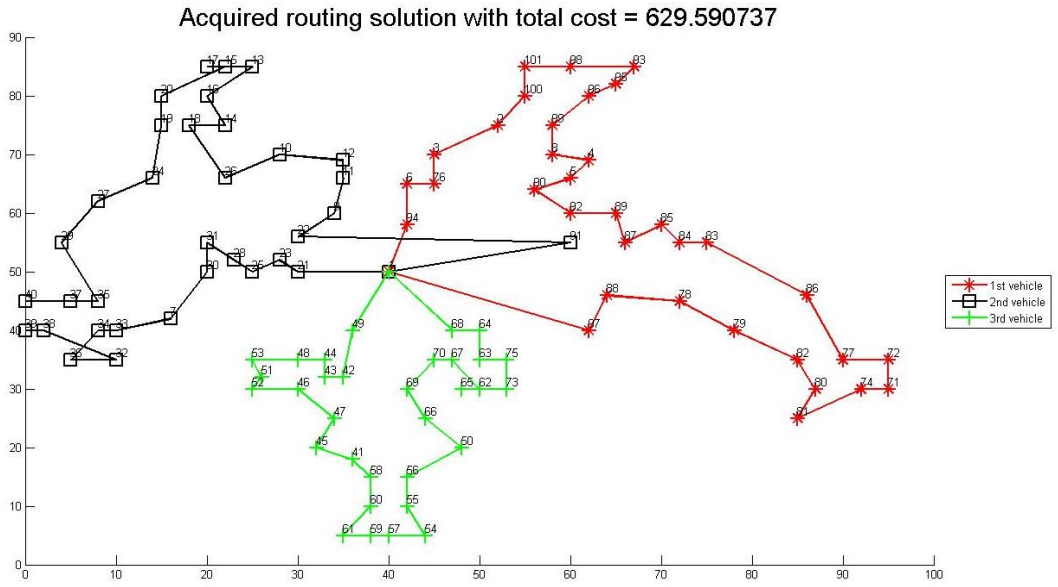


Σχήμα 14: *Δρομολόγηση οχημάτων C205*

ΠΕΛΑΤΕΣ ΑΝΑ ΟΧΗΜΑ											
1	94	6	76	3	2	100	101	98	93	95	96
99	8	4	5	90	92	89	87	85	84	83	86
77	72	71	74	81	80	82	79	78	97	88	91
1	0	0	0	0	0	0	0	0	0	0	0
1	21	23	25	28	31	30	7	33	34	32	36
38	39	40	37	35	29	27	24	19	20	17	15
13	16	14	18	26	10	12	11	9	22	1	0
1	68	64	63	75	73	62	65	67	70	69	66
50	56	55	54	57	59	61	60	58	41	45	47
46	52	51	53	48	44	43	42	49	1	0	0

Πίνακας 14: *Δρομολόγηση οχημάτων C205*

15. Αποτελέσματα προβλήματος C206

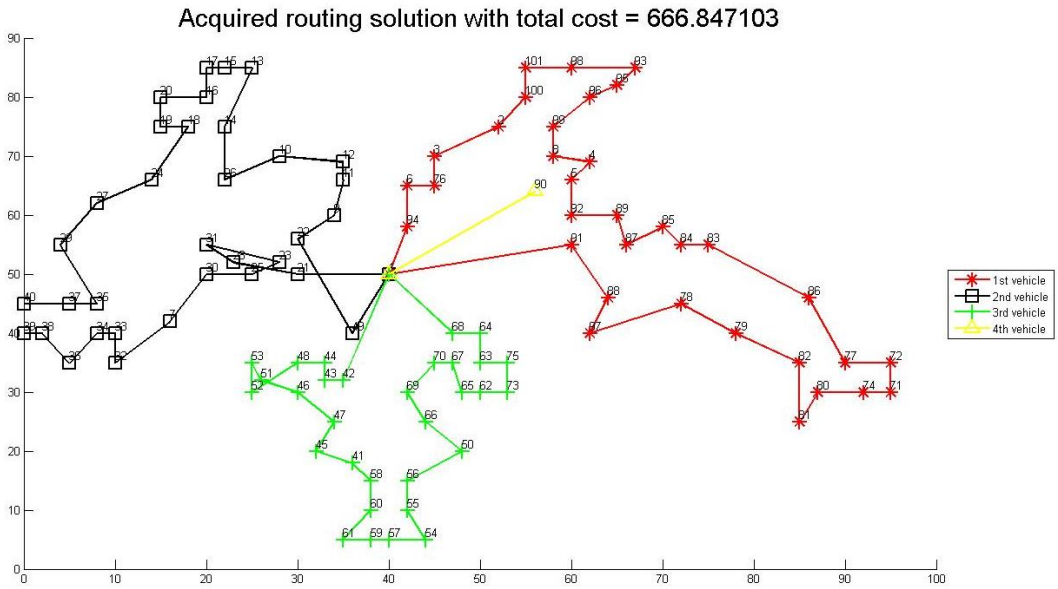


Σχήμα 15: *Δρομολόγηση οχημάτων C206*

ΠΕΛΑΤΕΣ ΑΝΑ ΟΧΗΜΑ											
1	94	6	76	3	2	100	101	98	93	95	96
99	8	4	5	90	92	89	87	85	84	83	86
77	72	71	74	81	80	82	79	78	88	97	1
1	21	23	25	28	31	30	7	33	34	36	32
38	39	40	37	35	29	27	24	19	20	15	17
13	16	14	18	26	10	12	11	9	22	91	1
1	68	64	63	75	73	65	62	67	70	69	66
50	56	55	54	57	59	61	60	58	41	45	47
46	52	51	53	48	44	43	42	49	1	0	0

Πίνακας 15: *Δρομολόγηση οχημάτων C206*

16. Αποτελέσματα προβλήματος C207

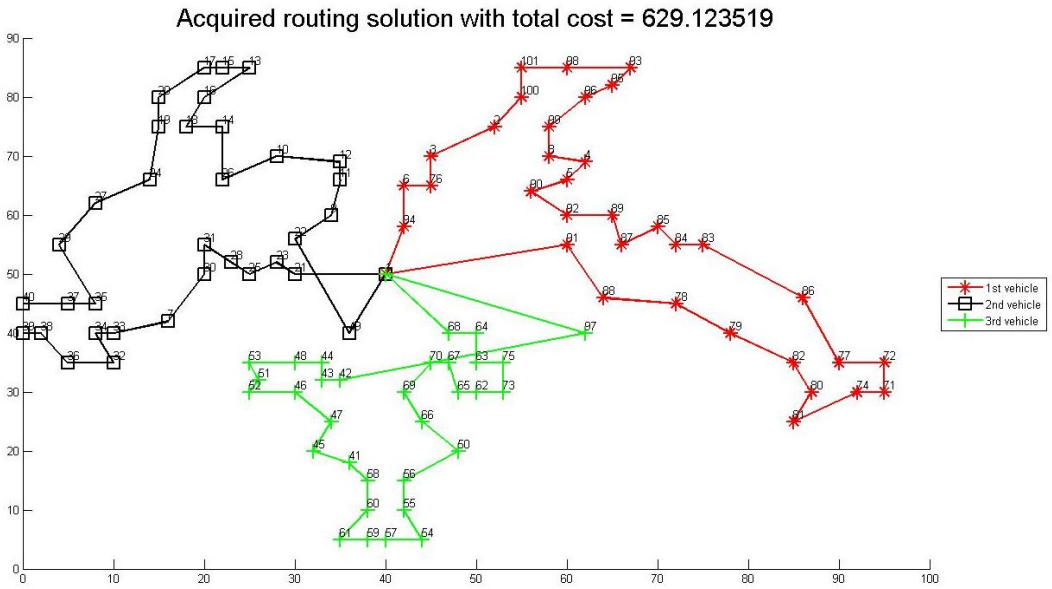


Σχήμα 16: *Δρομολόγηση οχημάτων C207*

ΠΕΛΑΤΕΣ ΑΝΑ ΟΧΗΜΑ											
1	94	6	76	3	2	100	101	98	93	95	96
99	8	4	5	92	89	87	85	84	83	86	77
72	71	74	80	81	82	79	78	97	88	91	1
1	21	28	31	23	25	30	7	32	33	34	36
38	39	40	37	35	29	27	24	18	19	20	16
17	15	13	14	26	10	12	11	9	22	49	1
1	68	64	63	75	73	62	65	67	70	69	66
50	56	55	54	59	57	61	60	58	41	45	47
46	51	53	52	48	44	43	42	1	0	0	0
1	90	1	0	0	0	0	0	0	0	0	0

Πίνακας 16: *Δρομολόγηση οχημάτων C207*

17. Αποτελέσματα προβλήματος C208



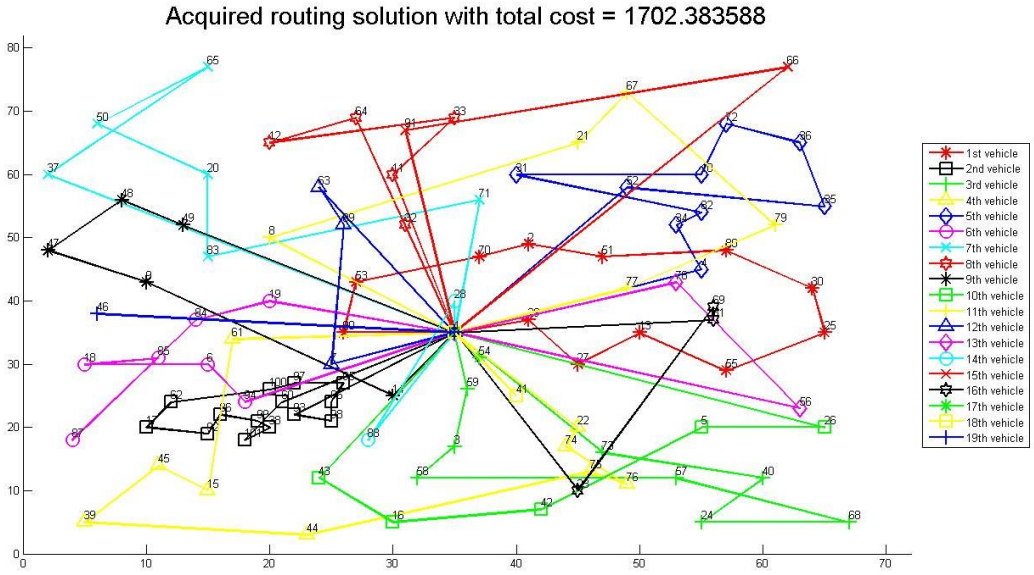
Σχήμα 17: *Δρομολόγηση οχημάτων C208*

ΠΕΛΑΤΕΣ ΑΝΑ ΟΧΗΜΑ											
1	94	6	76	3	2	100	101	98	93	95	96
99	8	4	5	90	92	89	87	85	84	83	86
77	72	71	74	81	80	82	79	78	88	91	1
1	21	23	25	28	31	30	7	33	34	32	36
38	39	40	37	35	29	27	24	19	20	17	15
13	16	18	14	26	10	12	11	9	22	49	1
1	68	64	63	75	73	62	65	67	70	69	66
50	56	55	54	57	59	61	60	58	41	45	47
46	52	51	53	48	44	43	42	97	1	0	0

Πίνακας 17: *Δρομολόγηση οχημάτων C208*

Προβλήματα κατηγορίας R1

18. Αποτελέσματα προβλήματος R101

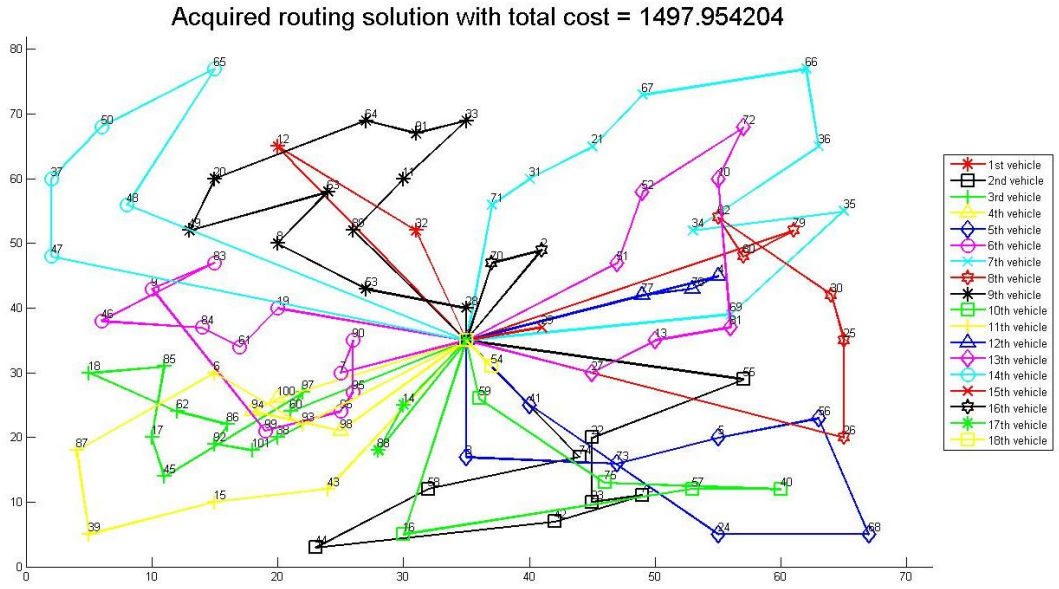


Σχήμα 18: *Δρομολόγηση οχημάτων R101*

ΠΕΛΑΤΕΣ ΑΝΑ ΟΧΗΜΑ															
1	90	53	70	2	51	80	4	25	55	13	27	29	1	0	0
1	95	60	99	38	97	100	86	92	17	62	6	61	84	19	1
1	22	40	24	68	56	5	76	74	1	0	0	0	0	0	0
1	93	15	87	39	44	58	14	98	94	1	0	0	0	0	0
1	82	34	31	52	10	91	33	64	32	1	0	0	0	0	0
1	46	48	49	9	96	18	101	1	0	0	0	0	0	0	0
1	28	77	30	35	69	21	1	0	0	0	0	0	0	0	0
1	8	63	12	67	81	1	0	0	0	0	0	0	0	0	0
1	37	65	50	20	47	1	0	0	0	0	0	0	0	0	0
1	3	16	23	73	57	1	0	0	0	0	0	0	0	0	0
1	66	79	72	36	78	1	0	0	0	0	0	0	0	0	0
1	54	75	26	1	0	0	0	0	0	0	0	0	0	0	0
1	89	11	1	0	0	0	0	0	0	0	0	0	0	0	0
1	43	88	1	0	0	0	0	0	0	0	0	0	0	0	0
1	42	59	1	0	0	0	0	0	0	0	0	0	0	0	0
1	83	45	85	1	0	0	0	0	0	0	0	0	0	0	0
1	71	1	0	0	0	0	0	0	0	0	0	0	0	0	0
1	7	1	0	0	0	0	0	0	0	0	0	0	0	0	0
1	41	1	0	0	0	0	0	0	0	0	0	0	0	0	0

Πίνακας 18: *Δρομολόγηση οχημάτων R101*

19. Αποτελέσματα προβλήματος R102

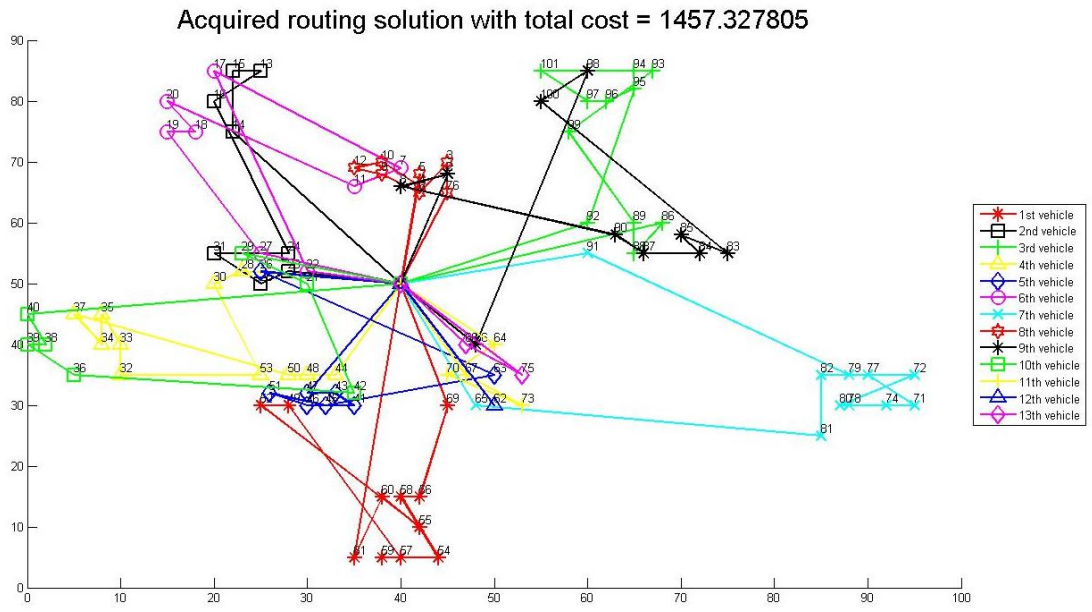


Σχήμα 19: *Δρομολόγηση οχημάτων R102*

ΠΕΛΑΤΕΣ ΑΝΑ ΟΧΗΜΑ												
1	95	85	1	0	0	0	0	0	0	0	0	0
1	97	16	44	42	23	75	76	41	1	0	0	0
1	90	38	17	45	39	87	86	99	98	101	60	1
1	55	56	26	1	0	0	0	0	0	0	0	0
1	57	73	24	68	40	5	74	1	0	0	0	0
1	7	96	6	94	62	15	92	18	46	84	61	1
1	4	79	35	82	80	34	30	81	69	1	0	0
1	70	31	21	52	78	27	1	0	0	0	0	0
1	8	49	20	12	11	64	91	33	71	32	29	1
1	19	48	9	83	47	1	0	0	0	0	0	0
1	93	88	43	3	14	28	1	0	0	0	0	0
1	77	22	59	1	0	0	0	0	0	0	0	0
1	67	66	36	10	72	51	2	1	0	0	0	0
1	89	63	65	37	50	1	0	0	0	0	0	0
1	54	1	0	0	0	0	0	0	0	0	0	0
1	100	58	1	0	0	0	0	0	0	0	0	0
1	13	25	1	0	0	0	0	0	0	0	0	0
1	53	1	0	0	0	0	0	0	0	0	0	0

Πίνακας 19: *Δρομολόγηση οχημάτων R102*

20. Αποτελέσματα προβλήματος R103

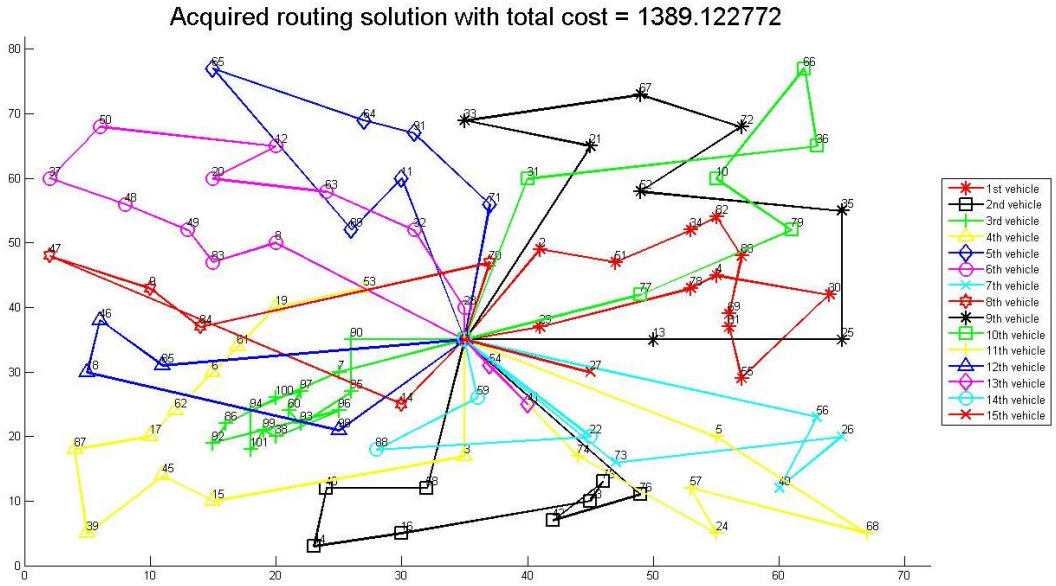


Σχήμα 20: *Δρομολόγηση οχημάτων R103*

ΠΕΛΑΤΕΣ ΑΝΑ ΟΧΗΜΑ												
1	28	21	3	98	93	94	96	101	100	97	8	1
1	23	26	22	25	10	31	24	27	1	0	0	0
1	84	87	77	71	72	74	78	80	82	79	81	1
1	99	95	9	11	6	4	83	91	7	76	2	1
1	86	88	85	90	92	89	65	64	1	0	0	0
1	14	15	16	13	17	20	19	18	1	0	0	0
1	33	34	32	38	35	39	40	30	29	70	66	1
1	73	62	45	41	42	43	36	37	1	0	0	0
1	57	60	55	54	59	61	53	44	51	47	1	0
1	48	56	58	49	50	52	46	1	0	0	0	0
1	12	5	67	69	1	0	0	0	0	0	0	0
1	68	1	0	0	0	0	0	0	0	0	0	0
1	63	75	1	0	0	0	0	0	0	0	0	0

Πίνακας 20: *Δρομολόγηση οχημάτων R103*

21. Αποτελέσματα προβλήματος R104

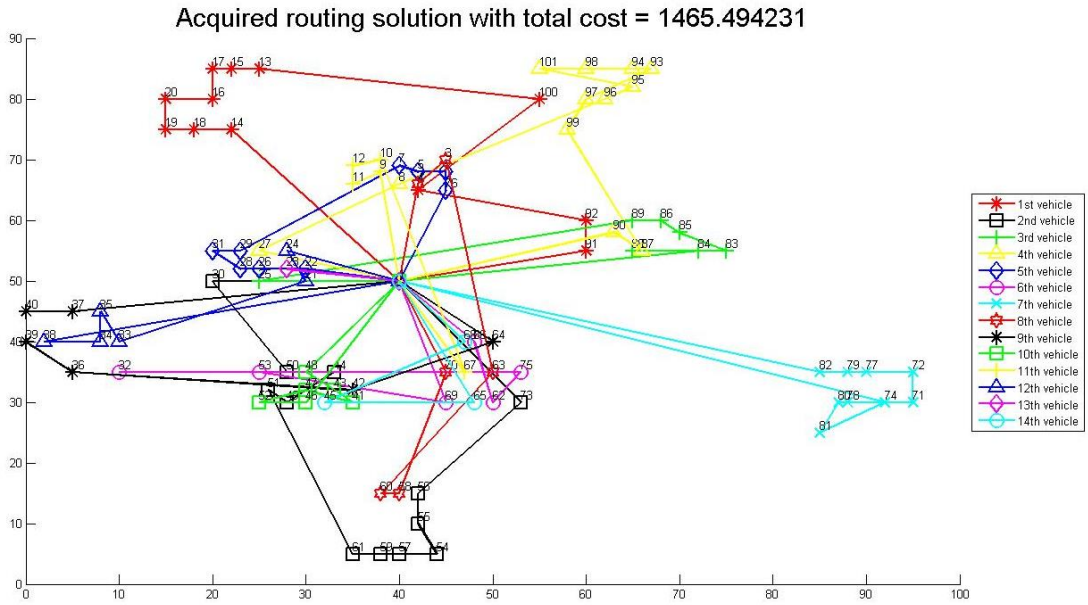


Σχήμα 21: *Δρομολόγηση οχημάτων R104*

ΠΕΛΑΤΕΣ ΑΝΑ ΟΧΗΜΑ															
1	29	13	81	69	77	80	79	4	82	34	51	28	1	0	0
1	43	44	16	88	58	75	42	74	1	0	0	0	0	0	0
1	94	60	93	99	38	86	100	97	7	95	14	96	98	59	1
1	90	61	6	62	45	39	15	17	87	92	101	1	0	0	0
1	63	33	64	65	91	71	1	0	0	0	0	0	0	0	0
1	53	32	11	12	20	49	48	47	46	84	19	1	0	0	0
1	54	55	5	78	1	0	0	0	0	0	0	0	0	0	0
1	83	9	37	50	8	1	0	0	0	0	0	0	0	0	0
1	52	67	36	66	72	31	21	89	1	0	0	0	0	0	0
1	41	76	73	23	85	18	1	0	0	0	0	0	0	0	0
1	3	22	57	24	56	1	0	0	0	0	0	0	0	0	0
1	68	40	26	25	1	0	0	0	0	0	0	0	0	0	0
1	70	2	1	0	0	0	0	0	0	0	0	0	0	0	0
1	10	35	30	1	0	0	0	0	0	0	0	0	0	0	0
1	27	1	0	0	0	0	0	0	0	0	0	0	0	0	0

Πίνακας 21: *Δρομολόγηση οχημάτων R104*

22. Αποτελέσματα προβλήματος R105

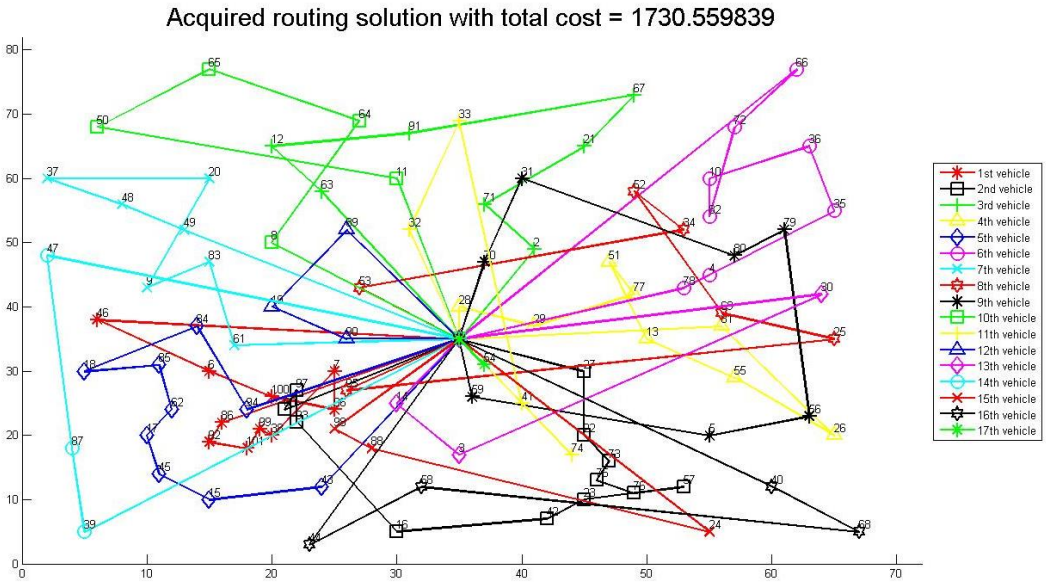


Σχήμα 22: *Δρομολόγηση οχημάτων R105*

ΠΕΛΑΤΕΣ ΑΝΑ ΟΧΗΜΑ													
1	14	18	19	20	16	15	17	13	6	91	90	64	1
1	60	56	55	54	57	59	61	58	49	53	33	22	1
1	66	75	65	62	67	70	27	1	0	0	0	0	0
1	92	88	86	96	93	94	98	101	100	95	97	99	1
1	51	43	41	25	31	29	30	50	21	1	0	0	0
1	11	12	10	76	23	3	1	0	0	0	0	0	0
1	87	84	77	71	74	78	80	81	72	1	0	0	0
1	63	48	46	44	24	47	1	0	0	0	0	0	0
1	52	34	36	32	39	38	1	0	0	0	0	0	0
1	68	42	28	35	40	37	1	0	0	0	0	0	0
1	83	79	82	85	89	1	0	0	0	0	0	0	0
1	26	4	2	7	5	8	1	0	0	0	0	0	0
1	45	1	0	0	0	0	0	0	0	0	0	0	0
1	9	73	69	1	0	0	0	0	0	0	0	0	0

Πίνακας 22: *Δρομολόγηση οχημάτων R105*

23. Αποτελέσματα προβλήματος R106

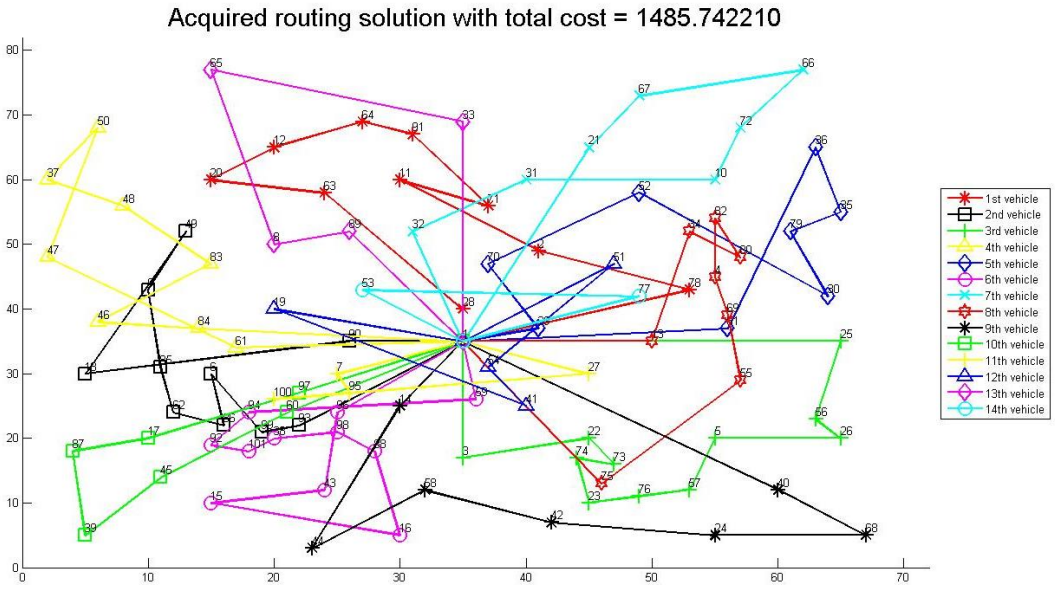


Σχήμα 23: *Δρομολόγηση οχημάτων R106*

ΠΕΛΑΤΕΣ ΑΝΑ ΟΧΗΜΑ													
1	46	6	100	96	7	38	99	101	92	86	1	0	0
1	60	97	93	16	42	23	57	76	75	73	22	27	1
1	63	12	91	67	21	71	2	1	0	0	0	0	0
1	28	29	77	51	13	55	26	81	1	0	0	0	0
1	43	15	45	17	62	85	18	84	94	1	0	0	0
1	66	72	82	10	36	35	4	78	1	0	0	0	0
1	49	48	37	20	9	83	61	1	0	0	0	0	0
1	53	34	52	69	25	95	1	0	0	0	0	0	0
1	70	31	80	79	56	5	59	1	0	0	0	0	0
1	8	64	65	50	11	1	0	0	0	0	0	0	0
1	74	41	33	32	1	0	0	0	0	0	0	0	0
1	89	19	90	1	0	0	0	0	0	0	0	0	0
1	30	3	14	1	0	0	0	0	0	0	0	0	0
1	39	87	47	1	0	0	0	0	0	0	0	0	0
1	24	88	98	1	0	0	0	0	0	0	0	0	0
1	40	68	58	44	1	0	0	0	0	0	0	0	0
1	54	1	0	0	0	0	0	0	0	0	0	0	0

Πίνακας 23: *Δρομολόγηση οχημάτων R106*

24. Αποτελέσματα προβλήματος R107

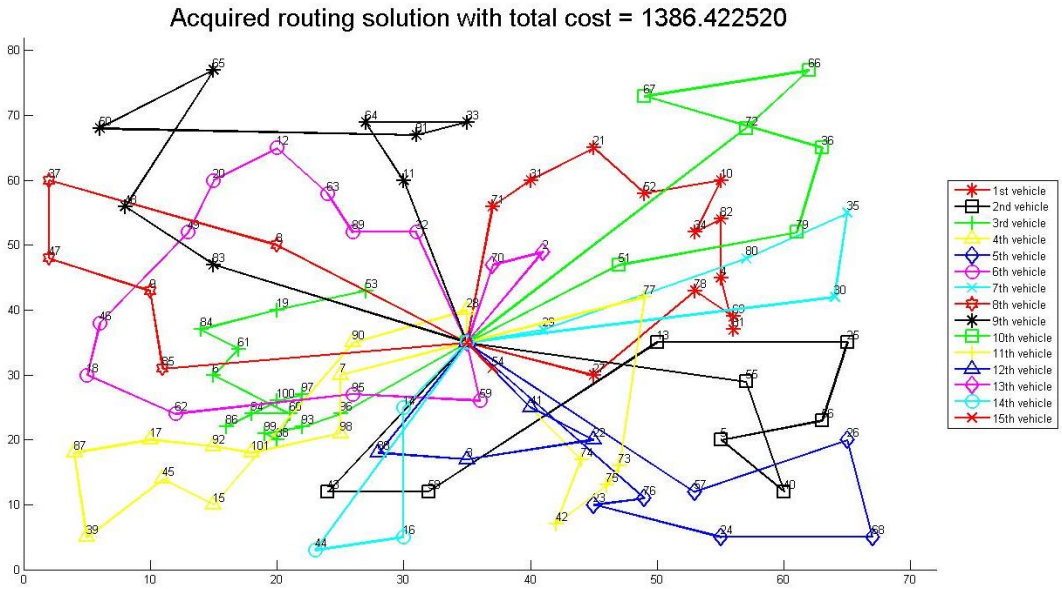


Σχήμα 24: *Δρομολόγηση οχημάτων R107*

ΠΕΛΑΤΕΣ ΑΝΑ ΟΧΗΜΑ												
1	28	63	20	12	64	91	71	11	2	78	1	0
1	93	99	6	86	62	85	9	49	18	90	1	0
1	3	22	73	74	23	76	57	5	26	56	25	1
1	84	46	83	48	37	50	47	61	1	0	0	0
1	29	70	52	30	79	35	36	81	1	0	0	0
1	96	43	15	16	88	98	38	101	92	94	59	1
1	32	31	10	72	66	67	21	1	0	0	0	0
1	13	34	80	82	4	69	55	75	1	0	0	0
1	40	68	24	42	58	44	14	1	0	0	0	0
1	60	45	39	87	17	97	1	0	0	0	0	0
1	7	95	100	27	1	0	0	0	0	0	0	0
1	19	41	54	51	1	0	0	0	0	0	0	0
1	89	8	65	33	1	0	0	0	0	0	0	0
1	77	53	1	0	0	0	0	0	0	0	0	0

Πίνακας 24: *Δρομολόγηση οχημάτων R107*

25. Αποτελέσματα προβλήματος R108

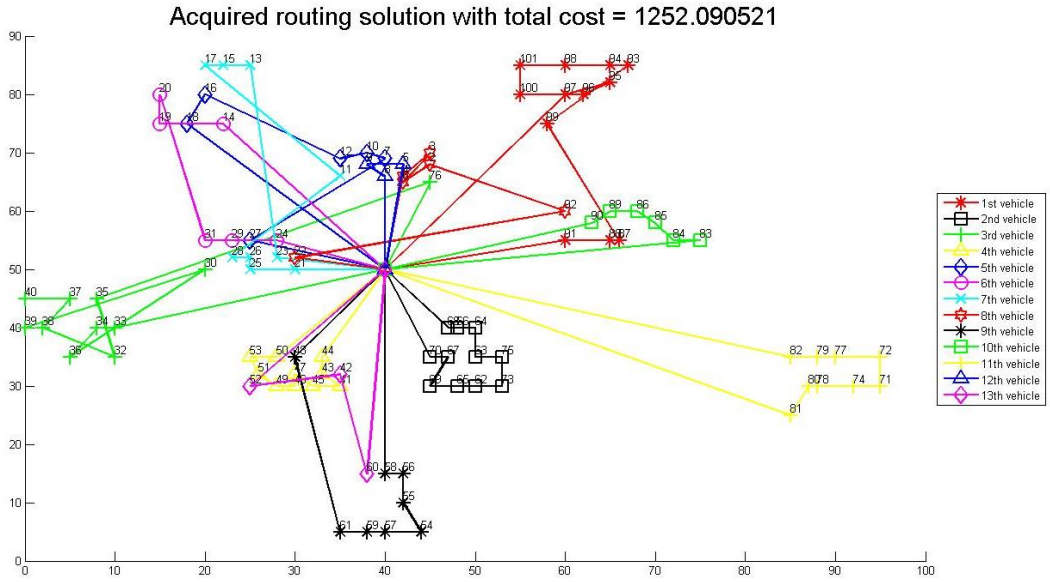


Σχήμα 25: *Δρομολόγηση οχημάτων R108*

ΠΕΛΑΤΕΣ ΑΝΑ ΟΧΗΜΑ															
1	71	31	21	52	10	34	82	4	81	69	78	27	1	0	0
1	43	58	13	25	56	5	40	55	1	0	0	0	0	0	0
1	96	93	99	38	97	100	86	94	60	6	61	84	19	53	1
1	28	90	15	45	39	87	17	92	101	98	7	1	0	0	0
1	76	23	24	68	26	57	1	0	0	0	0	0	0	0	0
1	32	89	63	12	20	49	46	18	62	95	59	1	0	0	0
1	30	35	80	29	1	0	0	0	0	0	0	0	0	0	0
1	85	9	47	37	8	1	0	0	0	0	0	0	0	0	0
1	11	64	33	91	50	65	48	83	1	0	0	0	0	0	0
1	72	66	67	36	79	51	1	0	0	0	0	0	0	0	0
1	77	73	75	42	74	1	0	0	0	0	0	0	0	0	0
1	88	3	22	41	1	0	0	0	0	0	0	0	0	0	0
1	2	70	1	0	0	0	0	0	0	0	0	0	0	0	0
1	44	16	14	1	0	0	0	0	0	0	0	0	0	0	0
1	54	1	0	0	0	0	0	0	0	0	0	0	0	0	0

Πίνακας 25: *Δρομολόγηση οχημάτων R108*

26. Αποτελέσματα προβλήματος R109

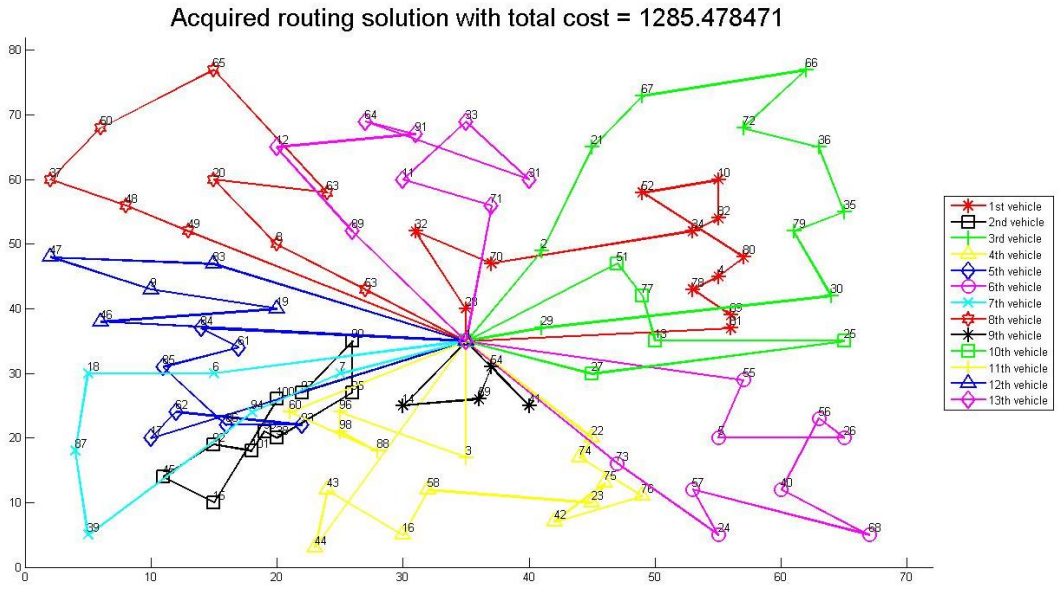


Σχήμα 26: *Δρομολόγηση οχημάτων R109*

ΠΕΛΑΤΕΣ ΑΝΑ ΟΧΗΜΑ													
1	91	88	87	95	93	94	98	101	100	97	96	99	1
1	68	66	64	63	75	73	62	65	69	70	50	1	0
1	33	34	36	30	39	40	37	38	32	49	48	1	0
1	44	43	42	45	47	46	52	35	51	53	1	0	0
1	21	9	8	10	90	2	1	0	0	0	0	0	0
1	76	4	31	92	89	86	1	0	0	0	0	0	0
1	14	18	19	20	16	17	15	13	3	1	0	0	0
1	25	28	29	27	24	22	1	0	0	0	0	0	0
1	56	55	54	57	59	61	60	67	1	0	0	0	0
1	26	11	12	6	7	5	1	0	0	0	0	0	0
1	82	79	77	72	71	74	78	80	81	1	0	0	0
1	84	85	83	1	0	0	0	0	0	0	0	0	0
1	58	41	23	1	0	0	0	0	0	0	0	0	0

Πίνακας 26: *Δρομολόγηση οχημάτων R109*

27. Αποτελέσματα προβλήματος R110

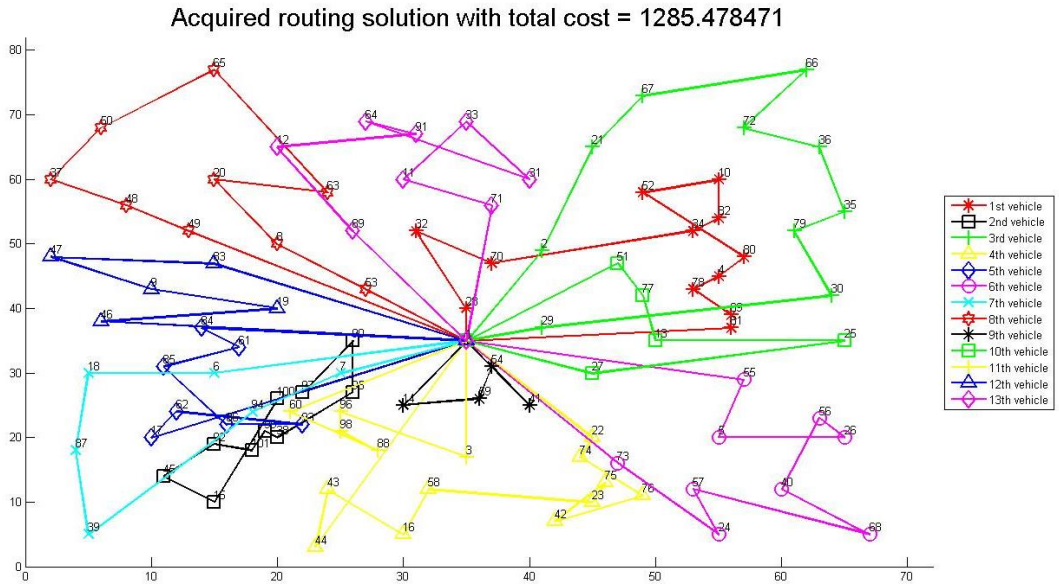


Σχήμα 27: *Δρομολόγηση οχημάτων R109*

ΠΕΛΑΤΕΣ ΑΝΑ ΟΧΗΜΑ													
1	28	32	70	34	82	10	52	80	4	78	69	81	1
1	100	15	45	92	101	99	38	95	90	97	1	0	0
1	29	30	79	35	36	72	66	67	21	2	1	0	0
1	22	74	76	42	75	23	58	16	43	44	1	0	0
1	17	62	93	86	85	61	84	1	0	0	0	0	0
1	73	24	57	68	40	56	26	5	55	1	0	0	0
1	7	94	39	87	18	6	1	0	0	0	0	0	0
1	53	8	20	63	65	50	37	48	49	1	0	0	0
1	14	59	54	41	1	0	0	0	0	0	0	0	0
1	51	77	13	25	27	1	0	0	0	0	0	0	0
1	60	88	98	96	3	1	0	0	0	0	0	0	0
1	83	47	9	19	46	1	0	0	0	0	0	0	0
1	89	12	91	64	31	33	11	71	1	0	0	0	0

Πίνακας 27: *Δρομολόγηση οχημάτων R110*

28. Αποτελέσματα προβλήματος R111

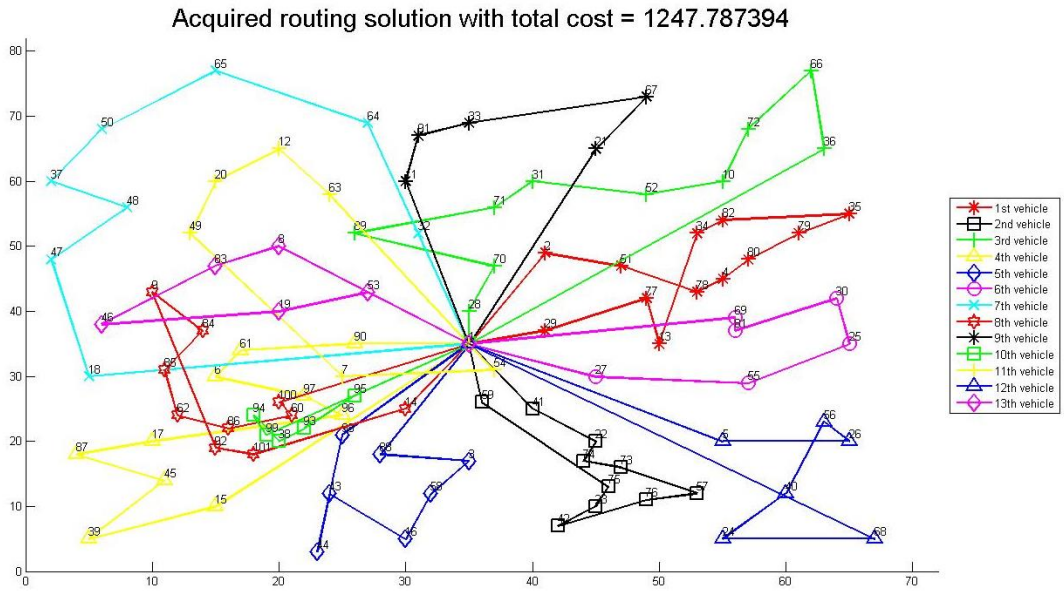


Σχήμα 28: *Δρομολόγηση οχημάτων R111*

ΠΕΛΑΤΕΣ ΑΝΑ ΟΧΗΜΑ													
1	28	32	70	34	82	10	52	80	4	78	69	81	1
1	100	15	45	92	101	99	38	95	90	97	1	0	0
1	29	30	79	35	36	72	66	67	21	2	1	0	0
1	22	74	76	42	75	23	58	16	43	44	1	0	0
1	17	62	93	86	85	61	84	1	0	0	0	0	0
1	73	24	57	68	40	56	26	5	55	1	0	0	0
1	7	94	39	87	18	6	1	0	0	0	0	0	0
1	53	8	20	63	65	50	37	48	49	1	0	0	0
1	14	59	54	41	1	0	0	0	0	0	0	0	0
1	51	77	13	25	27	1	0	0	0	0	0	0	0
1	60	88	98	96	3	1	0	0	0	0	0	0	0
1	83	47	9	19	46	1	0	0	0	0	0	0	0
1	89	12	91	64	31	33	11	71	1	0	0	0	0

Πίνακας 28: *Δρομολόγηση οχημάτων R111*

29. Αποτελέσματα προβλήματος R112



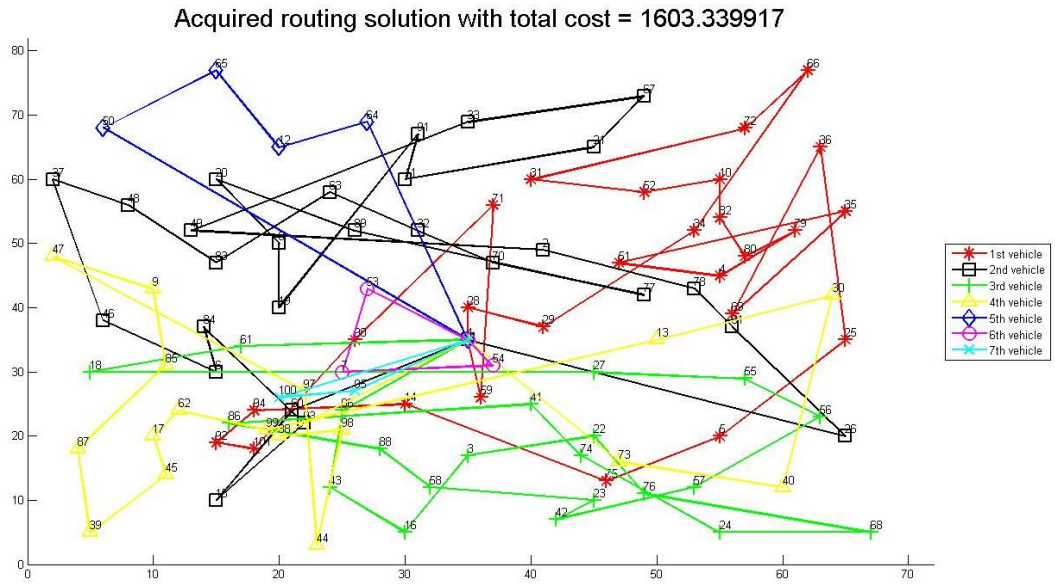
Σχήμα29: *Δρομολόγηση οχημάτων R112*

ΠΕΛΑΤΕΣ ΑΝΑ ΟΧΗΜΑ													
1	29	77	13	34	82	35	79	80	4	78	51	2	1
1	41	22	74	73	57	76	42	23	75	59	1	0	0
1	28	70	89	71	31	52	10	72	66	36	1	0	0
1	15	39	45	87	17	96	97	6	61	90	1	0	0
1	88	3	58	16	43	44	98	1	0	0	0	0	0
1	69	81	30	25	55	27	1	0	0	0	0	0	0
1	32	64	65	50	37	48	47	18	1	0	0	0	0
1	100	60	86	62	85	84	9	92	101	14	1	0	0
1	11	91	33	67	21	1	0	0	0	0	0	0	0
1	99	94	38	93	95	1	0	0	0	0	0	0	0
1	63	12	20	49	7	54	1	0	0	0	0	0	0
1	68	24	40	56	26	5	1	0	0	0	0	0	0
1	53	19	46	83	8	1	0	0	0	0	0	0	0

Πίνακας 29: *Δρομολόγηση οχημάτων R112*

Προβλήματα κατηγορίας R2

30. Αποτελέσματα προβλήματος R201

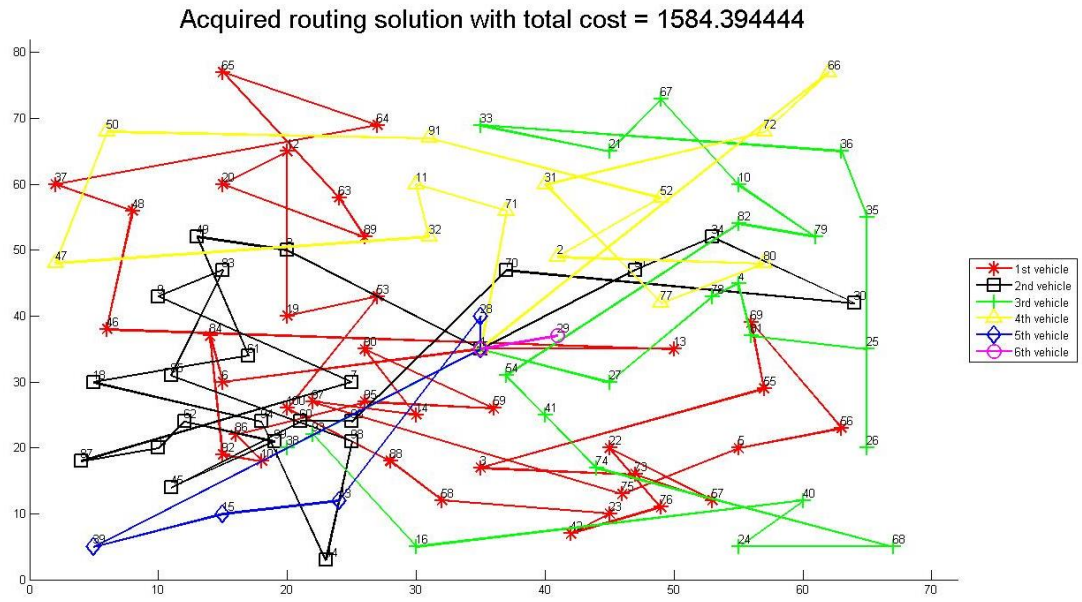


Σχήμα 30: *Δρομολόγηση οχημάτων R201*

ΠΕΛΑΤΕΣ ΑΝΑ ΟΧΗΜΑ											
1	28	29	34	66	72	31	52	10	82	80	79
4	51	35	69	36	25	5	75	14	94	92	101
90	71	59	1	0	0	0	0	0	0	0	0
1	60	15	93	84	6	46	37	48	83	63	32
70	77	89	20	8	19	91	11	21	67	33	49
2	78	81	26	1	0	0	0	0	0	0	0
1	96	43	16	3	22	76	68	24	74	41	86
88	58	23	42	57	56	55	27	18	61	1	0
1	73	40	30	13	99	62	17	45	39	87	85
9	47	97	44	98	38	1	0	0	0	0	0
1	64	12	65	50	1	0	0	0	0	0	0
1	53	7	54	1	0	0	0	0	0	0	0
1	100	95	1	0	0	0	0	0	0	0	0

Πίνακας 30: *Δρομολόγηση οχημάτων R201*

31. Αποτελέσματα προβλήματος R202

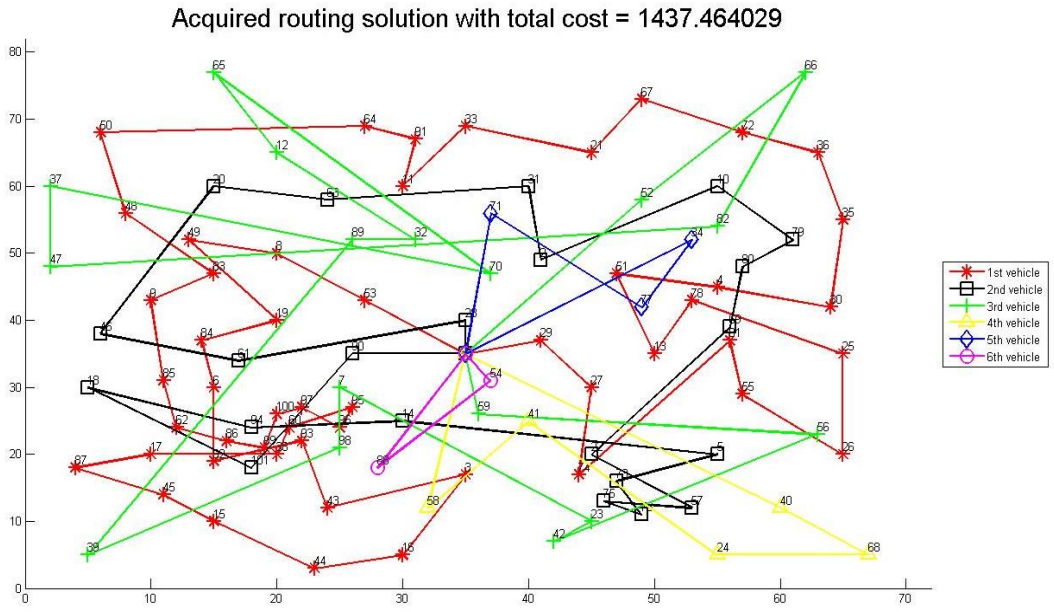


Σχήμα 31: *Δρομολόγηση οχημάτων R202*

ΠΕΛΑΤΕΣ ΑΝΑ ΟΧΗΜΑ											
1	13	46	48	37	64	65	63	89	20	12	19
53	100	88	58	23	42	76	22	57	73	3	55
69	56	5	75	97	14	90	59	95	86	101	92
84	6	1	0	0	0	0	0	0	0	0	0
1	51	34	30	70	96	60	45	99	62	17	87
7	9	83	85	98	44	94	18	61	49	8	1
1	38	93	16	40	24	68	74	41	54	82	79
10	67	21	33	36	35	26	25	81	4	78	27
1	0	0	0	0	0	0	0	0	0	0	0
1	66	72	31	77	80	2	52	91	50	47	32
11	71	1	0	0	0	0	0	0	0	0	0
1	28	43	15	39	1	0	0	0	0	0	0
1	29	1	0	0	0	0	0	0	0	0	0

Πίνακας 31: *Δρομολόγηση οχημάτων R202*

32. Αποτελέσματα προβλήματος R203

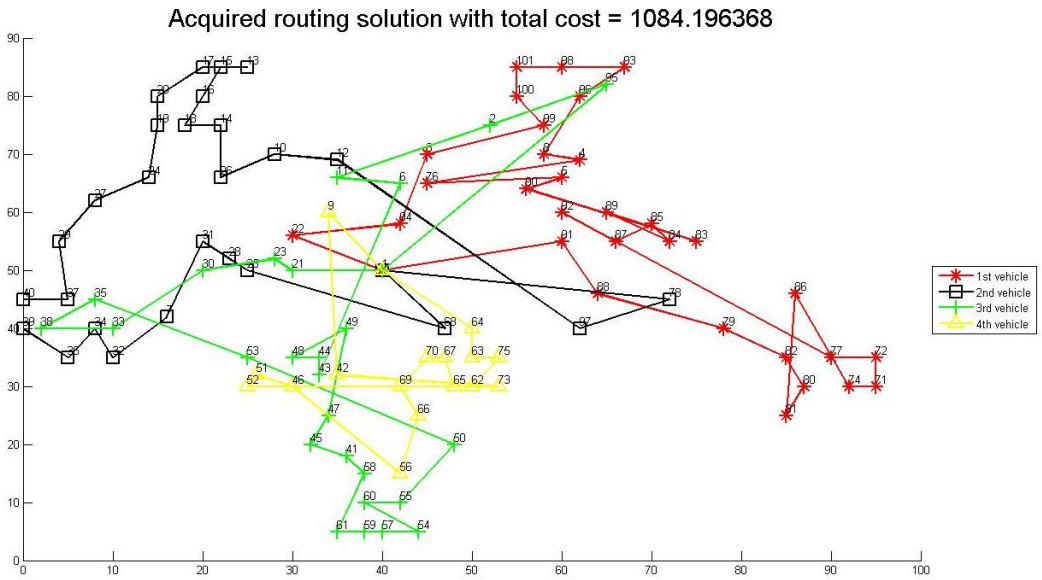


Σχήμα 32: *Δρομολόγηση οχημάτων R203*

ΠΕΛΑΤΕΣ ΑΝΑ ΟΧΗΜΑ											
1	53	8	49	19	84	6	92	93	43	3	16
44	15	45	87	17	38	60	95	96	97	100	99
86	62	85	9	83	48	50	64	91	11	33	21
67	72	36	35	30	4	51	13	78	25	26	55
81	74	27	29	1	0	0	0	0	0	0	0
1	28	61	46	20	63	31	2	10	79	80	69
22	57	75	76	73	5	14	94	18	101	90	1
1	52	66	82	47	37	70	65	12	32	89	39
98	7	23	42	56	59	1	0	0	0	0	0
1	40	68	24	41	58	1	0	0	0	0	0
1	34	77	71	1	0	0	0	0	0	0	0
1	88	54	1	0	0	0	0	0	0	0	0

Πίνακας 32: *Δρομολόγηση οχημάτων R203*

33. Αποτελέσματα προβλήματος R204

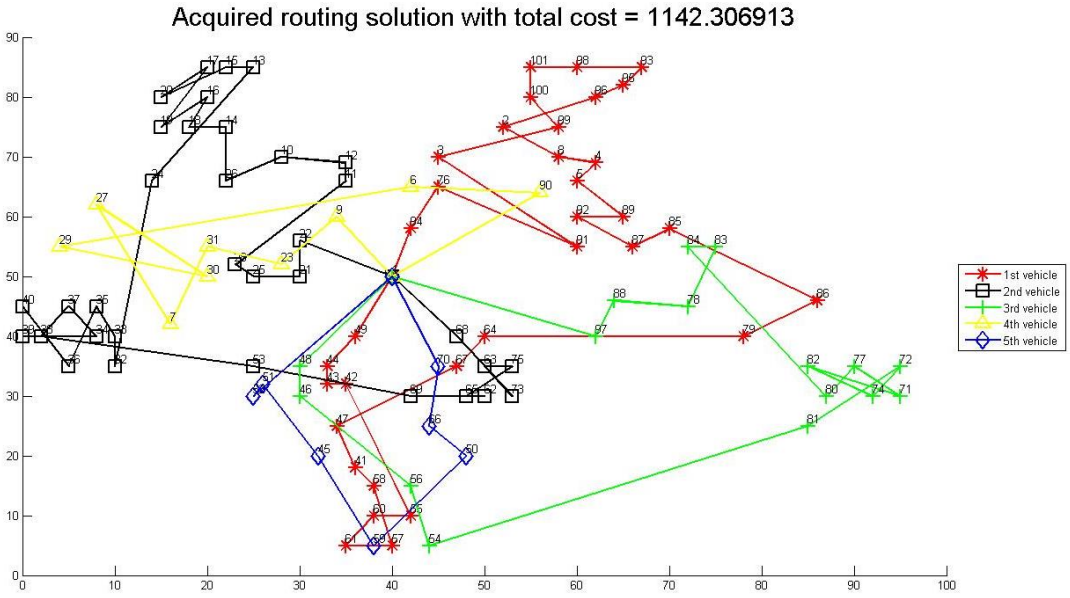


Σχήμα 33: *Δρομολόγηση οχημάτων R204*

ΠΕΛΑΤΕΣ ΑΝΑ ΟΧΗΜΑ											
1	22	94	3	99	100	101	98	93	96	8	4
76	5	90	83	89	84	85	87	92	77	72	71
74	86	81	80	82	79	88	91	1	0	0	0
1	68	25	28	31	7	32	34	36	39	40	37
29	27	24	19	20	17	13	15	16	18	14	26
10	12	97	78	1	0	0	0	0	0	0	0
1	21	23	30	33	38	35	53	50	55	60	54
57	59	61	58	41	45	47	49	48	44	43	6
11	2	95	1	0	0	0	0	0	0	0	0
1	64	63	75	62	65	67	70	69	66	56	46
51	52	73	42	9	1	0	0	0	0	0	0

Πίνακας 33: *Δρομολόγηση οχημάτων R204*

34. Αποτελέσματα προβλήματος R205

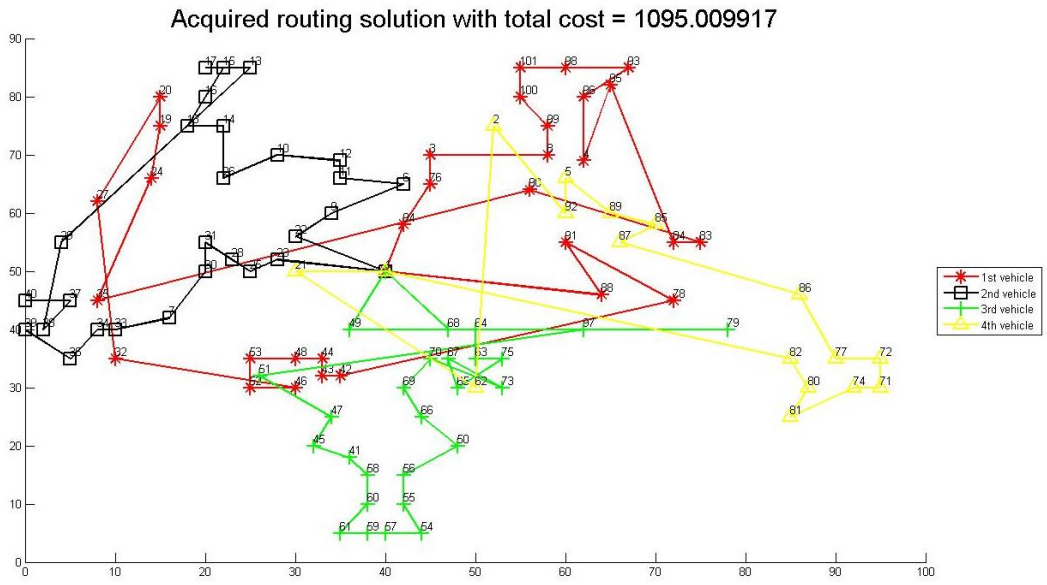


Σχήμα 34: *Δρομολόγηση οχημάτων R205*

ΠΕΛΑΤΕΣ ΑΝΑ ΟΧΗΜΑ																			
1	94	76	6	3	8	100	101	98	93	95	2	99	4	5	91	84	87	85	92
89	97	50	62	66	58	61	57	55	42	51	46	44	52	48	1				
1	69	63	75	73	47	43	7	34	40	35	33	32	39	29	37	36	27	24	19
17	13	15	20	16	18	14	26	10	12	11	22	21	31	25	1				
1	68	64	70	67	79	74	72	82	77	80	81	71	86	78	83	90	1	0	0
1	28	53	49	38	30	23	9	96	88	1	0	0	0	0	0	0	0	0	0
1	65	56	54	60	59	41	45	1	0	0	0	0	0	0	0	0	0	0	0

Πίνακας 34: *Δρομολόγηση οχημάτων R205*

35. Αποτελέσματα προβλήματος R206

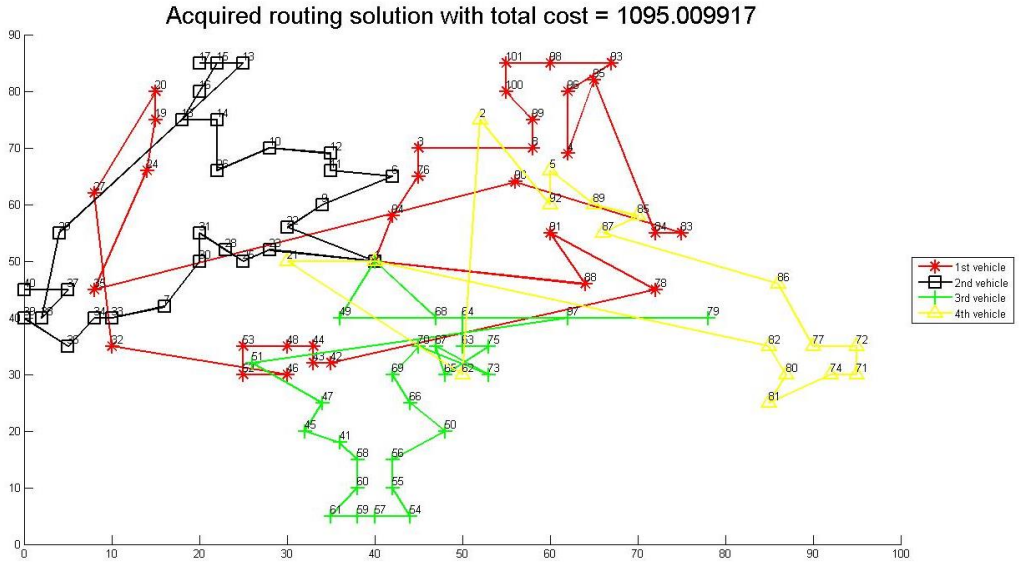


Σχήμα 35: *Δρομολόγηση οχημάτων R206*

ΠΕΛΑΤΕΣ ΑΝΑ ΟΧΗΜΑ											
1	94	76	3	8	99	100	101	98	93	96	4
95	84	83	90	35	24	19	20	27	32	46	52
53	48	44	43	42	78	91	88	1	0	0	0
1	23	25	28	31	30	7	33	34	36	39	40
37	38	29	13	17	15	16	18	14	26	10	12
11	6	9	22	1	0	0	0	0	0	0	0
1	68	64	63	75	65	67	73	70	69	66	50
56	55	54	57	59	61	60	58	41	45	47	51
97	79	49	1	0	0	0	0	0	0	0	0
1	21	62	2	92	5	89	85	87	86	77	72
71	74	81	80	82	1	0	0	0	0	0	0

Πίνακας 35: *Δρομολόγηση οχημάτων R206*

36. Αποτελέσματα προβλήματος R207

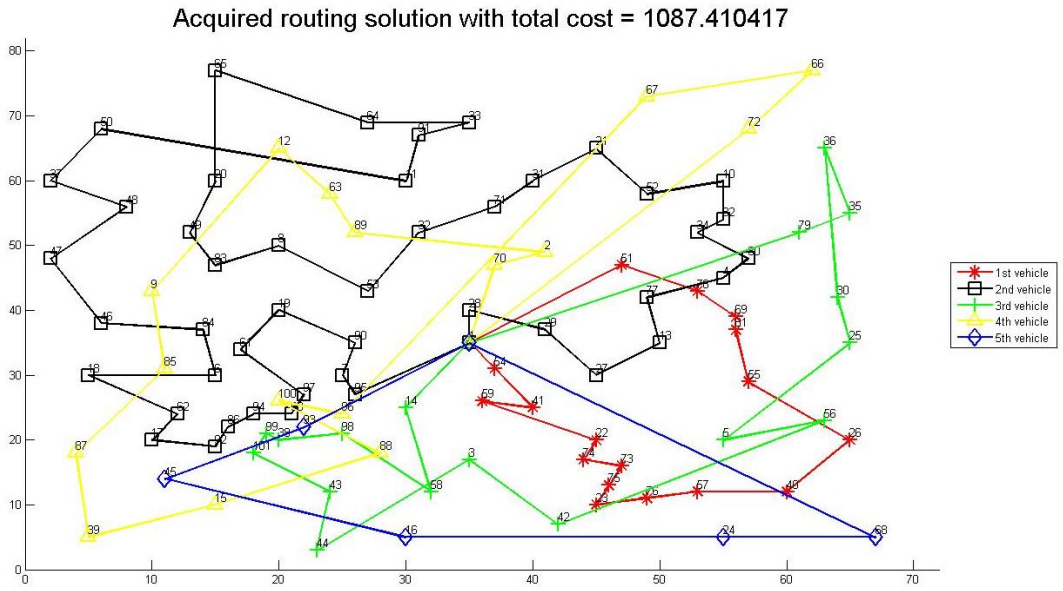


Σχήμα 36: *Δρομολόγηση οχημάτων R207*

ΠΕΛΑΤΕΣ ΑΝΑ ΟΧΗΜΑ											
1	68	94	2	96	95	100	101	98	93	4	99
76	6	8	5	90	3	13	20	17	15	72	71
74	81	80	82	79	78	88	84	1	0	0	0
1	53	23	31	30	33	34	38	36	32	39	40
35	37	29	27	24	19	16	18	14	26	10	12
11	22	9	91	1	0	0	0	0	0	0	0
1	21	63	75	73	62	65	67	70	69	66	50
56	60	55	54	57	59	61	58	41	45	47	46
52	51	44	1	0	0	0	0	0	0	0	0
1	97	64	25	28	7	92	89	85	87	83	86
77	48	43	49	42	1	0	0	0	0	0	0

Πίνακας 36: *Δρομολόγηση οχημάτων R207*

37. Αποτελέσματα προβλήματος R208

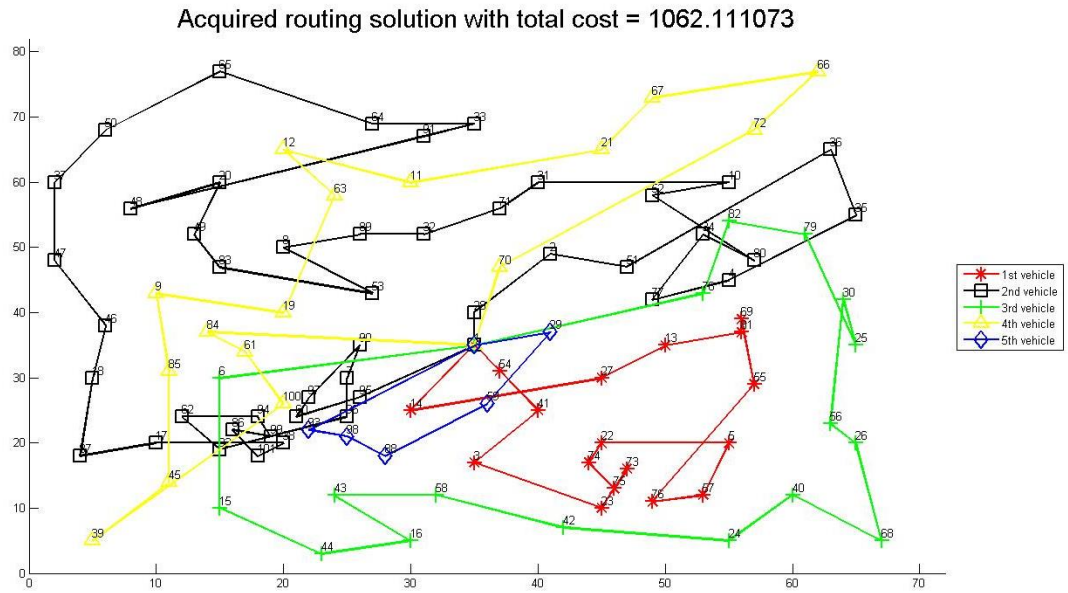


Σχήμα 37: *Δρομολόγηση οχημάτων R208*

ΠΕΛΑΤΕΣ ΑΝΑ ΟΧΗΜΑ											
1	54	41	59	22	74	73	75	23	76	57	40
26	55	81	69	78	51	1	0	0	0	0	0
1	28	29	27	13	77	4	80	34	82	10	52
21	31	71	32	53	8	83	49	20	65	64	33
91	11	50	37	48	47	46	84	6	18	62	17
91	11	50	37	48	47	46	84	6	18	62	17
1	79	35	36	30	25	5	56	42	3	44	43
101	99	38	98	58	14	1	0	0	0	0	0
1	70	2	89	63	12	9	85	87	39	15	88
100	96	67	66	72	1	0	0	0	0	0	0
1	93	45	16	24	68	1	0	0	0	0	0

Πίνακας 37: *Δρομολόγηση οχημάτων R208*

38. Αποτελέσματα προβλήματος R209

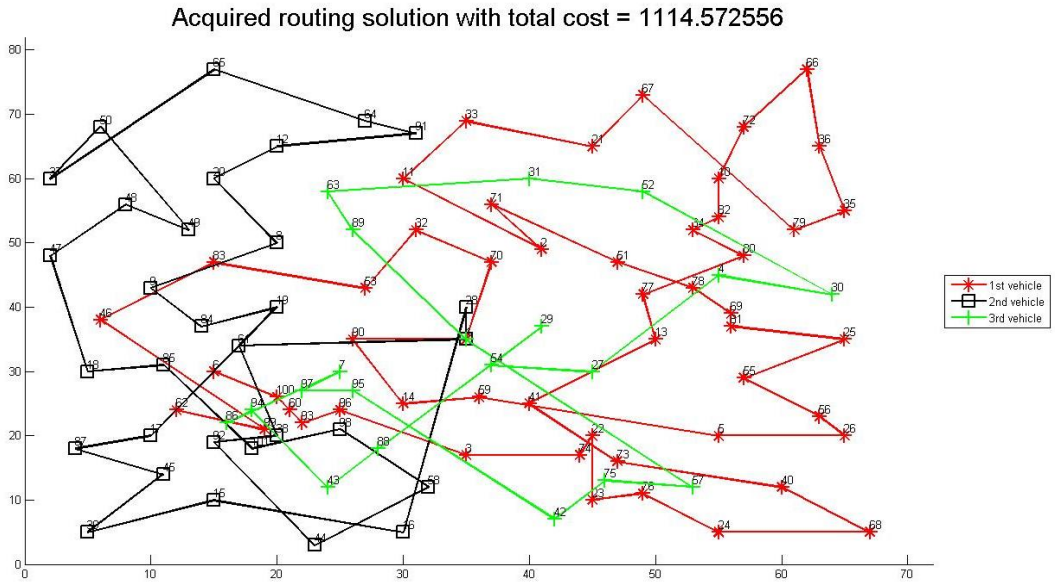


Σχήμα 38: *Δρομολόγηση οχημάτων R209*

ΠΕΛΑΤΕΣ ΑΝΑ ΟΧΗΜΑ											
1	54	41	3	23	73	75	74	22	5	57	76
55	69	81	13	27	14	1	0	0	0	0	0
1	28	2	51	36	35	4	77	34	80	52	10
31	71	32	89	8	53	83	49	20	48	91	33
64	65	50	37	47	46	18	87	17	38	101	86
99	94	62	92	96	7	90	97	60	95	1	0
1	6	15	44	16	43	58	42	24	40	68	26
56	30	25	79	82	78	1	0	0	0	0	0
1	70	72	66	67	21	11	12	63	19	9	85
45	39	100	61	84	1	0	0	0	0	0	0
1	93	98	88	59	29	1	0	0	0	0	0

Πίνακας 38: *Δρομολόγηση οχημάτων R209*

39. Αποτελέσματα προβλήματος R210

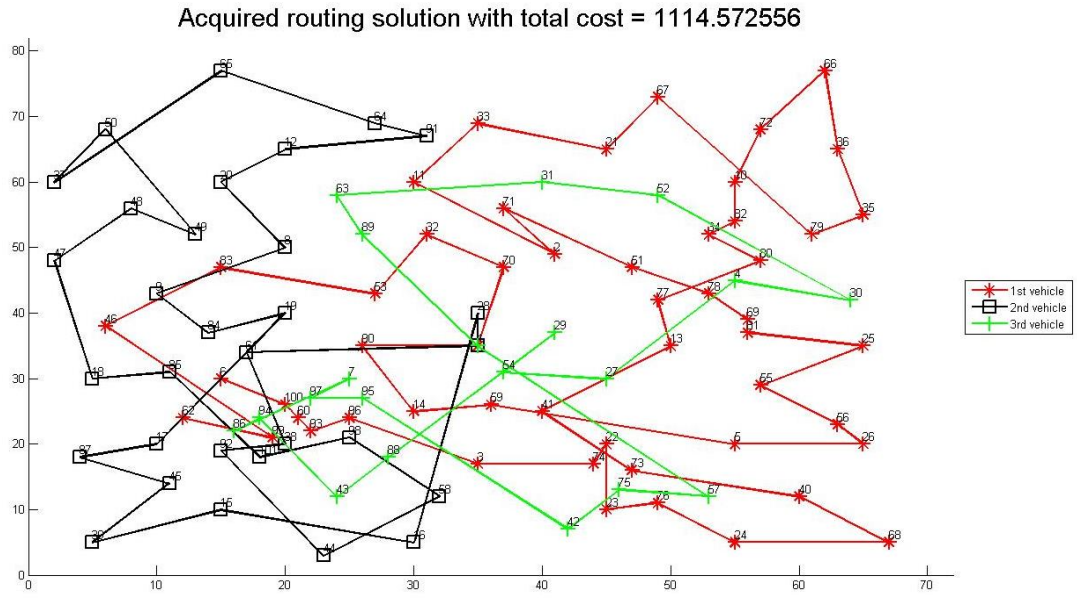


Σχήμα 39: *Δρομολόγηση οχημάτων R210*

ΠΕΛΑΤΕΣ ΑΝΑ ΟΧΗΜΑ											
1	70	32	53	83	46	99	62	6	100	60	93
96	3	74	22	23	76	24	68	40	73	41	13
77	80	34	82	10	72	66	36	35	79	67	21
33	11	2	71	51	78	69	81	25	55	56	26
5	59	14	90	1	0	0	0	0	0	0	0
1	28	16	15	39	45	87	17	19	84	9	8
20	12	91	64	65	37	50	49	48	47	18	85
101	98	58	44	92	38	61	1	0	0	0	0
1	89	63	31	52	30	4	27	54	29	88	43
94	86	7	97	95	42	75	57	1	0	0	0

Πίνακας 39: *Δρομολόγηση οχημάτων R210*

40. Αποτελέσματα προβλήματος R211



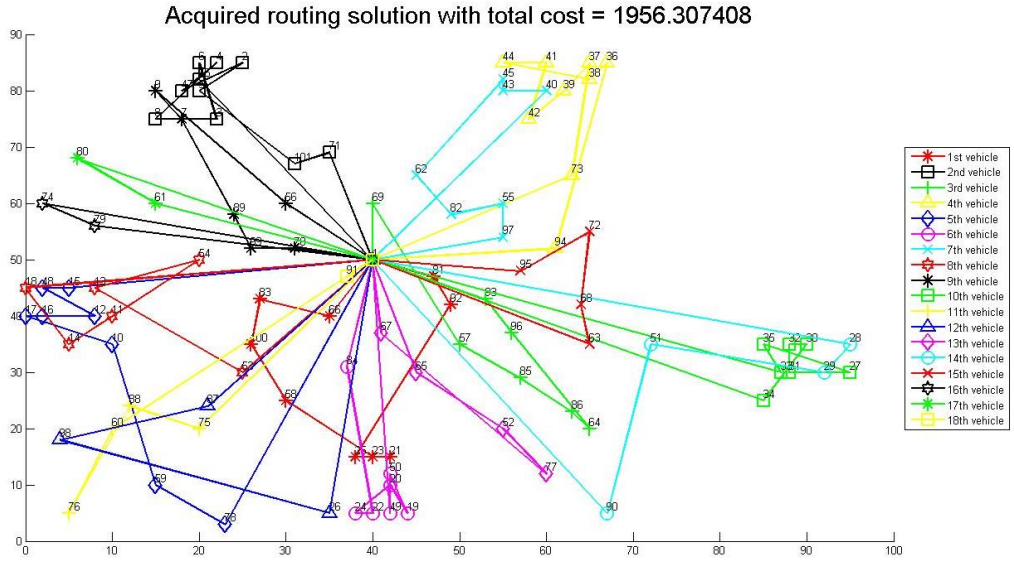
Σχήμα 40: *Δρομολόγηση οχημάτων R211*

ΠΕΛΑΤΕΣ ΑΝΑ ΟΧΗΜΑ											
1	70	32	53	83	46	99	62	6	100	60	93
96	3	74	22	23	76	24	68	40	73	41	13
77	80	34	82	10	72	66	36	35	79	67	21
33	11	2	71	51	78	69	81	25	55	56	26
5	59	14	90	1	0	0	0	0	0	0	0
1	28	16	15	39	45	87	17	19	84	9	8
20	12	91	64	65	37	50	49	48	47	18	85
101	98	58	44	92	38	61	1	0	0	0	0
1	89	63	31	52	30	4	27	54	29	88	43
94	86	7	97	95	42	75	57	1	0	0	0

Πίνακας 40: *Δρομολόγηση οχημάτων R211*

Προβλήματα κατηγορίας RC1

41. Αποτελέσματα προβλήματος RC101

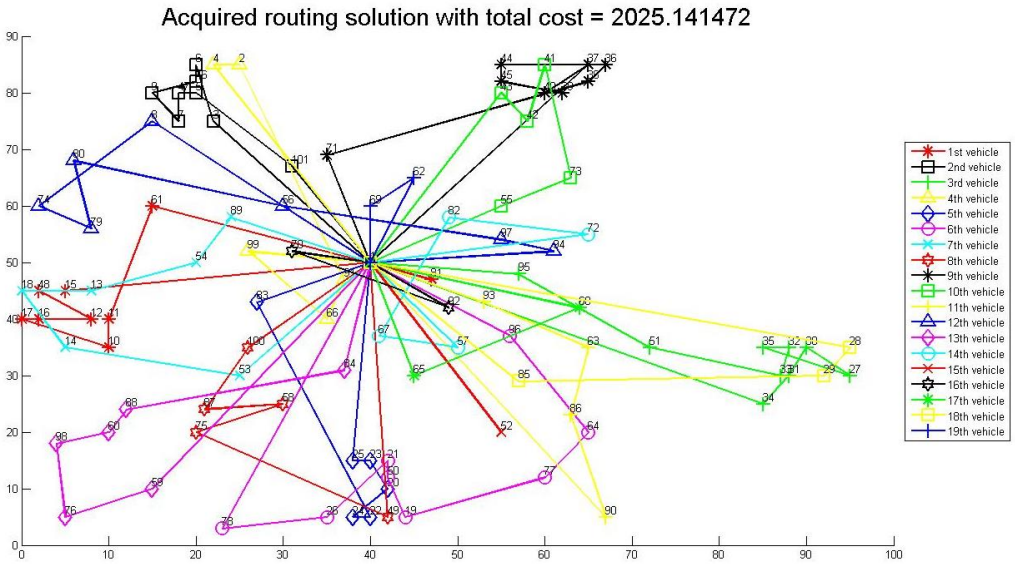


Σχήμα 41: *Δρομολόγηση οχημάτων RC101*

ΠΕΛΑΤΕΣ ΑΝΑ ΟΧΗΜΑ											
1	66	83	53	100	87	58	21	49	81	1	0
1	6	46	3	8	9	47	4	2	5	101	1
1	84	22	24	20	19	25	26	1	0	0	0
1	43	40	37	42	41	44	38	36	71	1	0
1	15	48	16	17	74	11	14	18	1	0	0
1	93	63	52	50	23	78	59	1	0	0	0
1	73	72	68	95	97	55	94	1	0	0	0
1	96	64	77	86	85	90	1	0	0	0	0
1	70	99	89	7	69	61	1	0	0	0	0
1	34	29	30	31	27	35	33	1	0	0	0
1	65	10	88	75	1	0	0	0	0	0	0
1	60	76	98	1	0	0	0	0	0	0	0
1	12	13	91	57	1	0	0	0	0	0	0
1	28	32	51	67	1	0	0	0	0	0	0
1	45	62	82	92	1	0	0	0	0	0	0
1	54	56	1	0	0	0	0	0	0	0	0
1	80	79	1	0	0	0	0	0	0	0	0
1	39	1	0	0	0	0	0	0	0	0	0

Πίνακας 41: *Δρομολόγηση οχημάτων RC101*

42. Αποτελέσματα προβλήματος RC102

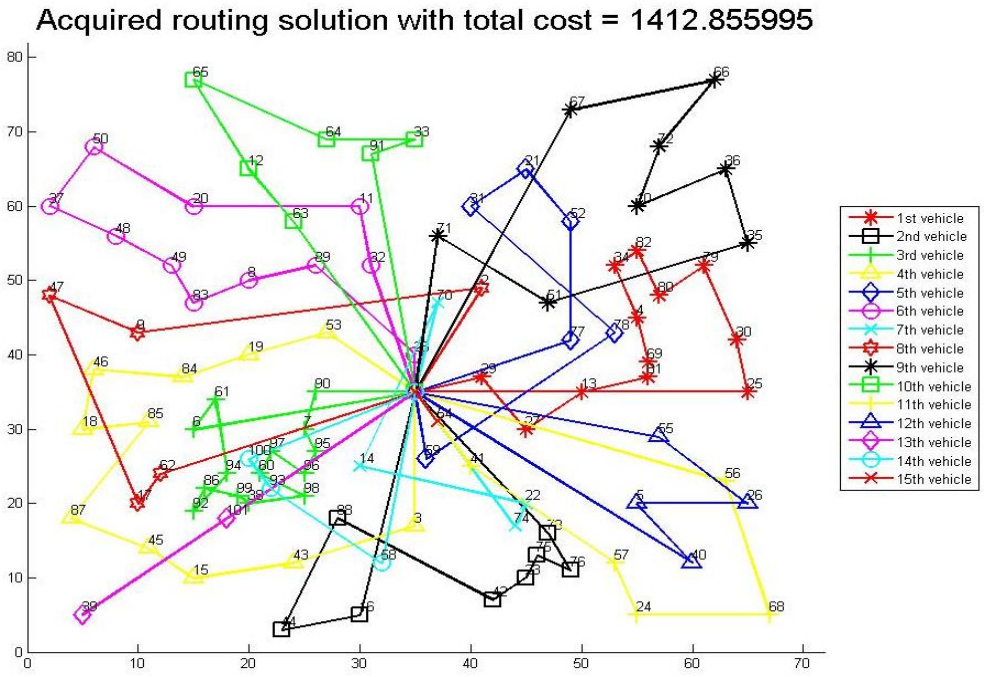


Σχήμα 42: *Δρομολόγηση οχημάτων RC102*

ΠΕΛΑΤΕΣ ΑΝΑ ΟΧΗΜΑ										
1	15	48	12	16	17	10	11	61	81	1
1	3	6	46	9	7	47	5	101	1	0
1	34	31	30	27	35	32	33	51	95	1
1	99	66	91	2	4	1	0	0	0	0
1	83	22	24	20	23	25	1	0	0	0
1	96	64	77	19	50	21	26	78	1	0
1	89	54	13	18	14	53	1	0	0	0
1	100	87	58	75	49	1	0	0	0	0
1	37	40	39	45	44	36	38	71	1	0
1	43	42	41	73	55	1	0	0	0	0
1	93	63	86	90	1	0	0	0	0	0
1	8	74	79	80	56	97	94	1	0	0
1	84	88	60	98	76	59	1	0	0	0
1	72	82	67	57	1	0	0	0	0	0
1	52	1	0	0	0	0	0	0	0	0
1	70	92	1	0	0	0	0	0	0	0
1	65	68	1	0	0	0	0	0	0	0
1	28	29	85	1	0	0	0	0	0	0
1	62	69	1	0	0	0	0	0	0	0

Πίνακας 42: *Δρομολόγηση οχημάτων RC102*

43. Αποτελέσματα προβλήματος RC103

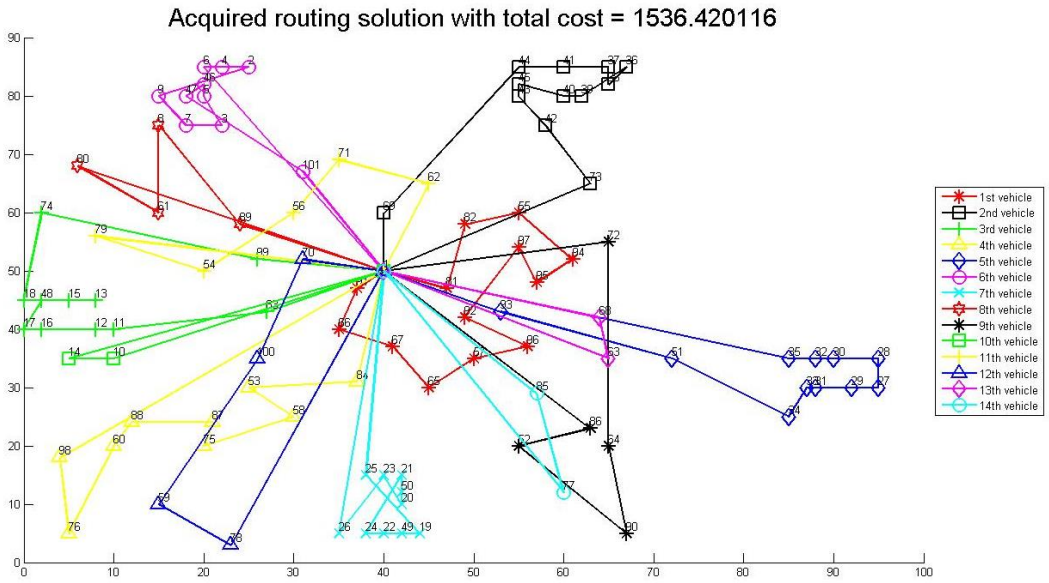


Σχήμα 43: *Δρομολόγηση οχημάτων RC103*

ΠΕΛΑΤΕΣ ΑΝΑ ΟΧΗΜΑ															
1	25	30	79	80	82	34	4	69	81	13	27	29	1	0	0
1	16	44	88	42	23	75	76	73	1	0	0	0	0	0	0
1	6	61	94	92	86	99	38	98	60	97	96	95	7	90	1
1	3	43	15	45	87	85	18	46	84	19	53	1	0	0	0
1	77	52	21	31	78	59	1	0	0	0	0	0	0	0	0
1	32	11	20	50	37	48	49	83	8	89	28	1	0	0	0
1	70	14	22	74	1	0	0	0	0	0	0	0	0	0	0
1	62	17	47	9	2	1	0	0	0	0	0	0	0	0	0
1	71	51	35	36	10	72	66	67	1	0	0	0	0	0	0
1	63	12	65	64	33	91	1	0	0	0	0	0	0	0	0
1	41	57	24	68	56	1	0	0	0	0	0	0	0	0	0
1	40	5	26	55	1	0	0	0	0	0	0	0	0	0	0
1	39	101	1	0	0	0	0	0	0	0	0	0	0	0	0
1	58	93	100	1	0	0	0	0	0	0	0	0	0	0	0
1	54	1	0	0	0	0	0	0	0	0	0	0	0	0	0

Πίνακας 43: *Δρομολόγηση οχημάτων RC103*

44. Αποτελέσματα προβλήματος RC104

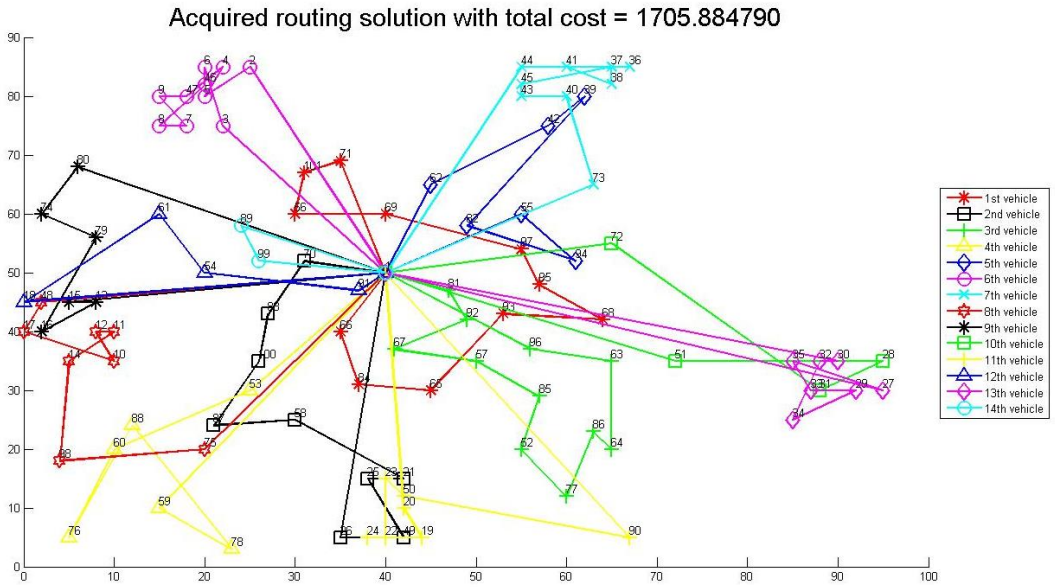


Σχήμα 44: *Δρομολόγηση οχημάτων RC104*

ΠΕΛΑΤΕΣ ΑΝΑ ΟΧΗΜΑ														
1	91	66	67	65	57	96	92	97	95	94	55	82	81	1
1	73	42	43	45	40	39	36	38	37	41	44	69	1	0
1	83	11	12	16	17	48	13	15	18	74	99	1	0	0
1	98	76	60	88	87	75	58	53	84	1	0	0	0	0
1	93	51	34	33	31	29	27	28	30	32	35	1	0	0
1	6	4	2	9	7	3	5	46	47	101	1	0	0	0
1	25	19	49	22	24	21	50	20	23	26	1	0	0	0
1	80	61	8	89	1	0	0	0	0	0	0	0	0	0
1	72	64	90	52	86	1	0	0	0	0	0	0	0	0
1	10	14	1	0	0	0	0	0	0	0	0	0	0	0
1	79	54	56	71	62	1	0	0	0	0	0	0	0	0
1	70	100	59	78	1	0	0	0	0	0	0	0	0	0
1	63	68	1	0	0	0	0	0	0	0	0	0	0	0
1	77	85	1	0	0	0	0	0	0	0	0	0	0	0

Πίνακας 44: *Δρομολόγηση οχημάτων RC104*

45. Αποτελέσματα προβλήματος RC105

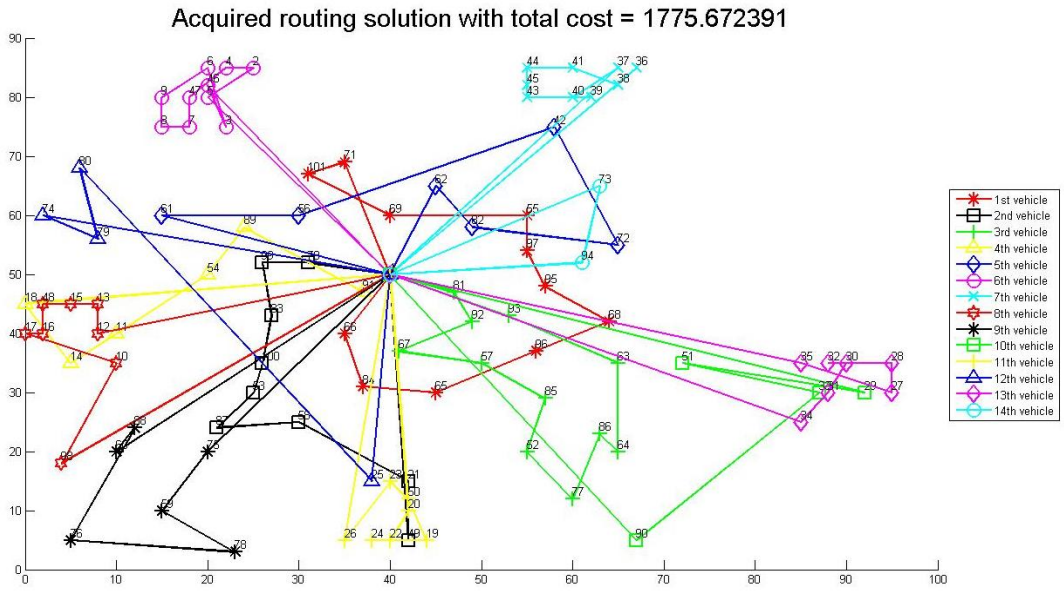


Σχήμα 45: *Δρομολόγηση οχημάτων RC105*

ΠΕΛΑΤΕΣ ΑΝΑ ΟΧΗΜΑ												
1	66	84	65	93	68	95	97	69	56	101	71	1
1	70	83	100	87	58	21	25	49	26	1	0	0
1	96	63	64	86	77	52	85	57	67	92	81	1
1	53	60	76	88	78	59	1	0	0	0	0	0
1	62	42	39	82	94	55	1	0	0	0	0	0
1	3	6	46	8	7	9	47	4	5	2	1	0
1	73	40	43	45	37	38	41	36	44	1	0	0
1	48	17	10	12	11	14	98	75	1	0	0	0
1	15	13	16	79	74	80	1	0	0	0	0	0
1	72	31	28	51	1	0	0	0	0	0	0	0
1	20	19	24	22	23	50	90	1	0	0	0	0
1	91	54	61	18	1	0	0	0	0	0	0	0
1	30	32	34	29	33	35	27	1	0	0	0	0
1	89	99	1	0	0	0	0	0	0	0	0	0

Πίνακας 45: *Δρομολόγηση οχημάτων RC105*

46. Αποτελέσματα προβλήματος RC106

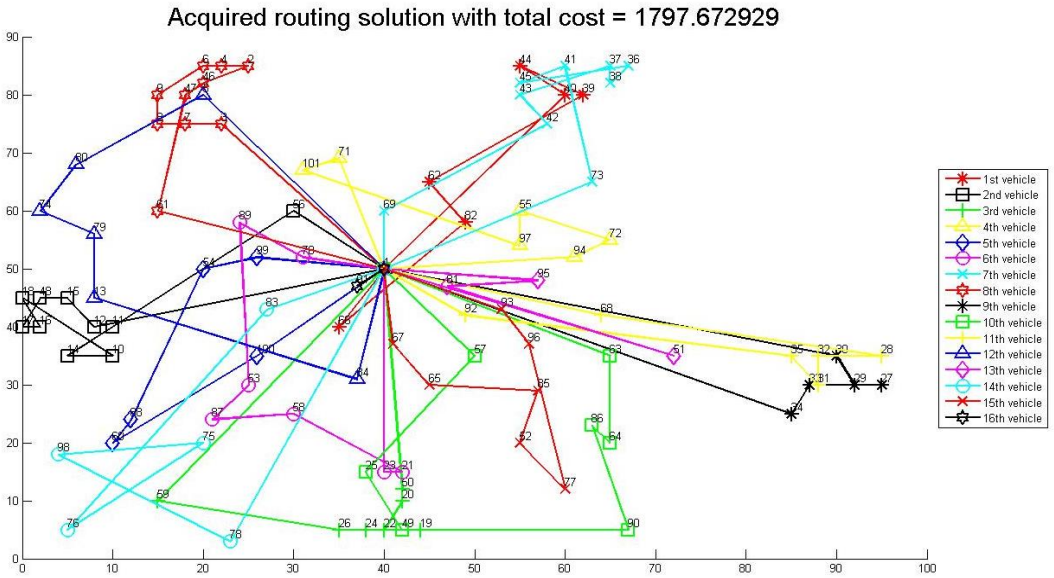


Σχήμα 46: *Δρομολόγηση οχημάτων RC106*

ΠΕΛΑΤΕΣ ΑΝΑ ΟΧΗΜΑ												
1	66	84	65	96	68	95	97	55	69	101	71	1
1	70	99	83	100	53	87	58	21	49	1	0	0
1	93	63	64	86	77	52	85	57	67	92	81	1
1	91	89	54	11	14	18	1	0	0	0	0	0
1	62	82	72	42	56	61	1	0	0	0	0	0
1	46	3	6	9	8	7	47	4	2	5	1	0
1	37	40	39	43	45	44	41	38	36	1	0	0
1	12	13	15	48	16	17	10	98	1	0	0	0
1	60	88	76	78	59	75	1	0	0	0	0	0
1	29	51	33	90	1	0	0	0	0	0	0	0
1	20	22	24	19	50	23	26	1	0	0	0	0
1	74	79	80	25	1	0	0	0	0	0	0	0
1	34	31	30	32	28	27	35	1	0	0	0	0
1	73	94	1	0	0	0	0	0	0	0	0	0

Πίνακας 46: *Δρομολόγηση οχημάτων RC106*

47. Αποτελέσματα προβλήματος RC107

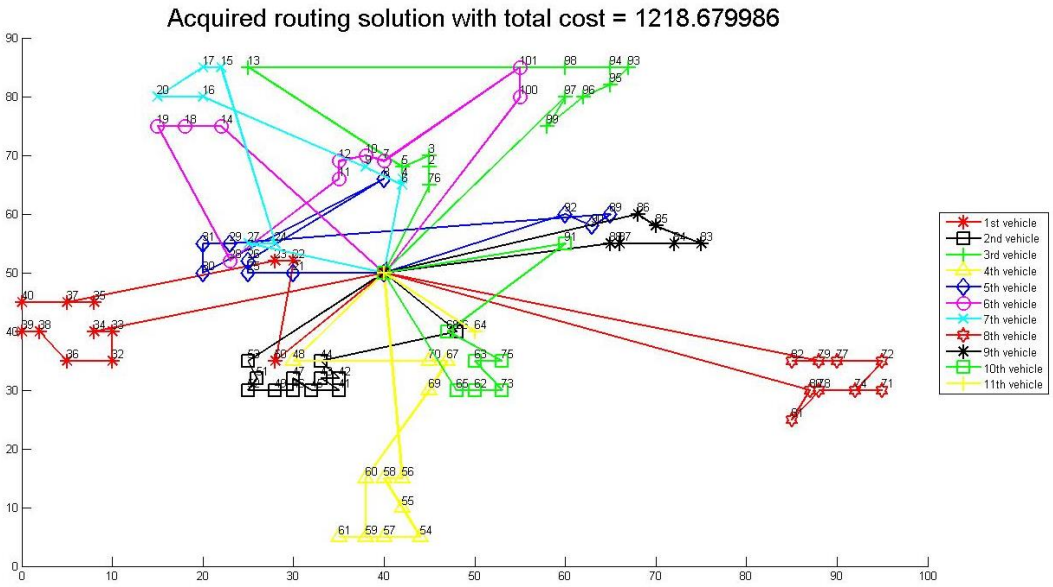


Σχήμα 47.: *Δρομολόγηση οχημάτων RC107*

ΠΕΛΑΤΕΣ ΑΝΑ ΟΧΗΜΑ											
1	66	82	62	39	44	40	1	0	0	0	0
1	12	15	48	17	16	18	10	14	11	56	1
1	50	20	22	19	24	26	59	1	0	0	0
1	94	72	55	97	101	71	1	0	0	0	0
1	99	54	88	60	100	1	0	0	0	0	0
1	70	89	53	87	58	21	23	1	0	0	0
1	73	41	45	36	38	37	43	42	69	1	0
1	3	7	8	9	6	4	2	46	47	61	1
1	30	29	27	33	34	1	0	0	0	0	0
1	63	64	86	90	49	25	57	1	0	0	0
1	68	28	32	31	35	92	1	0	0	0	0
1	84	13	79	74	80	5	1	0	0	0	0
1	95	81	51	1	0	0	0	0	0	0	0
1	83	76	75	98	78	1	0	0	0	0	0
1	93	96	77	52	85	65	67	1	0	0	0
1	91	1	0	0	0	0	0	0	0	0	0

Πίνακας 47.: *Δρομολόγηση οχημάτων RC107*

48. Αποτελέσματα προβλήματος RC108



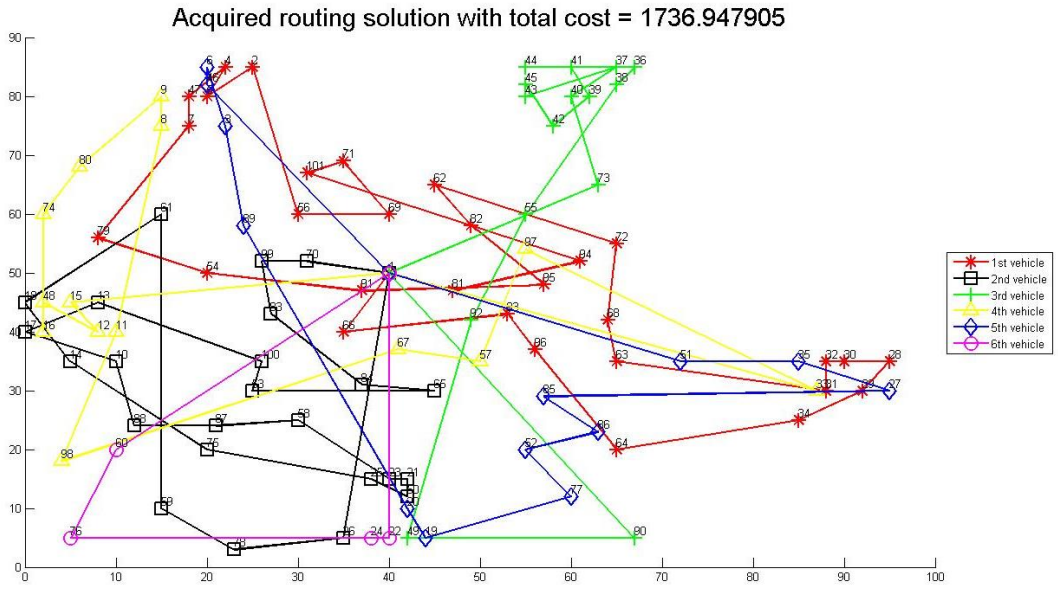
Σχήμα 48: *Δρομολόγηση οχημάτων RC108*

ΠΕΛΑΤΕΣ ΑΝΑ ΟΧΗΜΑ													
1	34	33	32	36	38	39	40	35	37	23	22	50	1
1	66	44	42	43	41	45	47	46	49	52	51	53	1
1	97	99	96	95	93	94	98	13	5	3	2	76	1
1	56	58	55	54	57	61	59	60	69	67	70	48	1
1	21	25	26	8	30	31	29	89	90	92	1	0	0
1	14	18	19	28	11	12	10	7	101	100	1	0	0
1	4	6	9	16	20	17	15	24	27	1	0	0	0
1	79	82	77	72	74	71	78	81	80	1	0	0	0
1	88	84	87	83	85	86	1	0	0	0	0	0	0
1	91	68	75	63	73	62	65	1	0	0	0	0	0
1	64	1	0	0	0	0	0	0	0	0	0	0	0

Πίνακας 48: *Δρομολόγηση οχημάτων RC108*

Προβλήματα κατηγορίας RC2

49. Αποτελέσματα προβλήματος RC201

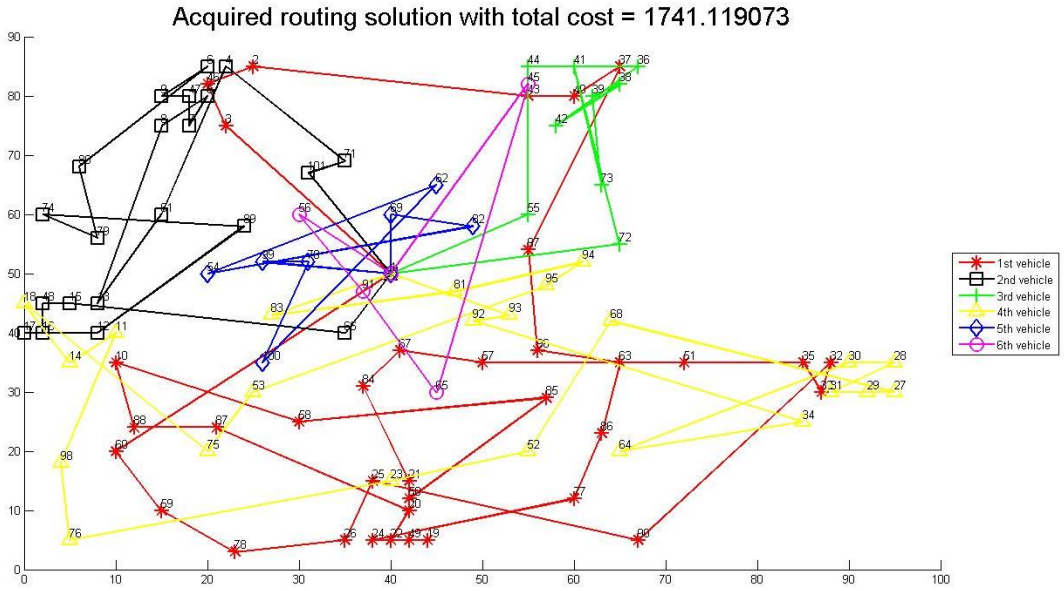


Σχήμα 49: *Δρομολόγηση οχημάτων RC201*

ΠΕΛΑΤΕΣ ΑΝΑ ΟΧΗΜΑ											
1	66	93	96	64	34	29	28	40	32	31	63
68	72	62	82	95	91	53	79	7	47	4	5
2	56	69	71	101	94	81	1	0	0	0	0
1	70	99	83	84	65	53	100	13	17	10	88
87	58	23	21	50	25	75	14	18	61	59	78
26	1	0	0	0	0	0	0	0	0	0	0
1	73	40	37	43	45	42	39	41	44	36	38
55	92	49	90	1	0	0	0	0	0	0	0
1	15	12	48	16	74	80	9	8	11	98	67
57	97	33	1	0	0	0	0	0	0	0	0
1	46	6	3	89	20	19	77	52	86	85	27
35	51	1	0	0	0	0	0	0	0	0	0
1	60	76	24	22	1	0	0	0	0	0	0

Πίνακας 49: *Δρομολόγηση οχημάτων RC201*

50. Αποτελέσματα προβλήματος RC202

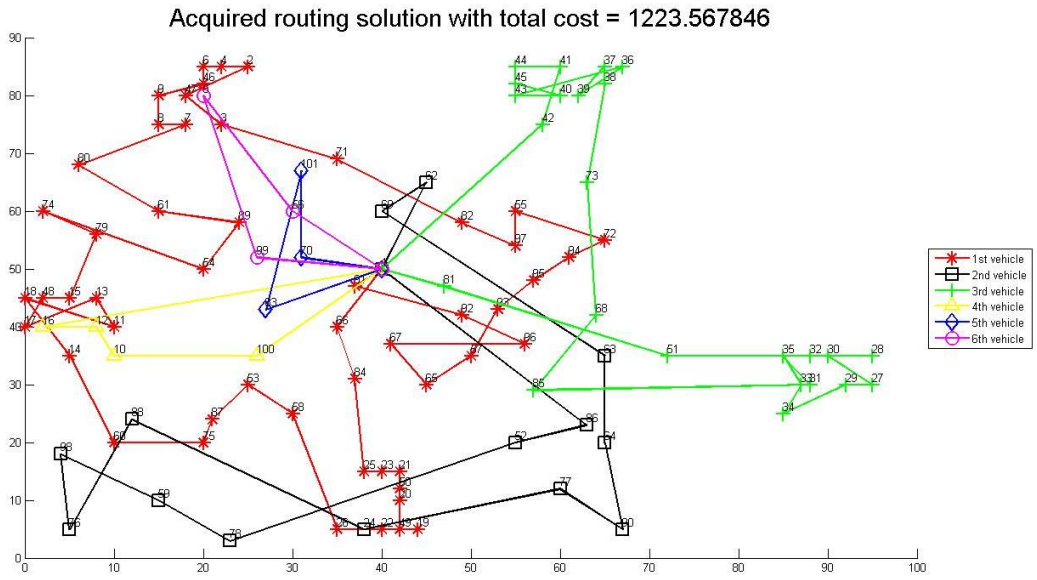


Σχήμα 50: *Δρομολόγηση οχημάτων RC202*

ΠΕΛΑΤΕΣ ΑΝΑ ΟΧΗΜΑ											
1	3	46	2	43	40	37	97	96	63	86	77
24	49	19	22	20	87	88	10	58	85	50	21
84	67	57	51	35	33	32	90	25	26	78	59
60	1	0	0	0	0	0	0	0	0	0	0
1	66	15	48	16	17	12	89	74	79	80	6
9	47	7	5	8	13	61	4	71	101	1	0
1	72	41	73	39	38	42	36	44	55	1	0
1	93	92	34	64	30	28	31	29	27	68	52
23	76	98	11	14	18	75	53	95	94	81	83
1	0	0	0	0	0	0	0	0	0	0	0
1	99	70	100	62	54	82	69	1	0	0	0
1	45	65	91	56	1	0	0	0	0	0	0

Πίνακας 50: *Δρομολόγηση οχημάτων RC202*

51. Αποτελέσματα προβλήματος RC203

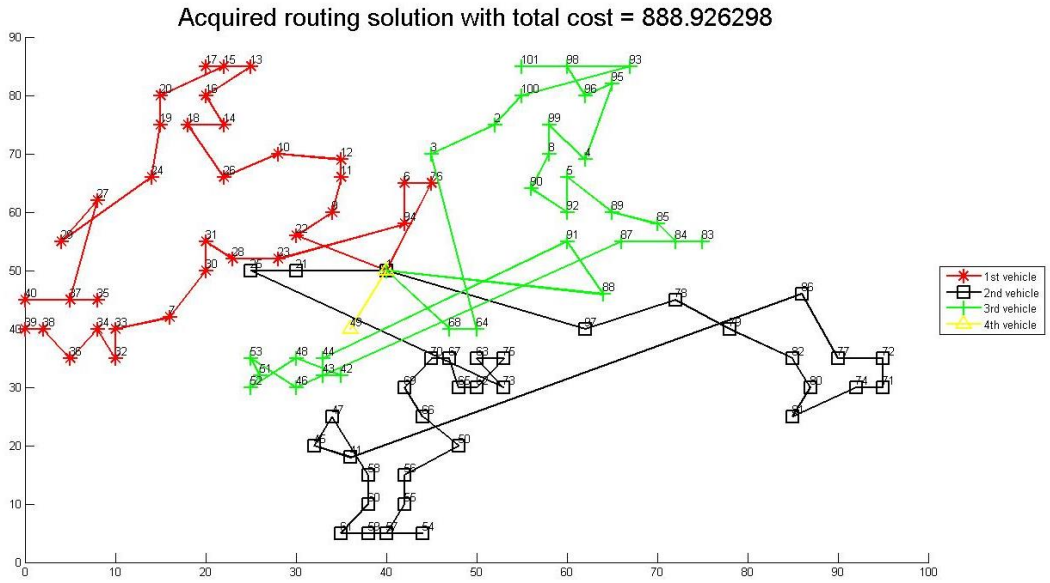


Σχήμα 51: *Δρομολόγηση οχημάτων RC203*

ΠΕΛΑΤΕΣ ΑΝΑ ΟΧΗΜΑ											
1	91	92	96	67	65	57	93	95	94	72	55
97	82	71	3	47	2	4	6	46	9	8	7
80	61	89	54	74	79	15	48	17	13	11	18
14	60	75	87	53	58	26	22	19	49	20	50
21	23	25	84	66	1	0	0	0	0	0	0
1	62	69	63	64	90	77	24	88	76	98	59
78	52	86	1	0	0	0	0	0	0	0	0
1	42	41	44	45	40	43	36	39	37	38	73
68	85	31	35	33	34	29	27	30	28	32	51
81	1	0	0	0	0	0	0	0	0	0	0
1	16	12	10	100	1	0	0	0	0	0	0
1	70	101	83	1	0	0	0	0	0	0	0
5	56	1	0	0	0	0	0	0	0	5	56

Πίνακας 51: *Δρομολόγηση οχημάτων RC203*

52. Αποτελέσματα προβλήματος RC204

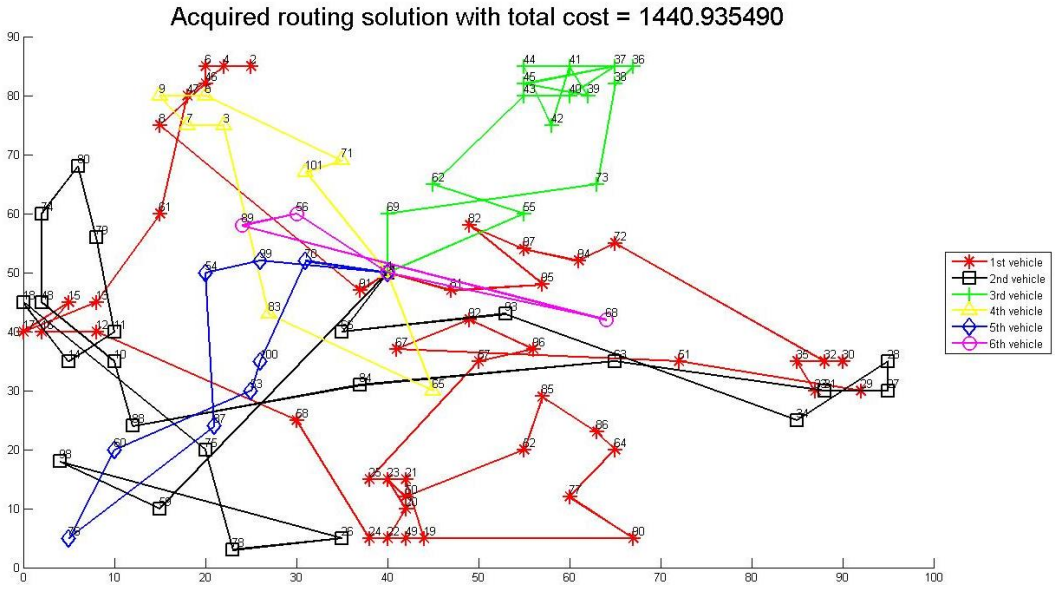


Σχήμα 52: *Δρομολόγηση οχημάτων RC204*

ΠΕΛΑΤΕΣ ΑΝΑ ΟΧΗΜΑ											
1	76	6	94	23	28	31	30	7	33	32	34
36	38	39	40	35	37	27	29	24	19	20	15
17	13	16	14	18	26	10	12	11	9	22	1
1	21	25	73	63	75	62	65	67	70	69	66
50	56	55	57	54	59	61	60	58	47	45	41
86	77	72	71	74	81	80	82	79	78	97	1
1	68	64	3	2	100	93	101	98	96	95	4
99	8	90	92	5	89	85	84	83	87	46	53
51	52	48	42	43	44	91	88	1	0	0	0
1	49	1	0	0	0	0	0	0	0	0	0

Πίνακας 52: *Δρομολόγηση οχημάτων RC204*

53. Αποτελέσματα προβλήματος RC205

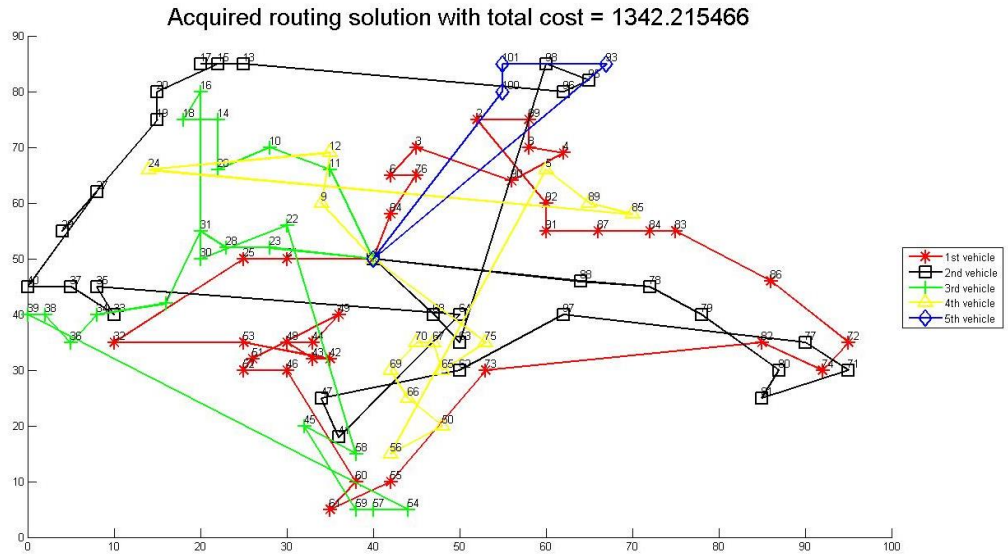


Σχήμα 53: *Δρομολόγηση οχημάτων RC205*

ΠΕΛΑΤΕΣ ΑΝΑ ΟΧΗΜΑ											
1	91	8	46	6	2	4	47	61	13	17	15
16	12	58	24	90	77	64	86	85	52	50	23
20	22	49	19	21	25	57	96	92	67	51	29
33	35	30	32	72	94	97	82	95	81	1	0
1	66	93	34	28	27	31	63	84	88	10	48
74	80	79	11	14	18	75	78	26	98	59	1
1	55	62	43	40	37	45	39	41	42	44	36
38	73	69	1	0	0	0	0	0	0	0	0
1	65	83	3	7	9	5	71	101	1	0	0
1	70	100	53	60	76	87	54	99	1	0	0
1	68	89	56	1	0	0	0	0	0	0	0

Πίνακας 53: *Δρομολόγηση οχημάτων RC205*

54. Αποτελέσματα προβλήματος RC206

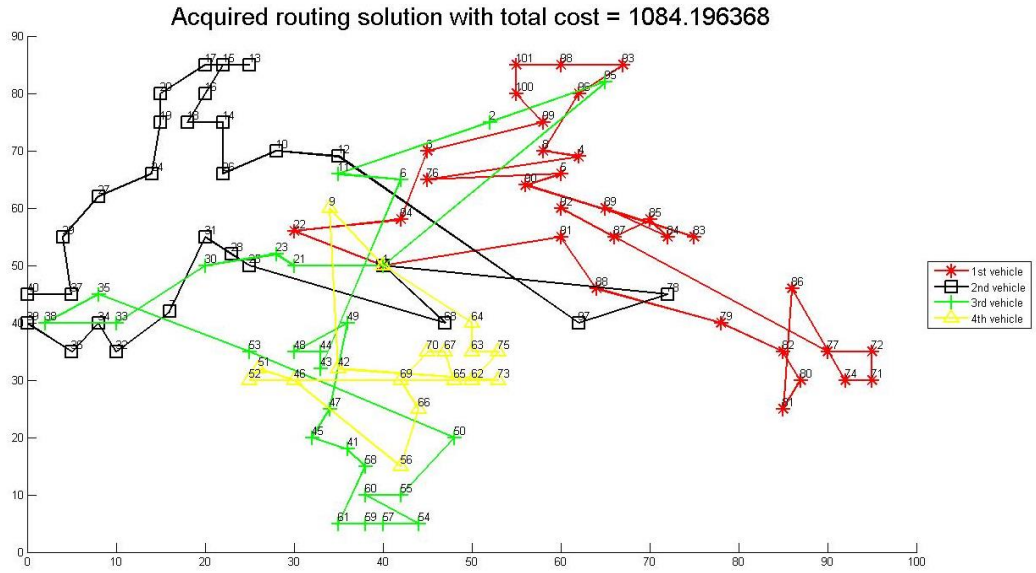


Σχήμα 54: *Δρομολόγηση οχημάτων RC206*

ΠΕΛΑΤΕΣ ΑΝΑ ΟΧΗΜΑ											
1	94	76	6	3	90	4	8	99	2	92	91
87	84	83	86	72	74	82	73	55	61	60	46
52	51	49	44	48	43	42	53	32	25	21	1
1	68	63	98	95	96	13	17	15	20	19	29
27	40	37	33	35	64	41	47	62	97	77	71
81	80	79	78	88	1	0	0	0	0	0	0
1	23	28	31	7	34	36	38	39	54	57	59
45	58	22	30	16	18	14	26	10	11	1	0
1	75	65	67	70	69	66	50	56	5	89	85
24	12	9	1	0	0	0	0	0	0	0	0
1	100	101	93	1	0	0	0	0	0	0	0

Πίνακας 54: *Δρομολόγηση οχημάτων RC206*

55. Αποτελέσματα προβλήματος RC207

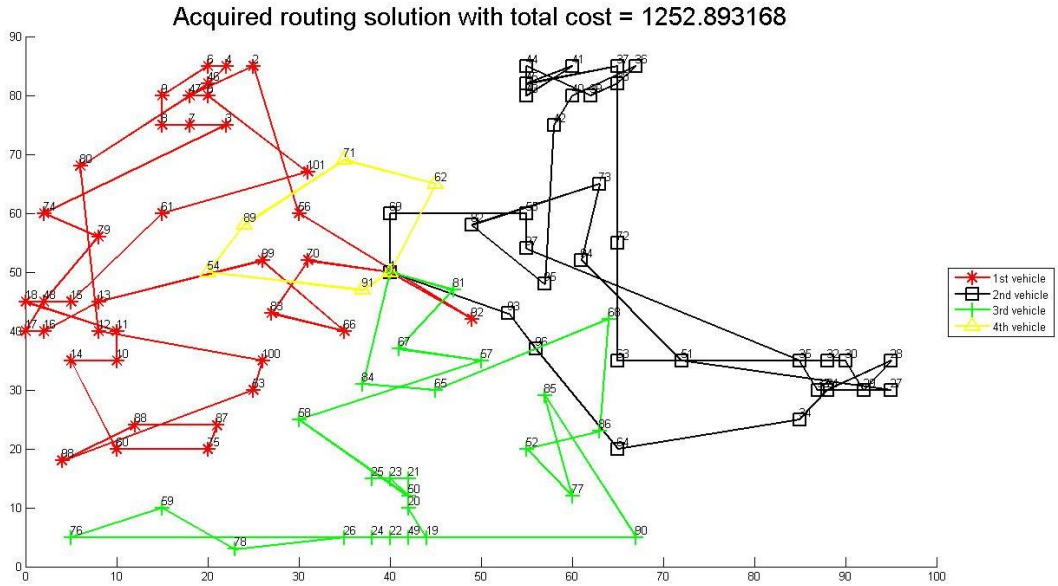


Σχήμα 55: *Δρομολόγηση οχημάτων RC207*

ΠΕΛΑΤΕΣ ΑΝΑ ΟΧΗΜΑ											
1	23	64	63	75	62	65	67	70	69	66	50
56	59	73	97	85	87	84	83	77	86	72	71
74	81	80	82	79	78	88	91	1	0	0	0
1	68	96	100	101	99	98	93	5	2	90	89
92	4	20	17	13	15	16	18	26	8	22	12
76	6	3	95	1	0	0	0	0	0	0	0
1	94	25	28	31	30	7	34	39	55	54	57
61	60	58	41	45	47	46	53	52	51	48	44
49	43	42	1	0	0	0	0	0	0	0	0
1	21	33	38	36	27	35	32	40	37	29	24
19	14	10	11	9	1	0	0	0	0	0	0

Πίνακας 55: *Δρομολόγηση οχημάτων RC207*

56. Αποτελέσματα προβλήματος RC208



Σχήμα 56: *Δρομολόγηση οχημάτων RC208*

ΠΕΛΑΤΕΣ ΑΝΑ ΟΧΗΜΑ											
1	70	83	66	99	13	16	17	48	79	74	3
7	8	9	6	4	46	80	12	100	53	98	88
87	75	60	14	10	11	18	15	61	101	5	47
2	56	92	1	0	0	0	0	0	0	0	0
1	93	96	64	34	31	28	29	30	32	63	72
37	45	41	43	44	39	38	36	40	42	95	82
73	94	51	27	33	35	97	55	69	1	0	0
1	84	65	68	86	52	77	85	90	24	26	78
59	76	22	49	19	20	21	25	23	50	58	57
67	81	1	0	0	0	0	0	0	0	0	0
1	91	54	89	71	62	1	0	0	0	0	0

Πίνακας 56: *Δρομολόγηση οχημάτων RC208*