# ΠΟΛΥΤΕΧΝΕΙΟ ΚΡΗΤΗΣ

ΣΧΟΛΗ ΜΗΧΑΝΙΚΩΝ ΠΑΡΑΓΩΓΗΣ & ΔΙΟΙΚΗΣΗΣ

Τσαμπουνάρη Ειρήνη

Επιβλέπων Καθηγητής: Μαρινάκης Ιωάννης

***Αλγόριθμος Βελτιστοποίησης Ζευγαρώματος Μελισσών για το Ανοιχτό Πρόβλημα Δρομολόγησης Οχημάτων***

*Χανιά, 2017*

# Ευχαριστίες

Θα ήθελα να εκφράσω τις ευχαριστίες και την ευγνωμοσύνη μου στον καθηγητή μου Μαρινάκη Ιωάννη για την πολύτιμη βοήθεια που μου παρείχε κατά την εκπόνηση της παρούσας διπλωματικής εργασίας καθώς και για την εμπιστοσύνη που μου έδειξε. Επιπλέον, θα ήθελα να ευχαριστήσω την οικογένεια μου για τη στήριξη που μου προσέφερε όλα αυτά τα χρόνια των σπουδών μου.

# Π ε ρ ι ε χ ό μ ε ν α

# Εισαγωγή

Σκοπός της τρέχουσας διπλωματικής εργασίας είναι η επίλυση του ***Ανοιχτού Προβλήματος Δρομολόγησης Οχημάτων (Vehicle Routing Problem- VRP)*** με τη χρήση του ***Αλγορίθμου Βελτιστοποίησης Ζευγαρώματος των Μελισσών (Honey Bees Mating Optimization - HBMO)***. Αρχικά, το πρώτο κεφάλαιο αναφέρεται σε βασικές έννοιες, όπως, η εφοδιαστική αλυσίδα, οι εξελικτικοί αλγόριθμοι και η νοημοσύνη σμήνους. Ακόμα, για την καλύτερη κατανόηση των επόμενων κεφαλαίων παρουσιάζεται αναλυτικά το πρόβλημα δρομολόγησης οχημάτων και η προέκταση αυτού, που αποτελεί το ανοιχτό πρόβλημα δρομολόγησης οχημάτων (OVRP). Τέλος, αναφέρεται στον Αλγόριθμο Βελτιστοποίησης Ζευγαρώματος των Μελισσών (HBMO) και γίνεται μία μικρή αναφορά στους Αλγόριθμους Βελτιστοποίησης Ζευγαρώματος Μπάμπουρων και Τεχνητής Αποικίας Μελισσών.

Εν συνεχεία, στο δεύτερο κεφάλαιο γίνεται αναφορά στην εφαρμογή του HBMO στο OVRP. Εξηγείται ο τρόπος δημιουργίας του αρχικού πληθυσμού, η εφαρμογή μεθόδων τοπικής αναζήτησης για τη βελτίωση των λύσεων και αναφέρεται το κόστος ως κριτήριο αξιολόγησης κάθε λύσης. Τέλος, παρουσιάζονται οι διαδικασίες των εξελικτικών αλγορίθμων που χρησιμοποιήθηκαν και η μέθοδος αποδοχής κατωφλίου.

Στο τρίτο και τελευταίο κεφάλαιο εμφανίζονται τα αποτελέσματα του κώδικα και σχολιάζονται σαφώς.

# Κεφάλαιο 1ο

# Βασικές Έννοιες

## 1.1 Ορισμός της Διαχείρισης Εφοδιαστικής Αλυσίδας

Ως Διαχείριση Εφοδιαστικής Αλυσίδας (SCM) ορίζεται ο σχεδιασμός, η οργάνωση, και ο συντονισμός όλων των δραστηριοτήτων της εφοδιαστικής αλυσίδας. Αναλυτικότερα, η Διαχείριση Εφοδιαστικής Αλυσίδας είναι ο συστηματικός, στρατηγικός συντονισμός των παραδοσιακών επιχειρηματικών λειτουργιών μέσα στην επιχείρηση και μεταξύ των επιχειρήσεων μέσα στην εφοδιαστική αλυσίδα, για τους σκοπούς βελτίωσης της μακροπρόθεσμης απόδοσης των μεμονωμένων επιχειρήσεων και της εφοδιαστικής αλυσίδας ως σύνολο. Με τον όρο Εφοδιαστική Αλυσίδα (ΕΑ) εννοούμε την ροή υλικών, πληροφοριών και υπηρεσιών από τους προμηθευτές πρώτων υλών μέσα από τα εργοστάσια και τις αποθήκες, στους τελικούς πελάτες.

Η Διαχείριση της Εφοδιαστικής Αλυσίδας (Supply Chain Management) αποτελεί ένα σχετικά νέο και πολλά υποσχόμενο τομέα της επιστήμης, με μεγάλη επίδραση στην αποτελεσματικότητα των σημερινών επιχειρήσεων στο ιδιαίτερα ανταγωνιστικό περιβάλλον της σύγχρονης επιχειρηματικότητας. Επιφέρει αποτελέσματα τόσο προς την κατεύθυνση της μείωσης του κόστους των επιχειρήσεων, όσο και προς την κατεύθυνση του βέλτιστου συντονισμού των διεργασιών της επιχείρησης που συνδέονται με τους προμηθευτές και τους διανομείς. Η ανάγκη που κυριαρχεί στις επιχειρήσεις ώστε να αποκτήσουν πλεονέκτημα απέναντι στους αντίπαλους-ανταγωνιστές τους και να καταφέρουν να τους ξεπεράσουν, είναι ουσιαστικά ο βασικότερος λόγος που η στρατηγική της εφοδιαστικής αλυσίδας κερδίζει ολοένα και περισσότερο έδαφος. Η στρατηγική της Διαχείρισης Εφοδιαστικής Αλυσίδας ουσιαστικά δημιουργεί τις προϋποθέσεις ώστε να βελτιωθεί η θέση της επιχείρησης σε σχέση με αυτήν των ανταγωνιστών της.

Η Εφοδιαστική αλυσίδα είναι ένας βασικός συντελεστής της εφαρμογής του μάνατζμεντ αφού ασχολείται με τον σχεδιασμό και την υλοποίηση των αποφάσεων και της

στρατηγικής ενός οργανισμού και λαμβάνει υπόψη μοντέλα και τεχνικές λήψης αποφάσεων, τη σύγχρονη τεχνολογία της πληροφορικής και των επικοινωνιών, και το επιχειρησιακό περιβάλλον. Για να ληφθεί και να υλοποιηθεί μια απόφαση λαμβάνονται υπόψη το εργασιακό κλίμα, οι ανθρώπινες και διοικητικές ικανότητες, η ύπαρξη και οργάνωση του πληροφοριακού συστήματος κ.ά. Δεν πρόκειται λοιπόν για μια μονοδιάστατη αλλά για μια πολυδιάστατη αντιμετώπιση.

Σε κάθε πρόβλημα προς επίλυση ακολουθείται μια σειρά βημάτων, που αποτελεί τη **μεθοδολογία της Εφοδιαστικής Αλυσίδας**, και παρουσιάζονται παρακάτω:

1. Εντοπισμός του προβλήματος και εφαρμογή μεθόδων ανάλυσης.
2. Μαθηματική ή συστημική μοντελοποίηση για την απεικόνιση του προβλήματος.
3. Επιλογή ή ανάπτυξη τεχνικών επίλυσης μέσω μαθηματικού προγραμματισμού, ευρετικών αλγορίθμων ή άλλων υπολογιστικών μεθόδων.
4. Υλοποίηση των παραπάνω σε κατάλληλη πλατφόρμα του πληροφοριακού συστήματος της εκάστοτε εταιρείας.
5. Σχεδιασμός και σύνθεση των πληροφοριακών συστημάτων που ενσωματώνουν τις τεχνικές επίλυσης και επιτρέπουν τη διεπαφή των μάνατζερ που πρέπει να λαμβάνουν τις αποφάσεις και των συστημάτων που περιέχουν τα δεδομένα και τις αλγοριθμικές προσεγγίσεις.

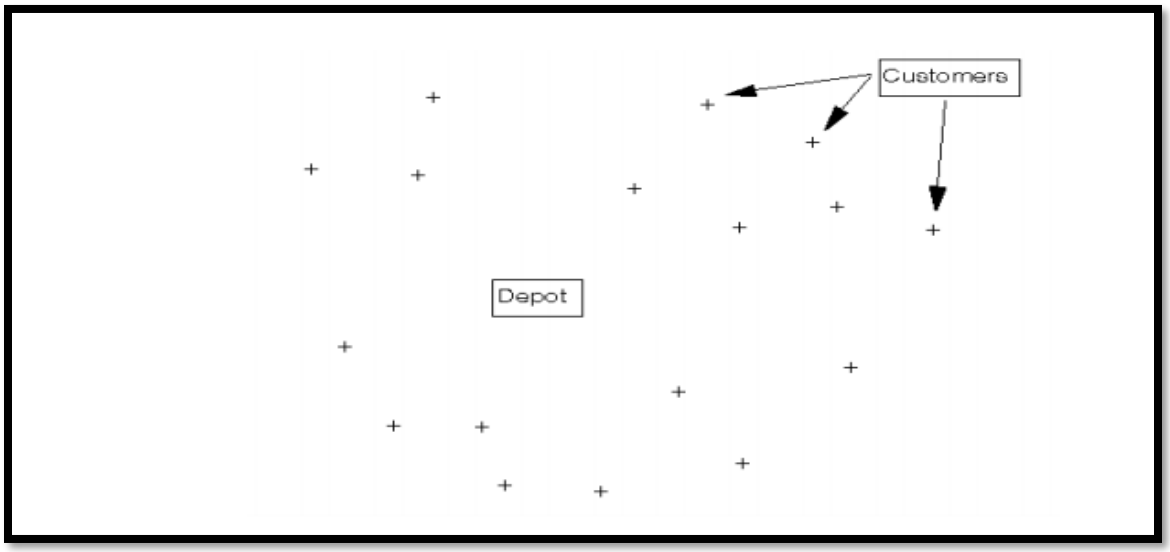
## *Ιστορική αναδρομή*

Η ανάγκη για συνεχή εφοδιασμό και για αποθέματα εμφανίστηκε από τα αρχαία χρόνια. Χαρακτηριστικό παράδειγμα είναι ότι η εκστρατεία του Μεγάλου Αλέξανδρου προς την Ασία , θα ήταν ανέφικτη χωρίς να διαθέτει το σωστό σύστημα εφοδιασμού. Επιπλέον άλλο ένα παράδειγμα, είναι αυτό της εποχής των Ρωμαϊκών χρόνων όταν ήταν αναγκαία η γρήγορη ανάπτυξη των λεγεώνων στα αυτοκρατορικά σύνορα για να αντιμετωπιστούν βαρβαρικές επιδρομές και έθετε απαιτήσεις στην ύπαρξη οδών και στην ταχεία μεταφορά διαταγών και ανδρών.

## 1.2 Vehicle Routing Problem(VRP): Το Πρόβλημα Δρομολόγησης Οχημάτων

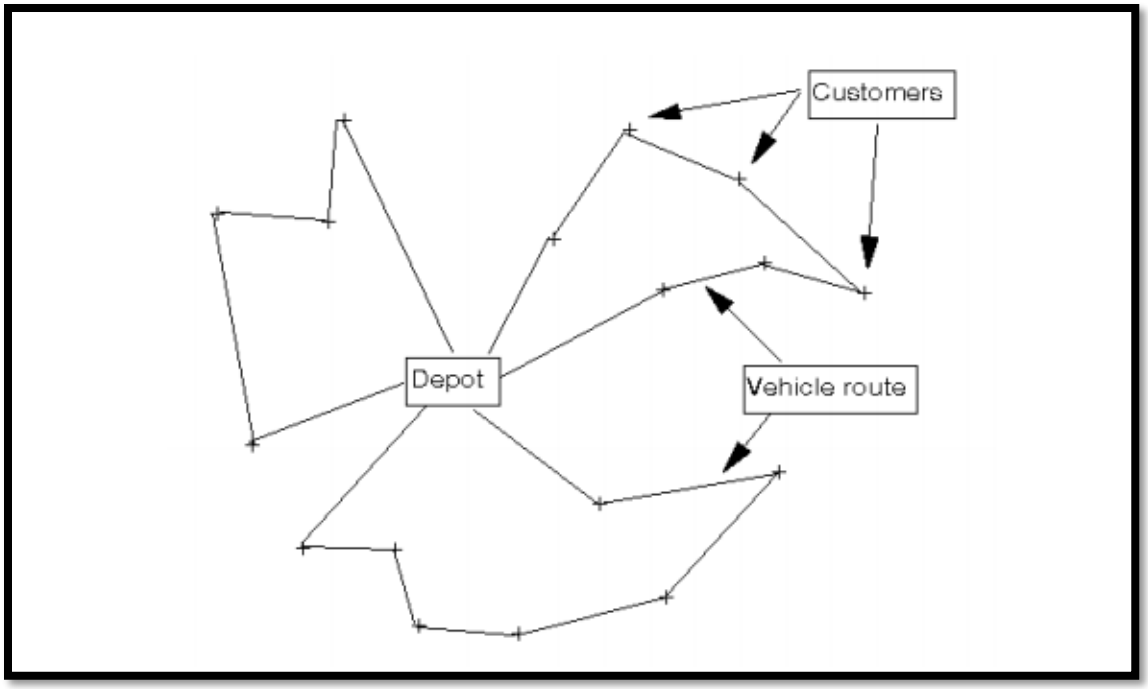
Το πρόβλημα δρομολόγησης οχημάτων αποτελεί ένα συνδυαστικό πρόβλημα βελτιστοποίησης ακέραιου προγραμματισμού που επιδιώκει την εξυπηρέτηση μεγάλου αριθμού πελατών με τη χρήση ενός στόλου οχημάτων και είναι ουσιαστικά επέκταση του προβλήματος του πλανόδιου πωλητή στις περιπτώσεις όπου ένα μόνο όχημα δε μπορεί να επισκεφθεί όλους τους πελάτες. Προτάθηκε από τους Dantzig και Ramser το 1959 και είναι ένα από τα πιο σημαντικά προβλήματα στον τομέα των μεταφορών και της διανομής. Στο πρόβλημα αυτό έχουμε τη διανομή διάφορων προϊόντων σε διαφορετικούς πελάτες με τη χρήση ενός στόλου οχημάτων με σκοπό την ικανοποίηση της εκάστοτε ζήτησης από τον κάθε πελάτη σε ένα πεπερασμένο χρονικό ορίζοντα. Στόχος είναι η ελαχιστοποίηση του κόστους διανομής των προϊόντων στους τελικούς πελάτες. Πολλές είναι οι παραλλαγές του προβλήματος, καθώς και οι μέθοδοι επίλυσης που έχουν προταθεί, αφού η εύρεση του ελάχιστου της συνάρτησης κόστους είναι μια σύνθετη διαδικασία.

Υποθέτουμε ένα πρόβλημα δρομολόγησης οχημάτων όπως παρουσιάζεται στο Σχήμα 1, όπου μια αποθήκη είναι περιτριγυρισμένη από πελάτες, οι οποίοι προμηθεύονται από εκεί προϊόντα. Η αποθήκη αναλαμβάνει να δημιουργήσει τις διαδρομές που θα ακολουθηθούν από τα οχήματα για τη διανομή των προϊόντων στους πελάτες.



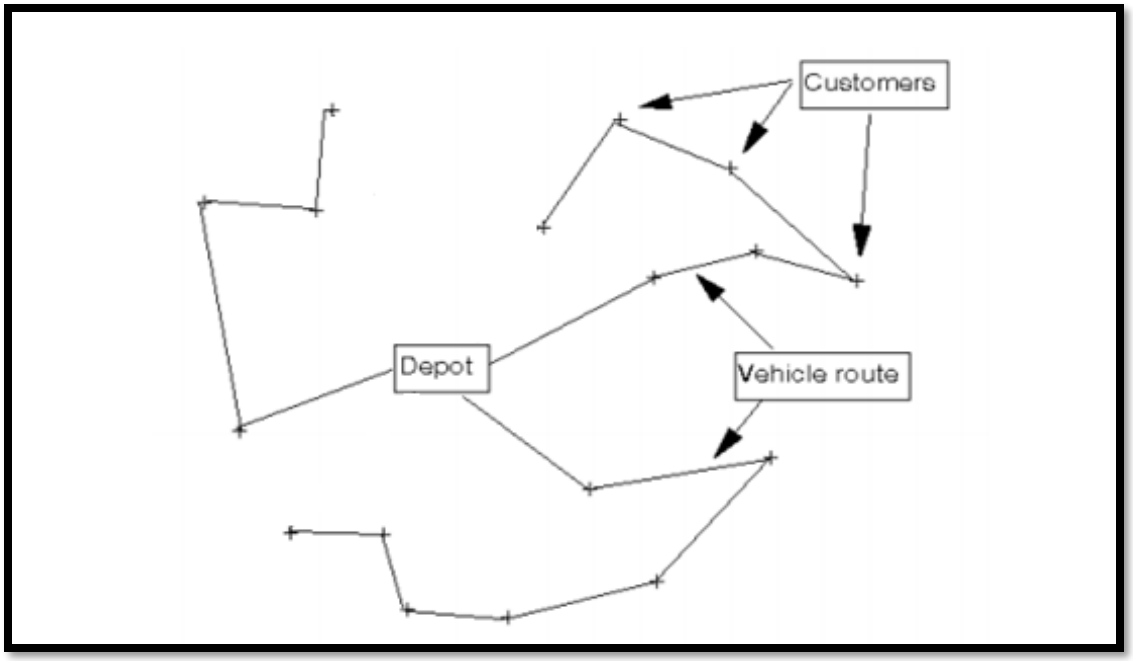

*Εικόνα 1: Αποθήκη (depot) και πελάτες (customers)*

Στο Σχήμα 2 βλέπουμε τις διαδρομές των οχημάτων από την αποθήκη προς τους πελάτες. Στη συγκεκριμένη περίπτωση, το VRP μετατρέπεται σε ένα πρόβλημα σχεδιασμού διαδρομών οχημάτων γνωστής χωρητικότητας, με στόχο την διανομή των προϊόντων στους πελάτες, σε συγκεκριμένο χρόνο, για να καλύπτεται η ζήτηση. Οι διαδρομές των οχημάτων σχεδιάζονται έτσι ώστε να ελαχιστοποιείται η συνολική απόσταση που θα διανύσουν τα οχήματα.




*Εικόνα 2: Η αποθήκη, οι πελάτες και οι κυκλικές διαδρομές οχημάτων με επιστροφή στην αποθήκη*

Στην τρέχουσα εργασία θα ασχοληθούμε με το **Ανοιχτό Πρόβλημα Δρομολόγησης Οχημάτων (Open Vehicle Routing Problem)**, το οποίο είναι μια παραλλαγή του παραπάνω, με διαφορά ότι κάθε όχημα δεν επιστρέφει στην αποθήκη μετά την τελευταία παράδοση στον τελευταίο πελάτη σε μια διαδρομή. Παρακάτω παρουσιάζεται το αντίστοιχο του σχήματος 1 και 2, σχήμα 3 του OVRP χωρίς επιστροφή στην αποθήκη.




*Εικόνα 3: Η αποθήκη, οι πελάτες και οι κυκλικές διαδρομές οχημάτων χωρίς επιστροφή στην αποθήκη*

Η πρακτική εφαρμογή αυτού του προβλήματος παρουσιάζεται σε εταιρείες που νοικιάζουν τα οχήματα, τα οποία θα εκτελέσουν τις διαδρομές, και δεν επιστρέφουν στην αποθήκη διότι μετά την τελευταία εξυπηρέτηση τα οχήματα αποδεσμεύονται από την εταιρεία.

## *Ορισμός του προβλήματος*

Ένα βασικό πρόβλημα δρομολόγησης οχημάτων αποτελείται από ένα σύνολο κόμβων τα οποία αντιπροσωπεύουν τους πελάτες μιας επιχείρησης και από τη ζήτηση του κάθε πελάτη αναλόγως των αναγκών του. Αυτή η ζήτηση πρέπει να ικανοποιηθεί από ένα σύνολο οχημάτων ανεξαρτήτως χρονικών περιορισμών αλλά και περιορισμών προτεραιότητας εξυπηρέτησης.

## *Χαρακτηριστικά του προβλήματος*

- Κάθε κύκλος περνάει από την αποθήκη
- Κάθε πελάτης επισκέπτεται από ένα μόνο κύκλο
- Το άθροισμα της ζήτησης των κόμβων που επισκέπτονται από έναν κύκλο δεν ξεπερνάει τη χωρητικότητα του οχήματος.

## *Στόχος του προβλήματος*

Ο στόχος του προβλήματος είναι η κατασκευή ενός συνόλου διαδρομών (μία για κάθε όχημα), χαμηλού κόστους, για τη μεγιστοποίηση του κέρδους της επιχείρησης. Αυτό επιτυγχάνεται με την ελαχιστοποίηση του αριθμού των οχημάτων.

## *Μαθηματική Περιγραφή του μοντέλου*

Υπάρχουν πολλοί τρόποι προτυποποίησης του συγκεκριμένου προβλήματος αλλά εδώ θα παρουσιαστεί μία από τις πιο κλασικές που έχει παρουσιαστεί από τους Fisher και `

$$\mathrm{X}ijk=\begin{cases}1, & \alpha\nu\ \tau o\ \acute{o}\chi\eta\mu\alpha\ k\ \epsilon\pi\iota\sigma\kappa\acute{\epsilon}\pi\tau\epsilon\tau\alpha\iota\ \tau o\nu\ \pi\epsilon\lambda\acute{\alpha}\tau\eta\ j\ \alpha\mu\acute{\epsilon}\sigma\omega\varsigma\ \mu\epsilon\tau\acute{\alpha}\ \tau o\nu\ \pi\epsilon\lambda\acute{\alpha}\tau\eta\ i\\ 0, & \alpha\lambda\lambda\iota\acute{\omega}\varsigma\end{cases}$$

$$Yjk=\begin{cases}1, & \alpha\nu\ o\ \pi\epsilon\lambda\acute{\alpha}\tau\eta\varsigma\ i\ \epsilon\pi\iota\sigma\kappa\acute{\epsilon}\pi\tau\epsilon\tau\alpha\iota\ \alpha\pi\acute{o}\ \tau o\ \acute{o}\chi\eta\mu\alpha\ k\\ 0, & \alpha\lambda\lambda\iota\acute{\omega}\varsigma\end{cases}$$

Το βασικό πρόβλημα δρομολόγησης οχημάτων μπορεί να καθοριστεί ως εξής:

$$\boldsymbol{min}\sum_{i,j}Cij\sum_{k}Xijk \qquad 1.1$$

Υπό των περιορισμών:

$$\sum_{k}Yik=\begin{cases}1, & i=2,\dots,n\\ m, & i=1\end{cases} \qquad 1.2$$

$$\sum\ q_i y_{ij}\le Q_k \qquad k=1,\dots,m \qquad 1.3$$

$$\sum_{i}Xijk=\sum_{j}Xjik=Yik \qquad i=1,\dots,n \quad k=1,\dots,m \qquad 1.4$$

$$\sum_{i,j\in S}Xijk\le|S|-1, \qquad \text{για όλα τα } S \subseteq \{2,\dots n\} \quad k=1,\dots,m$$

$$y_{ik} \in \{0,1\}, \quad i=1,\ldots,n, \quad k=1,\ldots,m$$

$$x_{ijk} \in \{0,1\}, \quad i,j=1,\ldots,n, \quad k=1,\ldots,m$$

Ο περιορισμός 1.2 δείχνει ότι κάθε πελάτης εκχωρείται σε ένα μόνο όχημα, εκτός από την αποθήκη που την επισκέπτονται όλα τα οχήματα. Ο περιορισμός 1.3 είναι αυτός της χωρητικότητας των οχημάτων και ο περιορισμός 1.4 δείχνει ότι ένα όχημα που επισκέπτεται ένα πελάτη φεύγει από τον πελάτη.

### 1.2.1 Άλλες παραλλαγές του προβλήματος VRP

Το πρόβλημα δρομολόγησης οχημάτων μπορεί να εμφανιστεί σε διάφορες παραλλαγές και παρουσιάζονται ακολούθως :

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

Το συγκεκριμένο πρόβλημα αφορά την περίπτωση όπου η επίσκεψη κάθε πελάτη πρέπει να γίνει μέσα στα πλαίσια ενός προκαθορισμένου χρονικού διαστήματος. Εδώ υπάρχει ένα σύνολο πελατών που βρίσκονται σε μια περιοχή διεσπαρμένοι και πρέπει να εξυπηρετηθούν από ένα στόλο οχημάτων που αρχικά βρίσκεται στην αποθήκη. Κάθε πελάτης εξυπηρετείται σε ένα καθορισμένο χρονικό διάστημα το οποίο αποτελεί και το χρονικό παράθυρο (time window). Ουσιαστικά, το πρόβλημα αυτό είναι μία επέκταση του προβλήματος VRP που ισχύουν οι ίδιοι περιορισμοί (χωρητικότητας) και απλά προστίθεται ένας ακόμα, αυτός των χρονικών παραθύρων. Τα επιπλέον δεδομένα που δίνονται για το πρόβλημα είναι η χρονική στιγμή που φεύγουν τα οχήματα από την αποθήκη, ο χρόνος ταξιδιού, μέχρι να φτάσει το όχημα στον κάθε πελάτη και ο χρόνος εξυπηρέτησης για κάθε πελάτη. Η εξυπηρέτηση πρέπει να ξεκινήσει μέσα στο χρονικό παράθυρο που μπορεί ο πελάτης να εξυπηρετηθεί και το όχημα μπορεί να παραμείνει στην τοποθεσία του πελάτη για συγκεκριμένο χρόνο.

- ***Η ύπαρξη πολλαπλών αποθηκών (Multidepot Vehicle Routing)***

Σε αυτήν εδώ την παραλλαγή χρησιμοποιούνται παραπάνω από μία αποθήκες και δύο είναι οι τρόποι επίλυσης. Είτε η κάθε μία από τις αποθήκες να έχει τα δικά της οχήματα, το οποίο αποτελεί έναν αριθμό από απλά VRPs, είτε ένα όχημα ξεκινάει από μία αποθήκη και τερματίζει σε μια άλλη ή σταματά ενδιάμεσα σε κάποια άλλη αποθήκη για να φορτώσει.

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

Άλλη μια επέκταση του βασικού προβλήματος VRP όπου υπάρχουν δύο υποσύνολα πελατών. Το πρώτο αφορά τους πελάτες που απαιτούν τη διανομή κάποιας ποσότητας προϊόντων (linehauls customers) και το δεύτερο υποσύνολο αφορά πελάτες που ο καθένας απαιτεί μια ποσότητα του προϊόντος να περισυλλεχθεί από αυτόν (with backhauls customers). Σε αυτό το πρόβλημα υπάρχει ένας πολύ βασικός περιορισμός ανάμεσα στους δύο τύπους πελατών: οποτεδήποτε εξυπηρετούνται από μία διαδρομή πελάτες και των δυο τύπων όλοι οι πελάτες του πρώτου τύπου πρέπει να εξυπηρετηθούν πριν από τους πελάτες του δεύτερου τύπου. Τα χαρακτηριστικά αυτού του προβλήματος είναι τα εξής:

1. Κάθε όχημα ακολουθεί μόνο μια διαδρομή που ξεκινά και καταλήγει στην αποθήκη.
2. Κάθε πελάτης επισκέπτεται από ένα μόνο κύκλο.
3. Για κάθε διαδρομή η συνολική φόρτωση και για τα δύο είδη πελατών δε ξεπερνάει τη χωρητικότητα των οχημάτων.
4. Το μέγιστο μήκος της κάθε διαδρομής δεν πρέπει να ξεπερνάει τη συνολική απόσταση που μπορεί να διανύσει το κάθε όχημα.
5. Σε κάθε διαδρομή οι πελάτες του δεύτερου τύπου επισκέπτονται έπειτα από τους πελάτες του πρώτου τύπου.
6. Διαδρομές που περιλαμβάνουν πελάτες μόνο του δεύτερου τύπου δεν επιτρέπονται.

## 1.3 Τι είναι οι Εξελικτικοί Αλγόριθμοι

Τα τελευταία τριάντα χρόνια, παρατηρείται ένα συνεχώς αυξανόμενο ενδιαφέρον για την ανάπτυξη συστημάτων επίλυσης προβλημάτων βασισμένων στις αρχές της φυσικής εξέλιξης. Τα συστήματα αυτά λειτουργούν διαιρώντας έναν πληθυσμό κωδικοποιημένων πιθανών λύσεων του προβλήματος που προσπαθούμε να επιλύσουμε, και εφαρμόζοντας πάνω σε αυτόν διάφορες διαδικασίες εμπνευσμένες από τη βιολογική εξέλιξη. Έτσι, περνώντας από γενιά σε γενιά, τα συστήματα αυτά δημιουργούν συνεχώς νέους πληθυσμούς πιθανών λύσεων εξελίσσοντας τους προηγούμενους πληθυσμούς.

Οι γενετικοί αλγόριθμοι είναι ένα παράδειγμα τέτοιου συστήματος που μαζί με τον εξελικτικό προγραμματισμό, τις στρατηγικές εξέλιξης, τα συστήματα ταξινόμησης και το γενετικό προγραμματισμό αποτελούν μια κατηγορία συστημάτων επίλυσης προβλημάτων που είναι ευρύτερα γνωστή με τον όρο **Εξελικτικοί Αλγόριθμοι**.

Οι ΕΑ είναι αλγόριθμοι ανίχνευσης-αναζήτησης, βασισμένοι στη μηχανική της φυσικής επιλογής και της φυσικής γενετικής. Οι ΕΑ μιμούνται τις διαδικασίες βιολογικής εξέλιξης με την υλοποίηση των ιδεών της φυσικής επιλογής και της επικράτησης του ισχυρότερου, έτσι ώστε να παρέχουν αποτελεσματικές λύσεις σε προβλήματα αναζήτησης και βελτιστοποίησης. Υπάρχουν διάφορα εξελικτικά υπολογιστικά μοντέλα τα οποία όμως βασίζονται στις ίδιες αρχές, δηλαδή στις αρχές προσομοίωσης της εξέλιξης ατομικών δομών μέσω των διαδικασιών της επιλογής και της αναπαραγωγής.

Τα χαρακτηριστικά των γενετικών αλγορίθμων είναι τα εξής :

1. Κατά τη διάρκεια της αναζήτησης της καλύτερης λύσης χρησιμοποιούν πληροφορίες από ένα πληθυσμό λύσεων που ονομάζονται **άτομα**.
2. Είναι επαναληπτική διαδικασία που διατηρεί το μέγεθος του πληθυσμού σταθερό σε κάθε επανάληψη που ονομάζεται **γενιά**.
3. Δημιουργούν μια νέα λύση ταιριάζοντας δυο λύσεις. Αυτό γίνεται με τη **διασταύρωση** και με τη **μετάλλαξη**. Η διασταύρωση παίρνει δύο άτομα που ονομάζονται **γονείς** και παράγει με την ανταλλαγή τμημάτων των γονέων δύο νέα άτομα που ονομάζονται **απόγονοι**.
4. Κάθε άτομο στον πληθυσμό αντιπροσωπεύει μια υποψήφια λύση.

## *Το παράδειγμα των λαγών*

Ας δούμε τους λαγούς και τον τρόπο που αναπαράγονται και εξελίσσονται από γενιά σε γενιά. Έστω ότι αρχίζουμε να παρατηρούμε ένα συγκεκριμένο πληθυσμό από λαγούς σε ένα οικοσύστημα. Όπως είναι φυσικό, κάποιοι από αυτούς θα είναι πιο γρήγοροι και πιο εύστροφοι από άλλους. Αυτοί οι λαγοί έχουν περισσότερες πιθανότητες να επιβιώσουν στο φυσικό τους περιβάλλον (δηλαδή να εξασφαλίζουν τροφή και να ξεφεύγουν από τα διάφορα αρπακτικά που τους καταδιώκουν, όπως τις αλεπούδες) από ότι κάποιοι πιο αργοί ή λιγότερο έξυπνοι λαγοί. Φυσικά δεν είναι λίγοι οι αργοί ή λιγότερο έξυπνοι λαγοί που καταφέρνουν να επιβιώσουν εξαιτίας της τύχης ή άλλων παραγόντων. Όλοι αυτοί οι λαγοί, που καταφέρνουν να επιβιώσουν, θα αρχίσουν την παραγωγή της επόμενης γενιάς τους, μιας γενιάς που θα συνδυάζει με διάφορους τρόπους όλα τα χαρακτηριστικά των μελών της προηγούμενης. Έτσι, μερικοί αργοί λαγοί θα αναμειχθούν με κάποιους γρήγορους, κάποιοι γρήγοροι με άλλους γρήγορους και ούτω καθεξής, δημιουργώντας έτσι σταδιακά έναν πληθυσμό που απαρτίζεται από κάποιους λαγούς που θα είναι εξυπνότεροι και ταχύτεροι από τους προγόνους τους και από κάποιους άλλους που θα είναι πιο αργοί και λιγότερο εύστροφοι, δηλαδή τα χαρακτηριστικά (γονίδια) θα διαιωνίζονται από γενιά σε γενιά άλλα σε μικρότερο κι άλλα σε μεγαλύτερο βαθμό.

## 1.4 Νοημοσύνη Σμήνους (swarm intelligence)

Η **νοημοσύνη σμήνους (swarm intelligence)** είναι η συλλογική συμπεριφορά μη κατανεμημένων, αυτοοργανωμένων φυσικών ή τεχνητών συστημάτων. Η ιδέα εφαρμόζεται στον τομέα της τεχνητής νοημοσύνης. Ο όρος εισήχθη από τους Gerardo Beni και Jing Wang το 1989. Η νοημοσύνη σμήνους είναι μια ιδιότητα συστημάτων που επιδεικνύουν συλλογικά ευφυή συμπεριφορά. Μέσα από απλούς κανόνες εμφανίζονται φαινόμενα (σμήνη πουλιών, κοπάδια ψαριών, άναμμα πυγολαμπίδων, τα μυρμήγκια υπολογίζουν βέλτιστες διαδρομές προς την τροφή τους, μέλισσες να ενημερώνουν τη κυψέλη για νέκταρ) τα οποία οδηγούνται μέσα από τη **συλλογική συμπεριφορά και ευφυΐα**.

Τα συστήματα νοημοσύνης σμήνους κατά κανόνα αποτελούνται από έναν πληθυσμό απλών, αυτόνομων πρακτόρων ή διαμεσολαβητών (boids) που αλληλοεπιδρούν μεταξύ τους και με το περιβάλλον τους σε τοπικό επίπεδο. Τα συστήματα είναι εμπνευσμένα από τη φύση και ειδικότερα από τα βιολογικά συστήματα. Οι πράκτορες ακολουθούν πολύ απλούς κανόνες, και παρόλο που δεν υπάρχει καμία συγκεντρωτική δομή ελέγχου να υπαγορεύει πως πρέπει να συμπεριφέρονται οι πράκτορες, τοπικές - και έως ένα βαθμό τυχαίες - αλληλεπιδράσεις μεταξύ τέτοιων πρακτόρων οδηγούν στην εμφάνιση μιας ευφυούς, καθολικής συμπεριφοράς, άγνωστης στους αυτόνομους πράκτορες. Φυσικά παραδείγματα της νοημοσύνης σμήνους περιλαμβάνουν τις αποικίες μυρμηγκιών, τη βακτηριδιακή ανάπτυξη, τις αγέλες ή τα κοπάδια ζώων και τα κοπάδια ψαριών.

Ένα άλλο παράδειγμα νοημοσύνης σμήνους είναι τα **σμήνη πουλιών** που κινούνται σαν ένα σώμα παρά του ότι αποτελούνται από πολλές ανεξάρτητες οντότητες. Αν παρατηρήσει κανείς τα πουλιά θα προσέξει ότι κινούνται σαν μια ομάδα με ένα κοινό προσανατολισμό. Αυτό το σύνολο που προκύπτει είναι μέσα από τη συνολική συμπεριφορά των πουλιών. Ένα πουλί βλέπει γύρω το περιβάλλον του και αναλόγως συνυπολογίζει σε σχέση με τα άλλα πουλιά για να πάρει τις δικές του αποφάσεις. Ο νόμος του Couzin αναφέρει τρεις αρχές: αν ένα πουλί βρίσκεται μακριά από τα άλλα τότε καταλαβαίνει ότι πρέπει να αναπτύξει ταχύτητα αν όμως πλησιάζει πολύ σε ένα άλλο τότε καταλαβαίνει ότι θα συγκρουστεί και μειώνει την ταχύτητα του, να υπάρχει

δηλαδή στο σμήνος μια συνοχή, αν βρει εμπόδιο το αποφεύγει, διαχωρίζεται από το σύνολο και ξανασμίγουν και τρίτο πρέπει να έχουν μια κοινή κατεύθυνση να ακολουθούν μια συγκεκριμένη πορεία. Αρχή της συλλογικής συμπεριφοράς είναι ότι τα πουλιά πάντα προσπαθούν να διατηρήσουν μια ελάχιστη απόσταση μεταξύ των ίδιων αλλά και των άλλων. Αυτός ο κανόνας έχει υψηλή προτεραιότητα και αντιστοιχεί στη συμπεριφορά των ζώων στη φύση.

## *Εφαρμογές*

Οι τεχνικές που βασίζονται στην νοημοσύνη σμήνους μπορούν να χρησιμοποιηθούν σε διάφορες εφαρμογές. Ο στρατός των ΗΠΑ ερευνά τεχνικές σμήνους για τον έλεγχο τηλεκατευθυνόμενων οχημάτων. Ο Ευρωπαϊκός Οργανισμός Διαστήματος μελετά ένα σμήνος σε τροχιά για την αυτοσυναρμολόγηση και τη συμβολομετρία. Η NASA ερευνά τη χρήση της τεχνολογίας σμήνους για χαρτογράφηση των πλανητών. Σε μια μελέτη του 1992 από τους Anthony M. Lewis και George A. Bekey συζητείται το ενδεχόμενο χρησιμοποίησης νοημοσύνης σμήνους για τον έλεγχο nanobots μέσα στο ανθρώπινο σώμα, με σκοπό τη θανάτωση καρκινικών όγκων. Αντίθετα οι Al-Rifaie και Aber έχουν χρησιμοποιήσει τον αλγόριθμο SDS (Stochastic Diffusion Search) για να βοηθήσουν στον εντοπισμό όγκων. Η νοημοσύνη σμήνους έχει επίσης εφαρμογές στην εξόρυξη δεδομένων.

## 1.5 Honey Bees Mating Optimization (HBMO): Ο Αλγόριθμος Βελτιστοποίησης Ζευγαρώματος Μελισσών

Ο Αλγόριθμος Βελτιστοποίησης Ζευγαρώματος Μελισσών (HBMO) προτάθηκε από τον Abass το 2001 και αναπαριστά τη διαδικασία ζευγαρώματος της βασίλισσας μέλισσας της κυψέλης.

### *Χαρακτηριστικά των μελισσών*

Οι μέλισσες είναι κοινωνικά έντομα και δουλεύουν ομαδικά για να κατασκευάζουν τις κυψέλες όπου ζουν μέσα. Μια αποικία από μέλισσες συνήθως αποτελείται από μία βασίλισσα, από μηδέν έως μερικές χιλιάδες κηφήνες (εξαρτάται από τη χρονική στιγμή κατά τη διάρκεια της σεζόν) και συνήθως από 10000-60000 εργάτριες. Η δουλειά της βασίλισσας είναι να γεννάει αυγά, ενώ μπορεί να γεννήσει μέχρι 1500 αυγά την ημέρα και ζει συνήθως από 5 μέχρι 6 χρόνια. Μόνο η βασίλισσα μπορεί να φάει βασιλικό πολτό, πράγμα που την κάνει μεγαλύτερη από όλες τις άλλες μέλισσες στην κυψέλη. Οι κηφήνες παίζουν το ρόλο του 'αρσενικού' στις κυψέλες αφού ο κύριος ρόλος τους είναι να γονιμοποιούν τη βασίλισσα. Στο τέλος της σεζόν οι κηφήνες φεύγουν από την κυψέλη για να πεθάνουν. Οι κηφήνες ποτέ δεν ζουν πάνω από 6 μήνες. Οι εργάτριες κάνουν όλες τις δουλειές που χρειάζεται για να λειτουργήσει η κυψέλη. Στην ουσία φυλούν την κυψέλη και είναι στείρα θηλυκά. Όταν είναι νέες μένουν στην κυψέλη και κάνουν όλες τις κατασκευαστικές δουλειές, ακόμα και τη φροντίδα των νεογνών. Οι μεγαλύτερες εργάτριες βγαίνουν έξω από την κυψέλη και ψάχνουν να βρουν νέκταρ, νερό, γύρη και γενικά οτιδήποτε χρειάζεται η κυψέλη. Οι εργάτριες που θα γεννηθούν στην αρχή της σεζόν θα ζήσουν περίπου 6 βδομάδες ενώ αυτές που θα γεννηθούν το φθινόπωρο θα ζήσουν μέχρι την επόμενη άνοιξη.

### *Διαδικασία ζευγαρώματος*

Στη διαδικασία ζευγαρώματος των μελισσών, η βασίλισσα ζευγαρώνει κατά τη διάρκεια των 'πτήσεων ζευγαρώματος' αρκετά μακριά από την κυψέλη. Η βασίλισσα ξεκινάει με ένα χορό που πραγματοποιείται στην κυψέλη και στη συνέχεια πραγματοποιεί μια 'πτήση ζευγαρώματος' κατά την οποία οι κηφήνες την ακολουθούν και ζευγαρώνουν μαζί της στον αέρα. Η βασίλισσα καταδιώκεται από ένα μεγάλο σμήνος από κηφήνες. Η γονιμοποίηση τελειώνει με το θάνατο του κηφήνα και με τη βασίλισσα να έχει πάρει το

σημάδι ότι ζευγάρωσε με τον κηφήνα. Η βασίλισσα μπορεί να ζευγαρώσει πολλές φορές κατά τη διάρκεια μιας πτήσης αλλά ο κηφήνας μονάχα μία φορά. Σε κάθε ζευγάρωμα με διαφορετικό κηφήνα, το σπέρμα του κηφήνα αποθηκεύεται στην σπερματοθήκη της βασίλισσας για να δημιουργήσει το γενετικό υλικό της αποικίας. Κάθε φορά που η βασίλισσα γεννάει γονιμοποιημένα αυγά χρησιμοποιεί μια μείξη από το σπέρμα που έχει αποθηκεύσει από όλους τους κηφήνες στη σπερματοθήκη της.

Μπορεί να θεωρηθεί ότι πρόκειται απλά για ένα γενετικό αλγόριθμο με τη χρήση ενός πολύ ισχυρού υπερ-γονέα (τη βασίλισσα). Αλλά φυσικά αυτή η μέθοδος δεν είναι απλά ένας γενετικός αλγόριθμος. Οι δύο βασικές διαφορές είναι ότι η βασίλισσα κινείται τυχαία στο χώρο και επιλέγει, όπως θα δούμε και παρακάτω, βάσει της ταχύτητάς της και της ενέργειάς της με ποιον κηφήνα θα ζευγαρώσει. Ακόμα και αν η βασίλισσα ζευγαρώσει με ένα κηφήνα, δεν δημιουργεί κατευθείαν ένα νεογνό, αλλά αποθηκεύει το γονότυπό του (μέρος της λύσης του) στη σπερματοθήκη της και το κάθε νεογνό (απόγονος) δημιουργείται μόνο όταν η 'πτήση ζευγαρώματος' έχει ολοκληρωθεί. Άλλη μία διαφορά από τους γενετικούς αλγόριθμους είναι ότι οι απόγονοι δεν δημιουργούνται από μία λύση (έναν κηφήνα) αλλά παίρνουν χαρακτηριστικά από πολλούς κηφήνες και από τη βασίλισσα.

## *Υλοποίηση Αλγορίθμου*

Το πρώτο βήμα που πρέπει να γίνει για την υλοποίηση του Αλγορίθμου Βελτιστοποίησης Ζευγαρώματος Μελισσών είναι να δημιουργηθεί ο πληθυσμός των μελισσών ο οποίος αποτελεί την αρχική κυψέλη. Η καλύτερη μέλισσα αποτελεί τη βασίλισσα και όλες οι υπόλοιπες τους κηφήνες, ενώ οι μέλισσες εργάτριες είναι μέθοδοι τοπικής αναζήτησης. Θα πρέπει να οριστεί ένας αριθμός ο οποίος θα ορίζει το μέγεθος της σπερματοθήκης της βασίλισσας. Η 'πτήση ζευγαρώματος' τελειώνει όταν γεμίσει η σπερματοθήκη της βασίλισσας. Επίσης θα πρέπει να οριστεί ο αριθμός των βασιλισσών και ο αριθμός των νεογνών. Όταν ξεκινάει η πτήση της βασίλισσας στην ουσία η βασίλισσα κινείται στο χώρο λύσεων με κάποια ταχύτητα και ζευγαρώνει με κάποια πιθανότητα με έναν κηφήνα. Στην αρχή της πτήσης η βασίλισσα έχει μια ενέργεια και όταν γυρίσει στην κυψέλη αυτή η ενέργεια βρίσκεται στο διάστημα από μηδέν έως το μέγεθος της σπερματοθήκης. Παρακάτω φαίνεται ο τύπος για την πιθανότητα να ζευγαρώσει ένας κηφήνας με τη βασίλισσα.

$$Prob(D) = e^{\frac{-\Delta(f)}{speed(t)}}$$

όπου Prob (D) αντιπροσωπεύει την πιθανότητα να ζευγαρώσει ο κηφήνας D με τη βασίλισσα, Δ(f) είναι η διαφορά στην τιμή της συνάρτησης ποιότητας της βασίλισσας και του κηφήνα D και Speed(t) είναι η ταχύτητα της βασίλισσας τη χρονική στιγμή t. Η πιθανότητα του ζευγαρώματος είναι υψηλή όταν η βασίλισσα βρίσκεται στο ξεκίνημα της 'πτήσης ζευγαρώματος' ή όταν η συνάρτηση ποιότητας του κηφήνα είναι περίπου τόσο καλή όσο της βασίλισσας. Μετά από κάθε μετάβαση της βασίλισσας στο χώρο, η ταχύτητά της και η ενέργειά της μειώνονται βάσει των τύπων.

$$Speed(t+1) = a \times Speed(t)$$

$$energy(t+1) = energy(t) - step$$

όπου a είναι ένας παράγοντας στο διάστημα (0,1) και αντιπροσωπεύει το ποσό που μειώνεται η ταχύτητα σε κάθε μετάβαση. Αν το a πάρει πολύ μεγάλη τιμή η ταχύτητα μειώνεται αργά, αντίθετα, αν το a πάρει μικρή τιμή η ταχύτητα μειώνεται πολύ γρήγορα. Το step αντιπροσωπεύει το ποσό που μειώνεται η ενέργεια σε κάθε μετάβαση. Υπάρχουν υλοποιήσεις του αλγορίθμου που αντί για την παραπάνω εξίσωση ενέργειας χρησιμοποιούν (Magda Marinakis και συνεργάτες, 2010 και Magda Marinakis και συνεργάτες, 2008) ένα τύπο της μορφής:

$$energy(t+1) = a \times energy(t)$$

ώστε η μείωση στην ταχύτητα και στην ενέργεια να είναι ανάλογες. Αρχικά, η ταχύτητα και η ενέργεια της βασίλισσας αρχικοποιούνται τυχαία. Ένας αριθμός από 'πτήσεις ζευγαρώματος' πραγματοποιούνται. Στο ξεκίνημα της πτήσης η ταχύτητα είναι συνήθως μεγάλη και έτσι η βασίλισσα κάνει πολύ μεγάλα βήματα στο χώρο. Καθώς η ενέργεια της βασίλισσας μειώνεται, η ταχύτητά της μειώνεται, πράγμα που έχει ως αποτέλεσμα οι δυνατότητες αναζήτησης που έχει η βασίλισσα να μειώνονται. Ο γονότυπος των κηφήνων που έχει αποθηκευτεί στη σπερματοθήκη της βασίλισσας διασταυρώνεται με της βασίλισσας με τη χρήση οποιουδήποτε τελεστή διασταύρωσης που έχει την ικανότητα να μπορεί να χρησιμοποιήσει παραπάνω από δύο γονείς.

Ο **ρόλος των εργατριών** περιορίζεται σε μια απλή **διαδικασία τοπικής αναζήτησης**, που στην ουσία παίζει το ρόλο του ταΐσματος των νεογνών με βασιλικό πολτό με στόχο να βρεθεί μια καλύτερη βασίλισσα. Για αυτό το λόγο οι εργάτριες δεν είναι ξεχωριστά μέλη

του πληθυσμού αλλά χρησιμοποιούνται ως διαδικασίες τοπικής αναζήτησης με στόχο να βελτιωθούν οι λύσεις που βρέθηκαν από τη διαδικασία ζευγαρώματος της βασίλισσας. Αν χρησιμοποιηθεί μία μόνο διαδικασία τοπικής αναζήτησης τότε δεν εκμεταλλευόμαστε το γεγονός ότι κάθε μία από τις εργάτριες έχει διαφορετικές ικανότητες και η επιλογή μιας εργάτριας είναι πολύ σημαντική για τη βελτίωση της λύσης του νεογνού. Οι διαφορετικές ικανότητες των εργατριών αντιστοιχούν σε διαφορετικές διαδικασίες τοπικής αναζήτησης, ανάλογα με το πρόβλημα που επιλύουμε. Έτσι, στο ξεκίνημα του αλγορίθμου θα πρέπει να καθορίσουμε το σύνολο των διαδικασιών τοπικής αναζήτησης και να αντιστοιχίσουμε κάθε μία εργάτρια σε μία διαδικασία τοπικής αναζήτησης ή σε συνδυασμό παραπάνω από μίας διαδικασίας τοπικής αναζήτησης. Η επιλογή από το νεογνό για το ποια εργάτρια θα το θρέψει, δηλαδή ποια μέθοδος τοπικής αναζήτησης θα εφαρμοστεί, μπορεί να γίνει με τυχαίο τρόπο ή με μια ποιο συγκεκριμένη διαδικασία. Για παράδειγμα το καλύτερο νεογνό να τραφεί από την ισχυρότερη εργάτρια (πιο αποτελεσματική, θεωρητικά, διαδικασία τοπικής αναζήτησης). **Όταν βρεθεί ένα νεογνό που έχει καλύτερη λύση από τη βασίλισσα τότε την αντικαθιστά και η τρέχουσα βασίλισσα φεύγει από την κυψέλη**. Όλα τα υπόλοιπα νεογνά θα είναι οι κηφήνες στην επόμενη 'πτήση ζευγαρώματος' της νέας βασίλισσας. Ο πληθυσμός των κηφήνων δεν μεταβάλλεται αφού όπως αναφέραμε και παραπάνω αν ένας κηφήνας επιλεγεί για ζευγάρωμα με τη βασίλισσα σε μία πτήση ζευγαρώματος, αμέσως μετά πεθαίνει, άρα διαγράφεται από το συνολικό πληθυσμό. Αν τώρα ο αριθμός των απογόνων είναι μεγαλύτερος από τον αριθμό των κηφήνων που διαγράφηκαν σε μια πτήση ζευγαρώματος τότε επιλέγεται ίσος αριθμός κηφήνων για την επόμενη πτήση όσο ήταν ο αρχικός πληθυσμός.

Παρακάτω παρουσιάζεται ένας αλγόριθμος υπό μορφή ψευδοκώδικα της παραπάνω μεθόδου:

**Αλγόριθμος** Βελτιστοποίηση Ζευγαρώματος Μελισσών

Αρχικοποίηση

Δημιουργία αρχικού πληθυσμού

Επιλογή βέλτιστου μέλους του πληθυσμού για βασίλισσα

Επιλογή μέγιστου αριθμού από πτήσεις ζευγαρώματος (M)

**do while** i≤ M

Αρχικοποίηση της σπερματοθήκης, της ενέργειας και της ταχύτητας της βασίλισσας

Επιλογή α και step

**do while** energy > 0 και spermatheca δεν είναι γεμάτη

Επιλογή ενός κηφήνα

**if** γίνει το ζευγάρωμα με τον κηφήνα **then**

**Πρόσθεσε** το σπέρμα του κηφήνα στη σπερματοθήκη

**Διαγραφή** του κηφήνα από τον πληθυσμό

**endif**

**Speed(t+1) = α × Speed(t)**

**energy(t+1) = energy(t) – step**

**enddo**

**do** j=1, Size of spermatheca

Επιλογή μιας ποσότητας σπέρματος από τη σπερματοθήκη

Δημιουργία ενός νεογνού με τη χρήση τελεστή διασταύρωσης

Επιλογή μιας εργάτριας

Βελτίωση της λύσης του νεογνού με τη χρήση της εργάτριας

**if** η λύση του νεογνού είναι καλύτερη από τη βασίλισσα **then**

**Αντικατάστησε** τη βασίλισσα με το νεογνό

**Διαγραφή** της παλιάς βασίλισσας από τον πληθυσμό

**else**

**Πρόσθεσε** το νεογνό στον πληθυσμό με τους κηφήνες

**endif**

**enddo**

**enddo**

**Επιστροφή** Βασίλισσας (Εύρεση βέλτιστης λύσης)

Θα μπορούσαμε να αναφέρουμε δύο κατηγορίες αλγορίθμων συγκεντρωτικά, οι οποίες είναι οι εξής:

- ***Αλγόριθμοι που βασίζονται στην προσομοίωση της διαδικασίας ζευγαρώματος (mating)***
- ***Αλγόριθμοι που βασίζονται στην προσομοίωση της διαδικασίας εύρεσης τροφής (foraging behavior)***

## 1.6 Λίγα λόγια για τον αλγόριθμο βελτιστοποίησης ζευγαρώματος μπάμπουρων και για τον αλγόριθμο τεχνητής αποικίας μελισσών

Ο **αλγόριθμος βελτιστοποίησης ζευγαρώματος μπάμπουρων** στηρίζεται στη συμπεριφορά των μπάμπουρων κατά τη διαδικασία του ζευγαρώματος και **προσομοιώνει τη διαδικασία ζευγαρώματος** της βασίλισσας των μπάμπουρων στην κυψέλη. Τα χαρακτηριστικά τους δε διαφέρουν πολύ από αυτά των μελισσών. Πρέπει να αναφερθεί ότι **ο αλγόριθμος αυτός ανήκει στην ίδια κατηγορία αλγορίθμων που ανήκει και ο αλγόριθμος του ζευγαρώματος των μελισσών** ωστόσο παρουσιάζουν κάποιες αξιόλογες διαφορές όπως ότι ο ρόλος των εργατριών είναι διαφορετικός (στην παρούσα περίπτωση οι εργάτριες αποτελούν διαφορετικές λύσεις και δεν είναι μέθοδοι τοπικής αναζήτησης), οι απόγονοι των μπάμπουρων είναι τριών ειδών (βασίλισσα, εργάτριες και κηφήνες) ενώ των μελισσών δύο (βασίλισσα και κηφήνες) κ.ά.

Ο **αλγόριθμος τεχνητής αποικίας μελισσών** είναι ένας αλγόριθμος βελτιστοποίησης που βασίζεται στην συμπεριφορά ενός σμήνους μελισσών. Εφαρμόζεται κυρίως σε συνεχή προβλήματα και στηρίζεται στη λογική ότι περιοχές με μεγάλες ποσότητες τροφής προσελκύουν περισσότερες μέλισσες. Προσομοιώνει τη διαδικασία του *'χορού των μελισσών'* (waggled dance) που ένα σμήνος από μέλισσες πραγματοποιεί κατά τη διαδικασία αναζήτησης τροφής. Αυτό που συμβαίνει στην πραγματικότητα είναι ότι οι μέλισσες φεύγουν από την κυψέλη για την αναζήτηση τροφής και όταν βρουν την τροφή, επειδή δεν μπορούν να την συλλέξουν και να την μεταφέρουν μόνες τους στην κυψέλη, επιστρέφουν στην κυψέλη, ξεφορτώνουν το νέκταρ που πήραν μαζί τους και ενημερώνουν τις υπόλοιπες μέλισσες για την πηγή τροφής που βρήκαν. Η ενημέρωση γίνεται με ειδικές κινήσεις, οι οποίες επιστημονικά ονομάζονται 'χορός εντός της

κυψέλης', με σκοπό την ανταλλαγή πληροφοριών μεταξύ τους. Οι κυριότερες πληροφορίες είναι το πόσο πλούσια σε τροφή (νέκταρ) είναι η πηγή που βρήκαν και πόσο απέχει από την κυψέλη. Σκοπός του 'χορού' είναι να ενημερώσουν και να πείσουν όσο το δυνατόν περισσότερες μέλισσες να τις ακολουθήσουν στις πηγές τροφής. Η κάθε μέλισσα χορεύει σε όσο το δυνατόν περισσότερους χώρους της κυψέλης για να πείσει όσο περισσότερες μέλισσες μπορεί. Επιστημονικές μελέτες έχουν δείξει ότι όταν οι μέλισσες χορεύουν, η κατεύθυνση που έχουν δείχνει την κατεύθυνση της πηγής τροφής σε σχέση με τον ήλιο, η ένταση των κινήσεων δείχνει την απόσταση της πηγής τροφής ενώ η χρονική διάρκεια του χορού δείχνει την ποσότητα της τροφής της πηγής. Ο αλγόριθμος αυτός αναφέρεται στους αλγορίθμους που βασίζονται στην **προσομοίωση εύρεσης τροφής (foraging behavior).**

# Κεφάλαιο 2ο

# Εφαρμογή του αλγορίθμου HBMO στο ανοιχτό πρόβλημα δρομολόγησης οχημάτων

## *Περιγραφή*

Σε αυτό το κεφάλαιο θα παρουσιαστεί το ανοιχτό πρόβλημα δρομολόγησης οχημάτων που επιλύθηκε με τον αλγόριθμο HBMO. Σκοπός είναι να διερευνηθεί ο αλγόριθμος αυτός και να συγκριθούν τα αποτελέσματα που δίνει με τα αντίστοιχα βέλτιστα σε συγκεκριμένα παραδείγματα. Τα παραδείγματα αυτά διαθέτουν δεδομένα (πλήθος των κόμβων, τη χωρητικότητα του οχήματος, το μέγιστο μήκος διαδρομής και το χρόνο εξυπηρέτησης και για αυτά τα δεδομένα διαθέτουν βέλτιστες τιμές κόστους που έχουν βρεθεί μέχρι σήμερα. Η υλοποίηση του έχει πραγματοποιηθεί στο περιβάλλον της Matlab.

Για τη μορφοποίηση ενός προβλήματος βελτιστοποίησης περιλαμβάνονται τα παρακάτω βασικά στοιχεία συνόλων:

- **Μεταβλητές απόφασης (decision variables)** : Είναι οι άγνωστοι που πρέπει να καθοριστούν από την επίλυση του προτύπου. Οι παράμετροι αποτελούν τις μεταβλητές ελέγχου του συστήματος.
- **Περιορισμοί (constraints)** : Για τη μέτρηση των φυσικών περιορισμών του συστήματος, το πρότυπο πρέπει να περιέχει περιορισμούς που να περιορίζουν τις μεταβλητές απόφασης στις επιτρεπτές τιμές του.
- **Αντικειμενική συνάρτηση (objective function)** : Η αντικειμενική συνάρτηση καθορίζει το μέτρο της αποτελεσματικότητας του συστήματος ως μαθηματική συνάρτηση των μεταβλητών απόφασής του. Γενικά η βέλτιστη λύση ενός μοντέλου επιτυγχάνεται όταν οι αντίστοιχες τιμές των μεταβλητών απόφασης οδηγούν στη βέλτιστη τιμή της αντικειμενικής συνάρτησης ικανοποιώντας ταυτοχρόνως όλους τους περιορισμούς.

Στο δικό μας πρόβλημα οι μεταβλητές απόφασης είναι ο αριθμός πελατών, η χωρητικότητα κάθε οχήματος, το μέγιστο μήκος διαδρομής, οι αποστάσεις των πελατών μεταξύ τους, ο χρόνος εξυπηρέτησης και η ζήτηση του κάθε πελάτη. Οι περιορισμοί είναι δύο: η συνολική απόσταση κάθε διαδρομής και ο χρόνος εξυπηρέτησης να μη ξεπερνάει το μέγιστο μήκος διαδρομής και η συνολική ζήτηση κάθε διαδρομής να μη ξεπερνάει τη χωρητικότητα του οχήματος. Τέλος, η αντικειμενική συνάρτηση είναι αυτή του κόστους όπου υπολογίζεται το συνολικό κόστος μιας λύσης (δηλαδή όλων των επιμέρους διαδρομών του OVRP) και αξιολογείται. Φυσικά το κόστος τίθεται προς ελαχιστοποίηση.

Παρακάτω παρουσιάζεται ο ακριβής αλγόριθμος της εργασίας υπό μορφή ψευδοκώδικα:

***Αλγόριθμος***

Αρχικοποίηση (εισαγωγή δεδομένων)

Δημιουργία αρχικών λύσεων

Βελτίωση λύσεων με μεθόδους τοπικής αναζήτησης (1)

Επιλογή βέλτιστης λύσης ως βασίλισσα

***for i=1***, Διασταύρωση όλων των λύσεων με τη βασίλισσα για δημιουργία απογόνων (3)

Μετάλλαξη των απογόνων

Βελτίωση των απογόνων με μεθόδους τοπικής αναζήτησης (2)

Δημιουργία ενός διανύσματος που θα περιέχει τη βασίλισσα, τις λύσεις του σημείου (1) και τις λύσεις του σημείου (2)

Επιλογή τις καλύτερης λύσης ως βασίλισσα και αντικατάσταση της και των καλύτερων N-1 λύσεων για να πάμε πάλι στο σημείο (3)

***end***

## 2.1 Travelling Salesman Problem (TSP) : Το πρόβλημα του πλανόδιου πωλητή

### *Αναπαράσταση λύσεων*

Ένας πωλητής έχει να επισκεφθεί όλες τις πόλεις μιας λίστας μια φορά. Για να γίνει αυτό ξεκινάει από την πόλη του και στο τέλος πρέπει να επιστρέψει σε αυτή. Ο πωλητής μπορεί να επιλέξει τη σειρά με την οποία θα επισκεφθεί τις πόλεις έτσι ώστε η απόσταση που θα διανύσει να είναι η μικρότερη δυνατή. Καθώς ο πωλητής προσπαθεί να βρει τη συντομότερη διαδρομή, αντιμετωπίζει το αποκαλούμενο πρόβλημα του πλανόδιου πωλητή TSP (Travelling Salesman Problem).

Στο ευκλείδειο πρόβλημα TSP η απόσταση μεταξύ των πόλεων ορίζεται όπως η ευκλείδεια απόσταση. Αυτό σημαίνει για δύο πόλεις με συντεταγμένες (a,b) και (c,d) αντίστοιχα πως η απόσταση υπολογίζεται ως εξής : $\boldsymbol{d = \sqrt{(a-c)^2 + (b-d)^2}}$ .

Θεωρούμε ένα πλήθος N πελατών που είναι διασκορπισμένοι στο χώρο (βλέπε σχήμα 1, Κεφάλαιο 1ο) και θεωρούμε και μία τυχαία λύση ως εξής:

*1 2 3 4 … N-1 N*

Πάντα η αποθήκη (depot) αναπαρίσταται με τη μονάδα. Άρα η παραπάνω λύση σημαίνει ότι το όχημα θα ξεκινήσει από την αποθήκη κι έπειτα θα πάει στον πελάτη 2, μετά θα συνεχίσει πηγαίνοντας στον πελάτη 3 κ.ο.κ. μέχρι να εξυπηρετήσει και τον πελάτη N.

## 2.2 Αλγόριθμοι που χρησιμοποιήθηκαν για την κατασκευή αρχικών λύσεων

Η πρώτη κατηγορία αλγορίθμων που έχουν εφαρμοστεί για την επίλυση του παραπάνω προβλήματος είναι οι αλγόριθμοι κατασκευής μίας αρχικής λύσης. Αυτοί οι αλγόριθμοι ανήκουν στην κατηγορία των άπληστων ευρετικών αλγορίθμων και είναι πολύ χρήσιμοι όταν χρειαζόμαστε μία καλή αρχική λύση. Παρακάτω παρουσιάζονται κάποιοι από αυτούς, οι οποίοι χρησιμοποιήθηκαν για τη δημιουργία αρχικών λύσεων στο δικό μας πρόβλημα. Ο πιο γνωστός από αυτούς είναι ο αλγόριθμος του

Πλησιέστερου Γείτονα (Nearest Neighbor) ο οποίος ήταν από τους πρώτους που προσπάθησαν να δώσουν μια λύση στο πρόβλημα του πλανόδιου πωλητή.

### 2.2.1 Nearest Neighbor (NN) : Ο αλγόριθμος του πλησιέστερου γείτονα

Σε αυτήν την περίπτωση ο πωλητής ξεκινά από μια πόλη και έπειτα επισκέπτεται την αμέσως κοντινότερη πόλη της προηγούμενης που βρισκόταν. Στη συνέχεια πηγαίνει στην πλησιέστερη αυτής που ήταν, που δεν έχει επισκεφθεί ως τώρα κ.ο.κ. Η διαδικασία αυτή συνεχίζεται μέχρι να επισκεφθεί όλες τις πόλεις και στο τέλος επιστρέφει σε αυτήν από όπου ξεκίνησε. Η διαδικασία με τα βήματα για την υλοποίηση του φαίνεται ακολούθως:

***Βήμα 1ο*** Ξεκινάμε από οποιονδήποτε κόμβο ως ξεκίνημα μονοπατιού.

***Βήμα 2ο*** Βρίσκουμε την πλησιέστερη απόσταση από τον τελευταίο κόμβο του μονοπατιού. Προσθέτουμε τον κόμβο αυτό στο μονοπάτι.

***Βήμα 3ο*** Επαναλαμβάνουμε το 2ο βήμα μέχρις ότου προστεθούν όλοι οι κόμβοι στο μονοπάτι.

Έστω ότι έχουμε, ενός συνόλου 10 κόμβων, τις αποστάσεις μεταξύ τους, οι οποίες έχουν προκύψει από τη σχέση **d =** $\sqrt{(a-c)^2+(b-d)^2}$ που αναφέρθηκε προηγουμένως. Προφανώς ο παρακάτω πίνακας είναι συμμετρικός.

| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|
| 1 | **0** | **31** | **16** | **6** | **1** | **49** | **20** | **89** | **10** | **15** |
| 2 | **31** | **0** | **4** | **22** | **56** | **40** | **25** | **35** | **5** | **41** |
| 3 | **16** | **4** | **0** | **25** | **51** | **13** | **90** | **36** | **42** | **17** |
| 4 | **6** | **22** | **25** | **0** | **38** | **85** | **8** | **11** | **27** | **30** |
| 5 | **1** | **56** | **51** | **38** | **0** | **39** | **95** | **2** | **101** | **16** |
| 6 | **49** | **40** | **13** | **85** | **39** | **0** | **3** | **62** | **15** | **71** |
| 7 | **20** | **25** | **90** | **8** | **95** | **3** | **0** | **29** | **10** | **86** |
| 8 | **89** | **35** | **36** | **11** | **2** | **62** | **29** | **0** | **71** | **9** |
| 9 | **10** | **5** | **42** | **27** | **101** | **15** | **10** | **71** | **0** | **43** |
| 10 | **15** | **41** | **17** | **30** | **16** | **71** | **86** | **9** | **43** | **0** |

*Πίνακας 1: Αποστάσεις από κόμβο σε κόμβο*

Σύμφωνα με την παραπάνω διαδικασία ξεκινάμε από τον πρώτο κόμβο. Πλησιέστερος σε αυτόν είναι ο κόμβος 5. Έπειτα ο κοντινότερος του που δεν έχει προστεθεί στη λύση είναι ο 8 κ.ο.κ. Τελικά **η σειρά των κόμβων** προκύπτει να είναι :

**1 – 5 – 8 – 10 – 3 – 2 – 9 – 7 – 6 – 4**

**με συνολικό κόστος** : 1 + 2 + 9 + 17 + 4 + 5 + 10 + 3 + 85 = **136**

### 2.2.2 Nearest Insertion: Αλγόριθμος πλησιέστερης εισαγωγής

Σε αυτή τη διαδικασία παίρνουμε ένα στοιχειώδη κύκλο και προσπαθούμε να βρούμε ποιος κόμβος (που δεν είναι αυτή τη στιγμή στον κύκλο) μπορεί να μπει αμέσως μετά τη διαδρομή και στη συνέχεια καθορίζουμε σε ποιο σημείο του κύκλου εισέρχεται. Στον αλγόριθμο αυτό εισάγουμε τον κόμβο που βρίσκεται πιο κοντά σε κάποιον άλλο της υπό-περιοδείας. Τα βήματα υλοποίησης του είναι:

***Βήμα 1ο*** Ξεκινάμε με ένα υπογράφημα που περιλαμβάνει μόνο τον κόμβο i.

***Βήμα 2ο*** Βρίσκουμε τον κόμβο k του οποίου το κόστος $c_{ik}$ είναι ελάχιστο και δημιουργούμε τη διαδρομή i – k – i.

***Βήμα 3ο*** Βήμα επιλογής. Δοθέντος μιας διαδρομής, βρίσκουμε ένα κόμβο k που δεν είναι στη διαδρομή και είναι πλησιέστερα σε οποιοδήποτε κόμβο της διαδρομής.

***Βήμα 4ο*** Βρίσκουμε το τόξο (i,j) στη διαδρομή που **ελαχιστοποιεί το** $c_{ik} + c_{kj} - c_{ij}$. Εισάγουμε το k ανάμεσα στα i και j.

***Βήμα 5ο*** Επιστρέφουμε στο βήμα 3 εκτός αν έχουμε κύκλο Χάμιλτον, δηλαδή έχουμε περάσει από όλους τους κόμβους ακριβώς μία φορά και έχουμε επιστρέψει στον αρχικό κόμβο.

Εφαρμόζοντας τα βήματα στον πίνακα των αποστάσεων δημιουργείται η αλληλουχία των κόμβων. Ακολούθως γίνεται αναλυτική παρουσίαση όλων των βημάτων:

| 1η επανάληψη | | 2η επανάληψη | | 3η επανάληψη | | 4η επανάληψη | | 5η επανάληψη | |
|---|---|---|---|---|---|---|---|---|---|
| **n** | **$c_{ij}$** | **n** | **$c_{ij}$** | **n** | **$c_{ij}$** | **n** | **$c_{ij}$** | **n** | **$c_{ij}$** |
| **1** | **-** | **1,5** | 1 | **1,8,5** | 90 | **1,4,5** | 43 | **1,7,5** | 114 |
| | | | | **1,5,8** | 2 | **5,4,8** | 47 | **5,7,8** | 122 |
| | | | | | | **8,4** | 11 | **8,7,4** | 26 |
| | | | | | | | | **4,7** | 8 |

| 6η επανάληψη | | 7η επανάληψη | | 8η επανάληψη | | 9η επανάληψη | | 10η επανάληψη | |
|---|---|---|---|---|---|---|---|---|---|
| **n** | **$c_{ij}$** | **n** | **$c_{ij}$** | **n** | **$c_{ij}$** | **n** | **$c_{ij}$** | **n** | **$c_{ij}$** |
| **1,6,5** | 87 | **1,10,5** | 30 | **1,9,5** | 110 | **1,2,5** | 86 | **1,3,5** | 66 |
| **5,6,8** | 99 | **5,10,8** | 23 | **5,9,10** | 128 | **5,2,10** | 81 | **5,3,10** | 52 |
| **8,6,4** | 136 | **8,10,4** | 28 | **10,9,8** | 105 | **10,2,8** | 67 | **10,3,8** | 44 |
| **4,6,7** | 80 | **4,10,7** | 108 | **8,9,4** | 87 | **8,2,4** | 46 | **8,3,4** | 50 |
| **7,6** | 3 | **7,10,6** | 154 | **4,9,7** | 29 | **4,2,7** | 39 | **4,3,7** | 107 |
| | | **6,10** | 71 | **7,9,6** | 22 | **7,2,6** | 62 | **7,3,6** | 100 |
| | | | | **6,9** | 15 | **6,2,9** | 30 | **6,3,9** | 40 |
| | | | | | | **9, 2** | 5 | **9,3,2** | 41 |
| | | | | | | | | **2,3** | 4 |

*Πίνακας 2: Επαναλήψεις της μεθόδου πλησιέστερης εκχώρησης*

### 2.2.3 Greedy Randomized Adaptive Search Procedure (GRASP): Διαδικασία Άπληστης Τυχαιοποιημένης Προσαρμοστικής Αναζήτησης

Η διαδικασία άπληστης τυχαιοποιημένης προσαρμοστικής αναζήτησης είναι ένας αλγόριθμος δύο φάσεων όπου στην πρώτη φάση μία λύση κατασκευάζεται βήμα προς βήμα και στη δεύτερη φάση χρησιμοποιείται ένας αλγόριθμος τοπικής αναζήτησης για να βελτιωθεί αυτή η λύση. Εδώ γίνεται λόγος για την πρώτη φάση, δηλαδή για την κατασκευή μιας λύσης. Θα μπορούσε να ισχυριστεί κανείς ότι αυτός ο αλγόριθμος είναι μία υποκατηγορία του αλγορίθμου του πλησιέστερου γείτονα. Συνδυάζοντας τον αλγόριθμο του πλησιέστερου γείτονα με μία λίστα περιορισμού των υποψηφίων (Restricted Candidate List), στην οποία συμμετέχουν οι καλύτερες επιλογές για κάθε μία

κίνηση προκύπτει ο αλγόριθμος GRASP. Το μέγεθος και ο τρόπος που κατασκευάζεται η λίστα περιορισμού των υποψηφίων είναι το στοιχείο που χαρακτηρίζει αυτόν τον αλγόριθμο και μπορεί να έχει ό,τι μέγεθος επιθυμεί ο καθένας. Αν, για παράδειγμα χρησιμοποιηθεί μία λίστα μεγέθους 4 τότε ο πωλητής έχει 25% πιθανότητα να επιλέξει τον επόμενο κόμβο. Αντίστοιχα, η λίστα αυτή έχει μέγεθος 3 τότε η πιθανότητα αυτή ανέρχεται στο 33% κ.ο.κ. Στην τρέχουσα εργασία έχει χρησιμοποιηθεί ο αλγόριθμος αυτός με λίστα μεγέθους δύο και τέσσερα.

## *Διαδικασία*

- ***Λίστα μεγέθους 4***

Έστω ότι ξεκινάμε από τον πρώτο κόμβο. Οι τέσσερις πλησιέστεροι κόμβοι που θα δημιουργήσουν τη λίστα είναι ο κόμβος 4 με κόστος $c_{1,4} = 6$, ο κόμβος 5 με κόστος $c_{1,5} = 1$, ο κόμβος 9 με κόστος $c_{1,9} = 10$ και ο κόμβος 10 με κόστος $c_{1,10} = 15$. Έστω ότι οι τέσσερις κόμβοι έχουν ίσες πιθανότητες να επιλεγούν. Τότε θα χρησιμοποιηθεί μία γεννήτρια τυχαίων αριθμών όπου αν η τιμή της είναι από 0 έως 0.25 τότε επιλέγεται ο πρώτος κόμβος δηλαδή ο 4, από 0.251 έως 0.5 επιλέγεται ο δεύτερος κόμβος, από 0.51 έως 0.75 επιλέγεται ο τρίτος και από 0.751 έως 1 επιλέγεται ο τέταρτος. Κάθε φορά ψάχνουμε τους τέσσερις κοντινότερους του τελευταίου κόμβου εξαιρουμένων όλων των προηγούμενων που έχουν ήδη χρησιμοποιηθεί για την κατασκευή της λύσης. Αν για παράδειγμα η τιμή της γεννήτριας είναι 0.46 τότε επιλέγεται ο δεύτερος κόμβος δηλαδή ο κόμβος 5. Έπειτα ψάχνουμε τους τέσσερις κοντινότερους κόμβους από τον 5 εκτός του 1, βρίσκουμε ένα τυχαίο αριθμό μέσω της γεννήτριας και επιλέγουμε τον κόμβο που θα εισαχθεί αναλόγως σε ποιο διάστημα τιμών κυμάνθηκε ο τυχαίος αριθμός. Η διαδικασία ολοκληρώνεται όταν και ο τελευταίος κόμβος εισαχθεί στη λύση.

- ***Λίστα μεγέθους 2***

Ακολουθείται η ίδια διαδικασία με την παραπάνω, με τη διαφορά ότι η πιθανότητα να επιλεχθεί ένας κόμβος είναι 50%. Τότε θα χρησιμοποιηθεί η γεννήτρια τυχαίων αριθμών κι αν η τιμή της κυμαίνεται από 0 έως 0.5 επιλέγεται ο πρώτος κόμβος, διαφορετικά επιλέγεται ο δεύτερος κόμβος.

### 2.2.4 Διαδικασία πλησιέστερης εκχώρησης συνδυασμένη με την Διαδικασία Άπληστης Τυχαιοποιημένης Προσαρμοστικής Αναζήτησης

Αυτός ο συνδυασμός διαδικασιών είναι μια παραλλαγή των δύο, εφαρμοσμένοι ταυτοχρόνως. Ακολουθείται η ίδια μέθοδος, της πλησιέστερης εκχώρησης μόνο που σε κάθε βήμα η επιλογή της επικρατέστερης διαδρομής ακολουθεί τη διαδικασία άπληστης τυχαιοποιημένης προσαρμοστικής αναζήτησης. Πιο αναλυτικά, σε αυτό το βήμα της επιλογής, η επιλογή γίνεται με τη βοήθεια μια γεννήτριας τυχαίων αριθμών από το 0 έως το 1. Αν η τιμή της γεννήτριας είναι κάτω από 0.5 τότε επιλέγεται η πρώτη υποψήφια λύση, αλλιώς η δεύτερη.

## 2.3 Κριτήριο αξιολόγησης λύσεων

Προκειμένου να χαρακτηριστεί μία λύση ως καλή ή κακή θα πρέπει να αξιολογηθεί με ένα κριτήριο. Στο πρόβλημα που πρέπει να λύσουμε το κριτήριο αυτό θα είναι το κόστος της λύσης. Έτσι, δημιουργήθηκε μια υπορουτίνα (subroutine) στο περιβάλλον της Matlab που υπολογίζει το αντίστοιχο κόστος κάθε λύσης. Ως ορίσματα δέχεται το διάνυσμα της λύσης, τον πίνακα των αποστάσεων των κόμβων, το μέγιστο μήκος διαδρομής, τη χωρητικότητα των οχημάτων, τη ζήτηση κάθε κόμβου και τον αριθμό κόμβων και στο τέλος επιστρέφει την τιμή του κόστους. Αρχικά λύνει το ανοιχτό πρόβλημα δρομολόγησης οχημάτων (OVRP) κι έπειτα υπολογίζει το κόστος όλων των διαδρομών που δημιουργήθηκαν. Παρακάτω ακολουθεί ο αλγόριθμος υπολογισμού του κόστους υπό μορφή ψευδοκώδικα:

***Αλγόριθμος***

Αρχικοποίηση

***for j=1***

***if*** άθροισμα της ζήτησης + ζήτηση του τρέχοντος κόμβου ≤ χωρητικότητα και άθροισμα των αποστάσεων + τρέχουσα απόσταση ≤ μέγιστου μήκους διαδρομής ***then***

Πρόσθεσε τον τρέχων κόμβο σε αυτή τη διαδρομή

Προσάρμοσε το άθροισμα της ζήτησης και το άθροισμα των αποστάσεων

***else***

Δημιούργησε νέα διαδρομή

Πρόσθεσε τον τρέχων κόμβο στη νέα διαδρομή

Προσάρμοσε το άθροισμα της ζήτησης και το άθροισμα των αποστάσεων

***end***

***end***

***for i=1***

Υπολόγισε το κόστος της διαδρομής

***end***

Πρόσθεσε όλα τα κόστη των διαδρομών

Επίστρεψε το κόστος της διαδρομής

## 2.4 Μέθοδοι Τοπικής Αναζήτησης

Η τοπική αναζήτηση αποτελεί μία από τις αρχαιότερες μεθόδους βελτιστοποίησης και στηρίζεται στη μέθοδο δοκιμής και σφάλματος**.** Έχει αποδειχθεί πολύ επιτυχημένη εφαρμοσμένη στην πράξη και έχει επιλύσει πολλά προβλήματα συνδυαστικής βελτιστοποίησης. Έπειτα παρουσιάζεται ένας γενικός αλγόριθμος.

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

Σε αυτό το σημείο θα παρουσιαστούν αναλυτικά οι μέθοδοι που χρησιμοποιήθηκαν στην παρούσα εργασία για τη βελτιστοποίηση του κόστους. Να αναφερθεί ότι οι αλγόριθμοι αυτοί εφαρμόστηκαν πάνω στο πρόβλημα του πλανόδιου πωλητή.

### 2.4.1 2-opt

Είναι ένας από τους πιο διαδεδομένους αλγόριθμους τοπικής αναζήτησης που επιλύει τέτοιου είδους προβλήματα. Αντικείμενο αυτή της μεθόδου είναι η διαγραφή 2 ακμών και η επανασύνδεση δύο μονοπατιών με διαφορετικό τρόπο. Με αυτόν τον τρόπο καθορίζεται μία καινούρια διαδρομή. Σκοπός του αλγορίθμου είναι το κόστος της νέας διαδρομής να είναι μικρότερο από το προηγούμενο. Τα βήματα φαίνονται ακολούθως:

***Βήμα 1ο*** Έστω T η τρέχουσα διαδρομή.

***Βήμα 2ο*** Εξετάζουμε για κάθε κόμβο i = 1, …,n όλες τις πιθανές 2-opt κινήσεις που μπορεί να γίνουν από την i και την επόμενη της μέσα στη διαδρομή. Σε περίπτωση που μειωθεί το κόστος κρατάμε τη διαδρομή αυτή και εφαρμόζουμε τις αλλαγές στη διαδρομή T.

***Βήμα 3ο*** Αν δε βελτιώνεται η περαιτέρω η λύση, σταματάμε.

Ένα σημείο όπου πρέπει να σταθούμε είναι στον τρόπο που γίνεται η επιλογή των τόξων διαγραφής και των τόξων που θα εισαχθούν στη λύση. Ένας τρόπος είναι η διαγραφή των χειρότερων τόξων και η αντικατάσταση τους. Χειρότερα τόξα ορίζονται εδώ αυτά με το μεγαλύτερο κόστος. Σε αυτή τη μέθοδο όταν διαγραφούν δύο τόξα υπάρχει μοναδικός τρόπος επανασύνδεσής τους διαφορετικά δε διατηρείται ο κύκλος. Ένας άλλος τρόπος είναι η διαγραφή του χειρότερου τόξου και η δοκιμή διάφορων παραλλαγών με τα υπόλοιπα τόξα. Ο τελευταίος τρόπος είναι η τυχαία διαγραφή δύο τόξων. Πλεονέκτημα αυτής της παραλλαγής είναι η ταχύτητα της αφού δεν κρίνεται απαραίτητος ο υπολογισμός των χειρότερων τόξων, βέβαια θα μπορούσε να αποτελέσει και μειονέκτημα ο ίδιος λόγος στην περίπτωση όπου δε μπορεί να βρει κάποια καλύτερη λύση. Η δεύτερη περίπτωση θα επιβράδυνε τον αλγόριθμο.

Ας δούμε πώς αλλάζει μία λύση που επιδέχεται 2-opt. Έστω μία αρχική λύση που έχει δημιουργηθεί με τυχαία αλληλουχία:

1 5 7 4 10 2 6 3 8 9

Το κόστος αυτής της λύσης σύμφωνα με τον πίνακα αποστάσεων του παραδείγματος είναι c=335.

Έστω ότι με έναν από τους παραπάνω τρόπους διαγράφονται τα τόξα 7-4 και 6-3 τότε η λύση θα παραλλαχθεί ως εξής:

1 5 7 | 4 10 2 6 | 3 8 9 ➡ 1 5 7 **6 2 10 4** 3 8 9

Έπειτα υπολογίζεται το κόστος και αναλόγως δεχόμαστε ή απορρίπτουμε τη λύση. Το νέο κόστος είναι *c=342*, συνεπώς απορρίπτεται η νέα λύση. Τέλος δοκιμάζουμε τη διαγραφή άλλων τόξων ώσπου το κόστος να κατέβει κάτω από την τιμή 335.

### 2.4.2 3-opt

Αυτή η διαδικασία αποτελεί μία παραλλαγή του αλγορίθμου 2-opt, αφού περιλαμβάνει τη διαγραφή 3 ακμών σε ένα δίκτυο, την επανασύνδεση τους με όλους τους δυνατούς τρόπους και στη συνέχεια την αξιολόγηση τους. Τα βήματα της διαδικασίας είναι τα εξής:

***Βήμα 1ο*** Έστω T η τρέχουσα διαδρομή.

***Βήμα 2ο*** Για κάθε κόμβο i ∈ V υπολογίζουμε ένα σύνολο από κόμβους N(i).

***Βήμα 3ο*** Εξετάζονται για κάθε κόμβο i = 1,…n όλες οι πιθανές 3-opt κινήσεις που μπορεί να γίνουν και οι οποίες διαγράφουν από τρεις πλευρές έχοντας η κάθε μία από αυτές μια πλευρά στην N(i). Αν με αυτό τον τρόπο μπορεί να μειωθεί το κόστος της διαδρομής τότε επιλέγεται η βέλτιστη 3-opt κίνηση και εφαρμόζουμε τις αλλαγές στη διαδρομή T.

***Βήμα 4ο*** Αν δεν επιδέχεται περαιτέρω βελτίωση η λύση, σταματάμε.

Και σε αυτή τη μέθοδο η επιλογή των τόξων προς διαγραφή είναι ένα κρίσιμο σημείο. Γι' αυτό πρέπει να δοθεί προσοχή να μη χωριστεί ο κύκλος σε 2 ή 3 κύκλους οπότε θα είχαμε μη-εφικτούς κύκλους δηλαδή και μη εφικτή λύση. Θα χρησιμοποιήσουμε το ίδιο διάνυσμα λύσης όπως προηγουμένως για παράδειγμα.

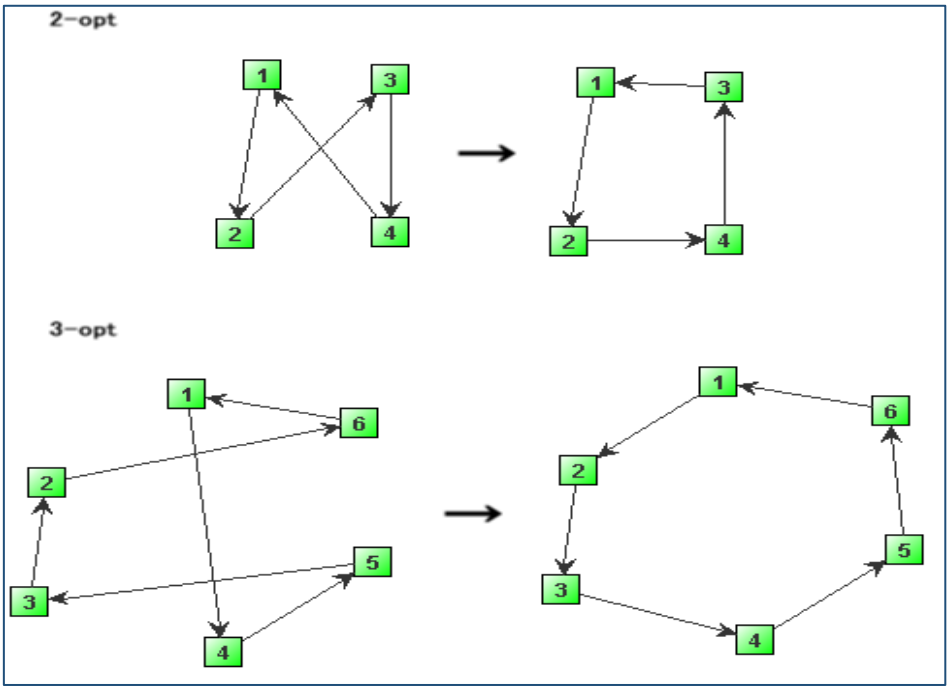
1 5 7 4 10 2 6 3 8 9

με κόστος *c=335*.

Ας υποθέσουμε ότι διαγράφονται τα τόξα 7-4, 2-6 και 8-9. Ένας τρόπος αναπαράστασης της νέας λύσης θα ήταν:

1 5 7 | 4 10 2 | 6 3 8 | 9 ⇨ 1 5 7 **2 10 4 8 3 6** 9

Το νέο κόστος είναι c=267 άρα η λύση γίνεται δεκτή.




*Εικόνα 4: Σχηματική αναπαράσταση των μεθόδων 2-opt και 3-opt*

## 2.4.3 1-0 relocate (1-0 επανατοποθέτηση)

Μια ακόμη πολύ σημαντική μέθοδος τοπικής αναζήτησης που έχει χρησιμοποιηθεί στην τρέχουσα εργασία είναι η 1-0 επανατοποθέτηση. Η διαδικασία είναι η επανατοποθέτηση ενός κόμβου από το σημείο που βρίσκεται σε ένα άλλο σημείο. Δηλαδή σύμφωνα με το παραπάνω διάνυσμα λύσης αυτό θα υλοποιούταν αν ο κόμβος 10 για παράδειγμα μεταφερόταν μεταξύ του 1 και του 5, δηλαδή:

1 5 7 4 10 2 6 3 8 9 ⇒ 1 **10** 5 7 4 2 6 3 8 9

Ακολουθείται η ίδια διαδικασία υπολογισμού κόστους.

### 2.4.4 2-0 relocate (2-0 επανατοποθέτηση)

Ο αλγόριθμος 2-0 επανατοποθέτηση λειτουργεί ακριβώς με τον ίδιο τρόπο με τον 1-0 επανατοποθέτηση. Η διαφορά είναι ότι αντί να μετακινηθεί ένας κόμβος μετακινούνται δύο γειτονικοί κόμβοι σε άλλη θέση. Πιο παραστατικά έχουμε:

1 5 7 4 10 2 6 3 8 9 ➡ 1 5 7 4 **3 8** 10 2 6 9

Οι γειτονικοί κόμβοι 3 και 8 μετακινήθηκαν ανάμεσα στους κόμβους 4 και 10. Ακολουθείται επίσης η ίδια διαδικασία υπολογισμού κόστους.

### 2.4.5 1-1 Exchange (1-1 ανταλλαγή)

Ένας ακόμα άλλος πολύ γνωστός αλγόριθμος βελτιστοποίησης που επιλύει το πρόβλημα του πλανόδιου πωλητή. Η διαδικασία αυτή αφορά δύο κόμβους, οι οποίοι ανταλλάσσουν θέση μεταξύ τους. Δηλαδή αν οι κόμβοι 2 και 9 ανταλλαχθούν τότε η λύση θα μεταβληθεί ως εξής:

1 5 7 4 10 2 6 3 8 9 ➡ 1 5 7 4 10 **9** 6 3 8 **2**

### 2.4.6 2-2 Exchange (2-2 ανταλλαγή)

Αποτελεί μια παραλλαγή του παραπάνω αλγορίθμου με τη διαφορά ότι ανταλλάσσονται 4 κόμβοι, έχουμε δηλαδή 2 ζεύγη ανταλλαγής.

1 5 7 4 10 2 6 3 8 9 ➡ 1 **8** 7 **9** 10 2 6 3 **5 4**

Ανταλλάχθηκαν οι κόμβοι 8 με 5 και 9 με 4.

*Όλες οι μέθοδοι που έχουν αναφερθεί ως τώρα έχουν ενσωματωθεί σε μία υπορουτίνα η οποία καλείται για κάθε λύση χωριστά. Ο αριθμός τους είναι έξι. Έτσι λοιπόν η βελτιστοποίηση πραγματοποιείται ως εξής: Αφού καλείται η συνάρτηση μείωσης κόστους, μια γεννήτρια αριθμών δίνει ακέραιους αριθμούς από το ένα έως το έξι. Ο αριθμός αυτός ορίζει πόσες από τις παραπάνω μεθόδους τοπικής αναζήτησης θα χρησιμοποιηθούν για τη μείωση του κόστους της προκείμενης λύσης. Η επιλογή του είδους της μεθόδου γίνεται τυχαία. Έτσι, εκτελούνται σειριακά οι αλγόριθμοι σε κάθε*

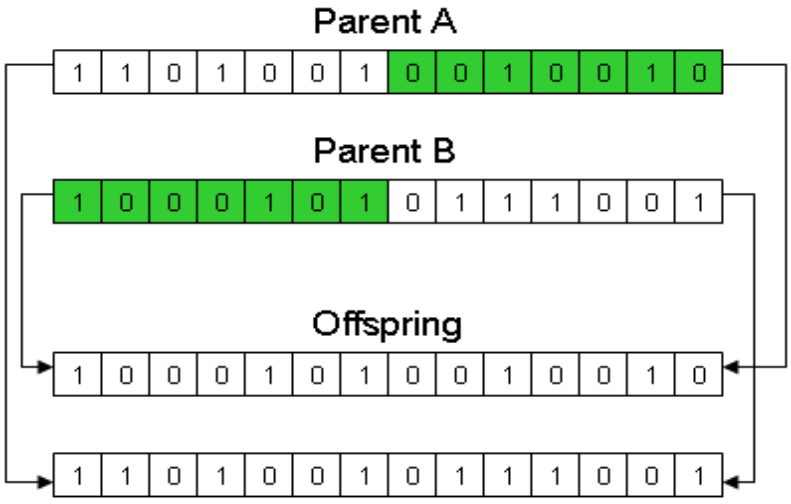
*διάνυσμα λύσης. Περιορισμό αποτελεί το ότι κάθε λύση βελτιώνεται κάθε φορά με διαφορετικό συνδυασμό αλγορίθμων.*

## 2.5 Διαδικασίες Εξελικτικών Αλγορίθμων

Όπως αναλύθηκε σε προηγούμενο κεφάλαιο σχετικά με τους εξελικτικούς αλγορίθμους ο πληθυσμός υφίσταται μια προσομοιωμένη γενετική εξέλιξη χρησιμοποιώντας διάφορους γενετικούς τελεστές όπως η ***διασταύρωση*** και η ***μετάλλαξη***. Έτσι λοιπόν, μετά τη δημιουργία αρχικών λύσεων και τη βελτιστοποίηση τους με τους παραπάνω τρόπους ακολουθούν οι διαδικασίες της διασταύρωσης (crossover) και της μετάλλαξης (mutation).

- ***Διασταύρωση***

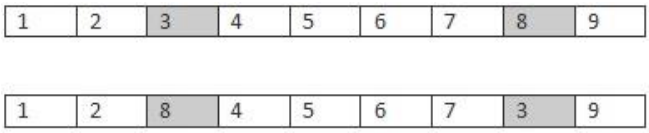
Είναι η διαδικασία κατά την οποία, επιλέγονται δύο άτομα που ονομάζονται *γονείς (parents)* και παράγει με την ανταλλαγή τμημάτων των γονέων δύο νέα άτομα που ονομάζονται απόγονοι *(offspring)*. Παρακάτω ακολουθεί σχηματική αναπαράσταση αυτού.




*Εικόνα 5: Αναπαράσταση διασταύρωσης ενός ζεύγους λύσης*

- ***Μετάλλαξη***

Είναι η διαδικασία όπου ένα υποσύνολο από μεταβλητές επιλέγονται τυχαία και οι τιμές τους αλλάζουν. Ο σκοπός της μετάλλαξης είναι η εισαγωγή νέου γενετικού υλικού στον πληθυσμό. Πιο παραστατικά:




*Εικόνα 6: Αναπαράσταση μετάλλαξης σε ζεύγος απογόνων*

## 2.6 Threshold Accepted Method : Η Μέθοδος Αποδοχής Κατωφλίου

Ανόπτηση ονομάζεται στη μεταλλουργία η θερμική κατεργασία στην οποία υποβάλλεται ένα μέταλλο ή κράμα, που έχει υποστεί κάποια κατεργασία π.χ. σφυρηλάτηση ή ενδοτράχυνση, προκειμένου στη συνέχεια υποβαλλόμενο σε ψύξη να βελτιωθεί η ευκαμψία του και να γίνει λιγότερο εύθρυπτο. Είναι δηλαδή η διαδικασία κατά την οποία ένα στερεό υλικό θερμαίνεται μέχρι το σημείο τήξης του και στη συνέχεια ακολουθεί αργά η ψύξη. Το στάδιο αποκτά κατώτερο επίπεδο ενέργειας με την παύση της ψύξης. Η στρατηγική μείωσης της ψύξης μπορεί να επιφέρει ατέλειες στο σύστημα. Ο αλγόριθμος της προσομοιωμένης ανόπτησης αποτυπώνει την ενέργεια ενός συστήματος το οποίο ψύχεται, μέχρι την ολοκλήρωση, τη σύγκλιση δηλαδή σε κάποιο σημείο ισορροπίας. Αυτό συμβαίνει όταν ο αλγόριθμος σταματήσει την κίνηση στον εφικτό χώρο αναζήτησης, τότε δηλαδή έχει εντοπιστεί το τελικό σημείο ισορροπίας.

Η ***μέθοδος αποδοχής κατωφλίου*** αποτελεί μια μέθοδο η οποία αποτελεί μια διαφορετική έκδοση της μεθόδου ανόπτησης. Ξεφεύγει από το τοπικό ελάχιστο μέσω ενός κατωφλίου αποδοχής T ως πάνω όριο στην αύξηση της τιμής της οποίας επιτρέπεται της αντικειμενικής συνάρτησης, από την μία κίνηση στην επόμενη. Έτσι καθορίζονται νέες αποστάσεις. Η μέθοδος αποδοχής κατωφλίου είναι αποτελεσματικότερη της μεθόδου ανόπτησης καθώς σε εκείνη την περίπτωση παράγεται τυχαίος αριθμός και η κάθε επανάληψη πρέπει να συμβαδίζει με την εκθετική συνάρτηση. Στο δικό μας αλγόριθμο η μέθοδος αυτή έχει ενσωματωθεί στην υπορουτίνα μείωσης κόστους με τη χρήση μεθόδων τοπικής αναζήτησης. Παρακάτω

παρατίθεται σε μορφή ψευδοκώδικα ένας γενικός αλγόριθμος της μεθόδου της αποδοχής κατωφλίου :

***Αλγόριθμος*** Αποδοχής Κατωφλίου

Αρχικοποίηση

Επίλεξε μια αρχική λύση So

Επίλεξε ένα αρχικό κατώφλι T =T max

Επίλεξε μια συνάρτηση μείωσης του κατωφλίου α (T)

***Repeat***

***Repeat***

Τυχαία επιλογή μιας γειτονιάς s N (So)

Δ= f (s) – f(So)

***If*** δ < T then

So = s

***endif***

***Until*** ο μέγιστος αριθμός επαναλήψεων ολοκληρωθεί

T= α (T)

***Until*** κάποιο κριτήριο τερματισμού ικανοποιηθεί.

Επίστρεψε τη βέλτιστη λύση

## 2.7 Αλγόριθμος Μεταβλητής Γειτονιάς Αναζήτησης (Variable Neighborhood Search (VNS))

Ο αλγόριθμος μεταβλητής γειτονιάς αναζήτησης ξεκινάει με μία μέθοδο τοπικής αναζήτησης και μόλις αυτή η μέθοδος βρει κάποιο τοπικό ελάχιστο δοκιμάζει κάποιες άλλες μεθόδους για να δει αν μπορεί να βελτιώσει τη λύση. Πρακτικά, στο δικό μας κώδικα ο αλγόριθμος αυτός έχει εφαρμοστεί στο καλύτερο διάνυσμα λύσης στο τέλος κάθε επανάληψης όπου έχουν εφαρμοστεί διαδοχικά όλες οι μέθοδοι τοπικής αναζήτησης για ένα συγκεκριμένο αριθμό επαναλήψεων.

## 2.8 Εγκλωβισμός σε τοπικό ελάχιστο

Στην περίπτωση που ο αλγόριθμος εγκλωβιστεί σε τοπικό ελάχιστο η διαδικασία που εφαρμόζεται είναι η εξής: Στο τέλος κάθε επανάληψης ένα ποσοστό 'κακών' λύσεων '*χαλάει*' με την εντολή 'randperm' της Matlab. Δηλαδή οι λύσεις αυτές αποσυντίθενται εντελώς και αναδημιουργούνται με ένα τυχαίο τρόπο. Σκοπός αυτού είναι να συμπεριληφθούν και κακές λύσεις, με μεγάλο κόστος δηλαδή, έτσι ώστε να μεγαλώσει το εύρος των λύσεων στο χώρο. Με αυτόν τον τρόπο, μπορεί μεν να αποκτάμε και κακές λύσεις αλλά ο αλγόριθμος γίνεται πιο ευέλικτος, με μεγαλύτερο εύρος λύσεων, κι έτσι η πιθανότητα εγκλωβισμού σε τοπικά ακρότατα μειώνεται.

# Κεφάλαιο 3ο

## Αποτελέσματα

Για τον έλεγχο της αποτελεσματικότητας και της σωστής λειτουργίας του αλγορίθμου εκτελέστηκε μια σειρά από προβλήματα. Σε αυτό το κεφάλαιο θα γίνει παρουσίαση των αποτελεσμάτων του αλγορίθμου HBMO καθώς και σύγκριση αυτών με τις τιμές των βέλτιστων. Η σύγκριση θα γίνει σε σχέση με το τελικό κόστος, για το οποίο επιδιώκεται η ελαχιστοποίηση του. Η εφαρμογή του αλγορίθμου θα γίνει σε 14 διαφορετικά προβλήματα OVRP με τη χρήση του αλγορίθμου HBMO τα οποία θα έχουν διαφορετικό αριθμό κόμβων (πελατών) διαφορετικών συντεταγμένων, διαφορετική χωρητικότητα οχημάτων και χρόνο εξυπηρέτησης. Αρχικά, θα παρουσιαστούν τα δεδομένα του πρώτου προβλήματος αναλυτικά (θα συμπεριληφθούν και συντεταγμένες) ενώ των υπολοίπων πιο περιεκτικά. Τα προβλήματα αναφέρονται ως C.

Παρουσίαση προβλήματος C1

| | |
|---|---|
| Αριθμός πελατών N | 51 |
| Χωρητικότητα Οχημάτων | 160 |
| Μέγιστο Μήκος Διαδρομής | ∞ |
| Χρόνος Εξυπηρέτησης | 0 |

| Κόμβος | Συντ. X | Συντ. Ψ | Κόμβος | Συντ. X | Συντ. Ψ |
|---|---|---|---|---|---|
| 1 | 30 | 40 | 7 | 21 | 47 |
| 2 | 37 | 52 | 8 | 17 | 63 |
| 3 | 49 | 49 | 9 | 31 | 62 |
| 4 | 52 | 64 | 10 | 52 | 33 |
| 5 | 20 | 26 | 11 | 51 | 21 |
| 6 | 40 | 30 | 12 | 42 | 41 |

| | | | | | |
|---|---|---|---|---|---|
| 13 | 31 | 32 | 33 | 38 | 46 |
| 14 | 5 | 25 | 34 | 46 | 10 |
| 15 | 12 | 42 | 35 | 61 | 33 |
| 16 | 36 | 16 | 36 | 62 | 63 |
| 17 | 52 | 41 | 37 | 38 | 46 |
| 18 | 27 | 23 | 38 | 32 | 22 |
| 19 | 17 | 33 | 39 | 45 | 35 |
| 20 | 13 | 13 | 40 | 59 | 15 |
| 21 | 57 | 58 | 41 | 5 | 6 |
| 22 | 62 | 42 | 42 | 10 | 17 |
| 23 | 42 | 57 | 43 | 21 | 10 |
| 24 | 16 | 57 | 44 | 5 | 64 |
| 25 | 8 | 52 | 45 | 30 | 15 |
| 26 | 7 | 38 | 46 | 39 | 10 |
| 27 | 27 | 68 | 47 | 32 | 39 |
| 28 | 30 | 48 | 48 | 25 | 32 |
| 29 | 8 | 52 | 49 | 25 | 55 |
| 30 | 7 | 38 | 50 | 48 | 28 |
| 31 | 58 | 27 | 51 | 56 | 37 |
| 32 | 37 | 69 | | | |

*Πίνακας 3: Συντεταγμένες κόμβων προβλήματος C1*

Ο πίνακας συντεταγμένων παρουσιάζει σε ποιο σημείο ακριβώς βρίσκονται οι κόμβοι στο χώρο. Είναι δηλαδή μία δισδιάστατη απεικόνιση των κόμβων. Παρακάτω παρουσιάζεται ο πίνακας της ζήτησης ο οποίος περιέχει τη ζήτηση του κάθε κόμβου. Την ίδια μορφή έχουν όλα τα παραδείγματα που έχει τρέξει ο κώδικας. Ακολούθως παρουσιάζονται οι τιμές των μεταβλητών όλων των παραδειγμάτων καθώς και οι βέλτιστες τιμές κόστους αντιστοίχως.

| Κόμβος | Ζήτηση | Κόμβος | Ζήτηση | Κόμβος | Ζήτηση | Κόμβος | Ζήτηση |
|---|---|---|---|---|---|---|---|
| 1 | 0 | 14 | 23 | 27 | 7 | 40 | 14 |
| 2 | 7 | 15 | 21 | 28 | 15 | 41 | 7 |
| 3 | 30 | 16 | 10 | 29 | 14 | 42 | 27 |
| 4 | 16 | 17 | 15 | 30 | 6 | 43 | 13 |
| 5 | 9 | 18 | 3 | 31 | 19 | 44 | 11 |
| 6 | 21 | 19 | 41 | 32 | 11 | 45 | 16 |
| 7 | 15 | 20 | 9 | 33 | 12 | 46 | 10 |
| 8 | 19 | 21 | 28 | 34 | 23 | 47 | 5 |
| 9 | 23 | 22 | 8 | 35 | 26 | 48 | 25 |
| 10 | 11 | 23 | 8 | 36 | 17 | 49 | 17 |
| 11 | 5 | 24 | 16 | 37 | 6 | 50 | 18 |
| 12 | 19 | 25 | 10 | 38 | 9 | 51 | 10 |
| 13 | 29 | 26 | 28 | 39 | 15 | | |

*Πίνακας 4: Ζήτηση κόμβων προβλήματος C1*

Στον παρακάτω πίνακα, η πρώτη στήλη περιέχει το πρόβλημα, η δεύτερη τον αριθμό των κόμβων του κάθε προβλήματος, η τρίτη τη χωρητικότητα των οχημάτων, η τέταρτη το μέγιστο μήκος διαδρομής (max tour length ή mtl), η πέμπτη το χρόνο εξυπηρέτησης (service time ή st) και τέλος η έκτη την τιμή του καλύτερου κόστους του προβλήματος OVRP που έχει επιλυθεί με τον αλγόριθμο HBMO.

| Number | N | Capacity | mtl | st | Best Cost |
|---|---|---|---|---|---|
| C1 | 51 | 160 | ∞ | 0 | **416.06** |
| C2 | 76 | 140 | ∞ | 0 | **567.14** |
| C3 | 101 | 200 | ∞ | 0 | **640.25** |
| C4 | 151 | 200 | ∞ | 0 | **738.49** |
| C5 | 200 | 200 | ∞ | 0 | **902.17** |

| | | | | | |
|---|---|---|---|---|---|
| C6 | 51 | 160 | 180 | 10 | **412.96** |
| C7 | 76 | 140 | 144 | 10 | **583.19** |
| C8 | 101 | 200 | 207 | 10 | **644.63** |
| C9 | 151 | 200 | 180 | 10 | **765.95** |
| C10 | 200 | 200 | 180 | 10 | **884.28** |
| C11 | 121 | 200 | ∞ | 0 | **683.15** |
| C12 | 101 | 200 | ∞ | 0 | **536.37** |
| C13 | 121 | 200 | 648 | 50 | **905.18** |
| C14 | 101 | 200 | 936 | 90 | **593.95** |

*Πίνακας 5: Δεδομένα Προβλημάτων*

## *Επιλογή Αρχικού Πληθυσμού*

Αρχικά, στον αλγόριθμο που δημιουργήθηκε, έγινε ένα πλήθος από εκτελέσεις τριών περιπτώσεων. Η πρώτη περίπτωση βασιζόταν σε ένα αρχικό πληθυσμό ο οποίος είχε δημιουργηθεί από τον *Αλγόριθμο του Πλησιέστερου Γείτονα* και από τη *Διαδικασία Άπληστης Τυχαιοποιημένης Προσαρμοστικής Αναζήτησης*, όπως έχει ήδη περιγράφει στο Κεφάλαιο 2ο. Ο αρχικός πληθυσμός της δεύτερης περίπτωσης προέκυψε από τον *Αλγόριθμο Πλησιέστερης Εισαγωγής* και από τη *Διαδικασία Άπληστης Τυχαιοποιημένης Προσαρμοστικής Αναζήτησης* εφαρμοσμένη στον Αλγόριθμο Πλησιέστερης Εισαγωγής. Τέλος, ο αρχικός πληθυσμός της τρίτης περίπτωσης ήταν αποτέλεσμα του συνδυασμού της πρώτης και της δεύτερης περίπτωσης. Έτσι, μετά από αλλεπάλληλες εκτελέσεις του κώδικα, αντίστοιχα σε κάθε μία από τις αναφερθέντες περιπτώσεις, προέκυψε το συμπέρασμα ότι ο συνδυασμός τους αποδίδει καλύτερες (δηλαδή χαμηλότερες) τιμές κόστους. Γι' αυτό και οι επόμενες εκτελέσεις, με σκοπό τη διερεύνηση της αποτελεσματικότητας του κώδικα, βασίστηκαν στην τρίτη περίπτωση, δηλαδή στην κατασκευή του αρχικού πληθυσμού με το συνδυασμό των δύο πρώτων περιπτώσεων. Σε αυτό το σημείο πρέπει να αναφερθεί ότι κρίνεται σημαντικό το στάδιο κατασκευής αρχικών λύσεων, δηλαδή η αποτελεσματικότητα του τελικού κώδικα εξαρτάται από το σημείο αυτό σημαντικά. Επίσης, στο στάδιο αυτό πρέπει να υπάρχει ποικιλομορφία, δηλαδή να υπάρχουν διαφορετικές αρχικές λύσεις ώστε ο αλγόριθμος

να είναι πιο ευέλικτος στον δισδιάστατο χώρο και να μην εγκλωβίζεται σε τοπικά ακρότατα. Ο εγκλωβισμός αυτός στέκεται εμπόδιο στον κώδικα για την εύρεση της βέλτιστης λύσης. Αυτό λοιπόν, το να μην εγκλωβίζεται, επιτυγχάνεται με την κατασκευή των αρχικών λύσεων με διάφορες μεθόδους. Επιπλέον, πέρα από την ευελιξία του αλγόριθμου και από το έχει '*καλές*' αρχικές λύσεις, είναι εξίσου ουσιώδες ο αλγόριθμος να έχει την ικανότητα με τις μεθόδους τοπικής αναζήτησης να *βελτιώνει – μειώνει* το κόστος κατά τη διάρκεια εκτέλεσης.

Μετά από σειρά εκτελέσεων σε κάθε παράδειγμα σημειώθηκαν οι βέλτιστες τιμές σε κάθε παράδειγμα (3η στήλη) καθώς και η απόκλιση τους από τα βέλτιστα που έχουν σημειωθεί γενικά (2η στήλη). Η απόκλιση κάθε βέλτιστης τιμής των εκτελέσεων από τη βέλτιστη τιμή που έχει σημειωθεί έως τώρα παρουσιάζεται στην 4η στήλη (Var%) και στην 5η στήλη έχει σημειωθεί ο μέσος όρος των βέλτιστων τιμών των εκτελέσεων που έγιναν. Στον παρακάτω πίνακα παρουσιάζονται αναλυτικά οι τιμές:

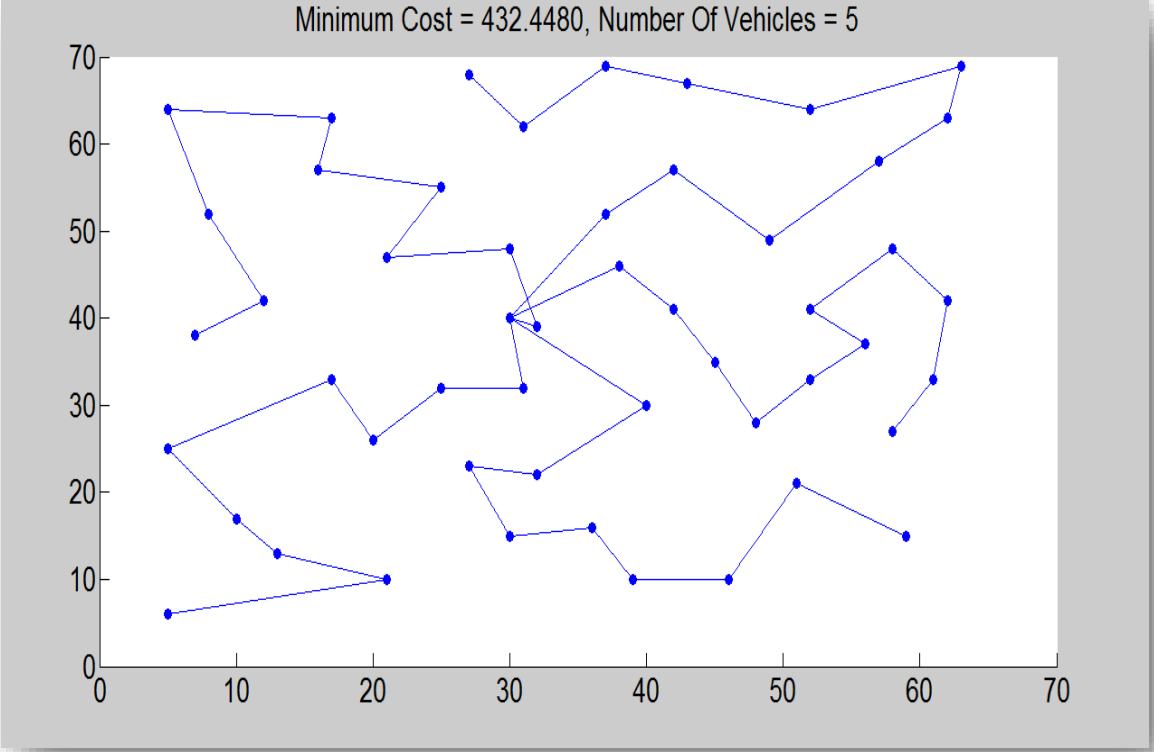
| Number | Best Cost | Best Cost | Var% | Average |
|---|---|---|---|---|
| C1 | **416.06** | *432.448* | **3,9%** | 446.8926667 |
| C2 | **567.14** | *600.6228* | **5,9%** | 611.5241667 |
| C3 | **640.25** | *704.1699* | **10,7%** | 707.4438333 |
| C4 | **738.49** | *856.4446* | **16,8%** | 870.0424 |
| C5 | **902.17** | *994.9932* | **11,3%** | 1.02E+03 |
| C6 | **412.96** | *431.1457* | **4,4%** | 434.691633 |
| C7 | **583.19** | *618.0772* | **5,9%** | 626.5154 |
| C8 | **644.63** | *697.3865* | **8,1%** | 710.8659 |
| C9 | **765.95** | *878.1053* | **15,8%** | 894.6162 |
| C10 | **884.28** | *1.03E+03* | **17,6%** | 1.05E+03 |
| C11 | **683.15** | *719.2229* | **13,3%** | 719.8932667 |
| C12 | **536.37** | *549.1568* | **2,7%** | 560.3155 |
| C13 | **905.18** | *939.0204* | **4%** | 941.5995 |
| C14 | **593.95** | *612.8292* | **3,5%** | 617.0472 |

*Πίνακας 6: Αποτελέσματα*

## 3.1 Γραφική απεικόνιση λύσεων

Στη συνέχεια θα παρουσιαστούν τα διαγράμματα (plots) της Matlab στα οποία γίνεται δισδιάστατη αναπαράσταση των διαδρομών κάθε βέλτιστης λύσης. Κάτω από κάθε διάγραμμα ακολουθεί και ο πίνακας των διαδρομών κάθε οχήματος.

**Παράδειγμα C1**

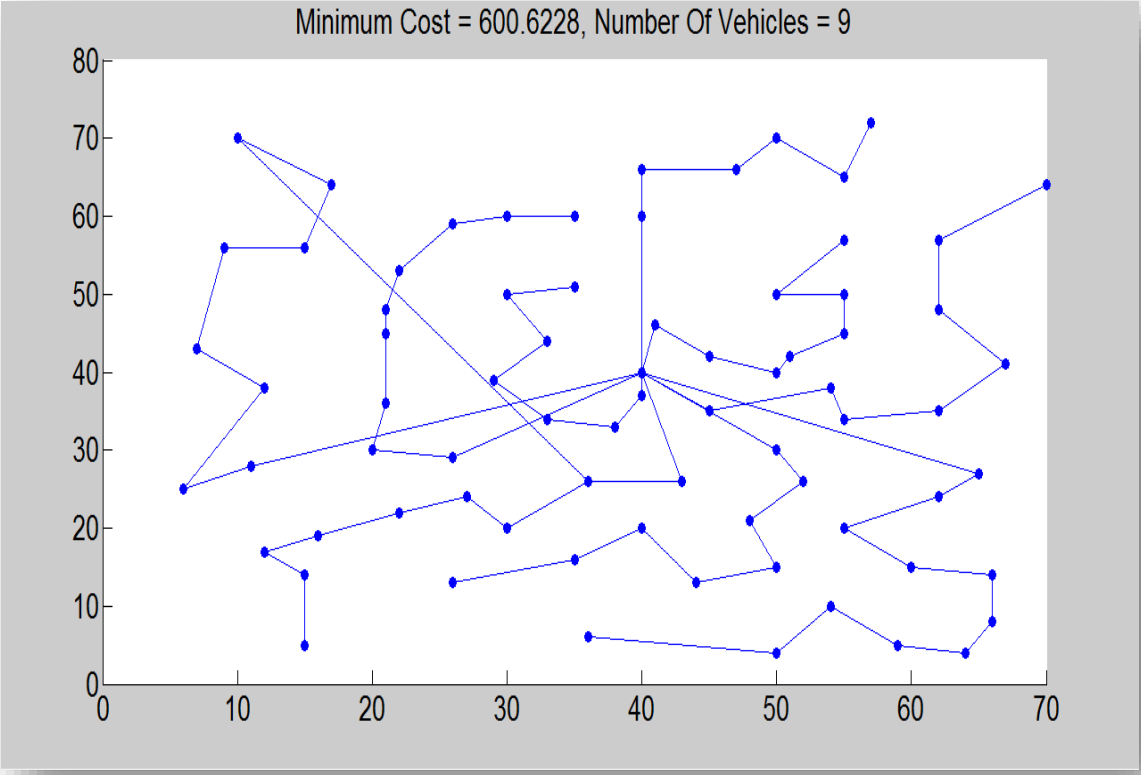



*Εικόνα 7: Αναπαράσταση της βέλτιστης λύσης του 1ου παραδείγματος*

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

*Πίνακας 7: Διαδρομές βέλτιστης λύσης 1ου παραδείγματος*

**Παράδειγμα C2**

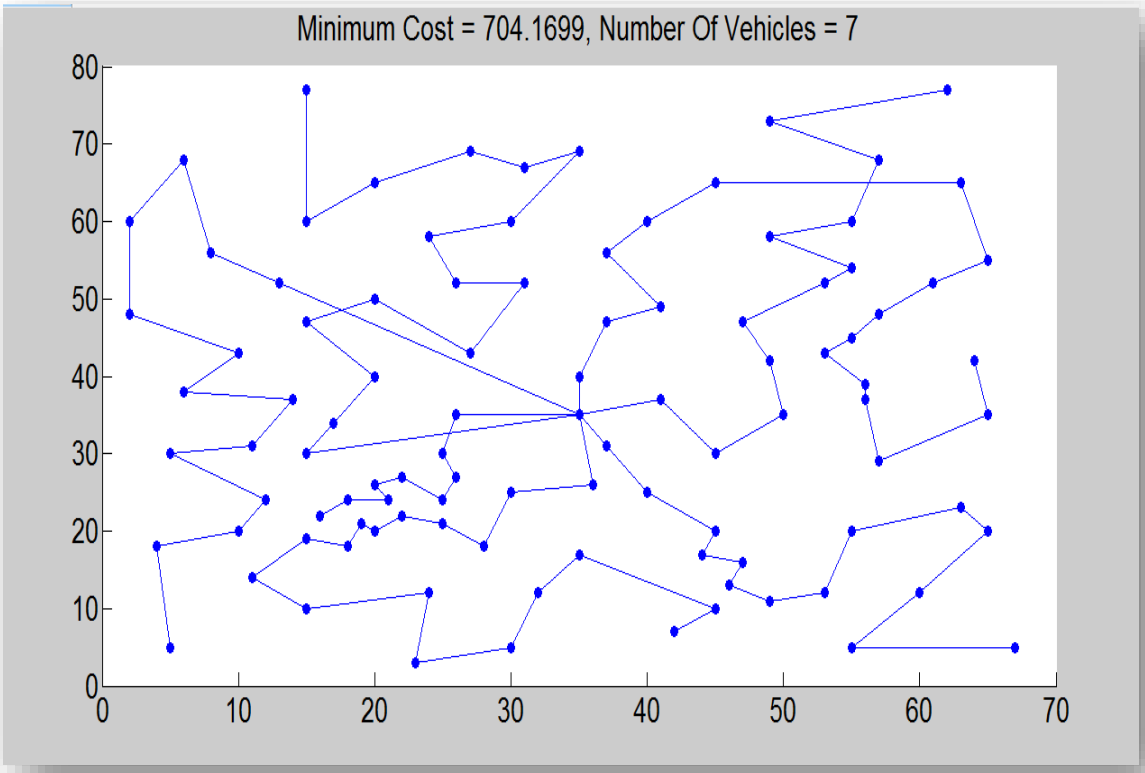



*Εικόνα 8: Αναπαράσταση της βέλτιστης λύσης του 2ου παραδείγματος*

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

*Πίνακας 8: Διαδρομές βέλτιστης λύσης 2ου παραδείγματος*

**Παράδειγμα C3**

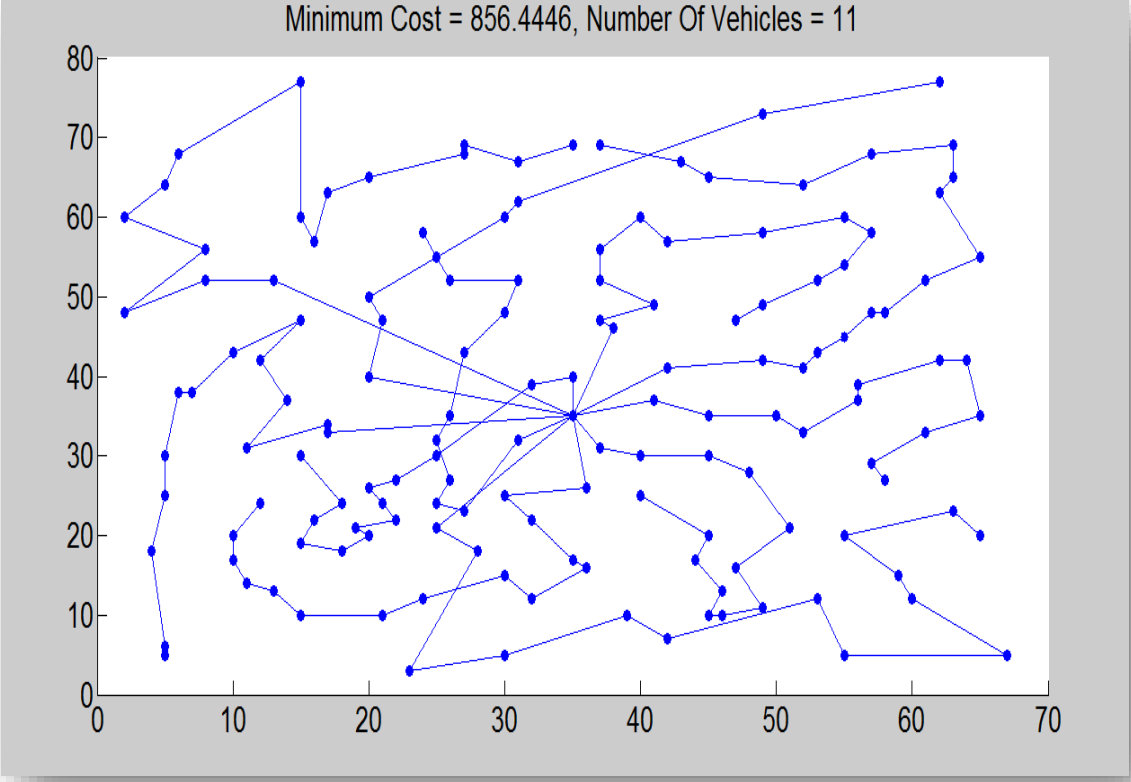



*Εικόνα 9: Αναπαράσταση της βέλτιστης λύσης του 3ου παραδείγματος*

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

*Πίνακας 9: Διαδρομές βέλτιστης λύσης 3ου παραδείγματος*

**Παράδειγμα C4**

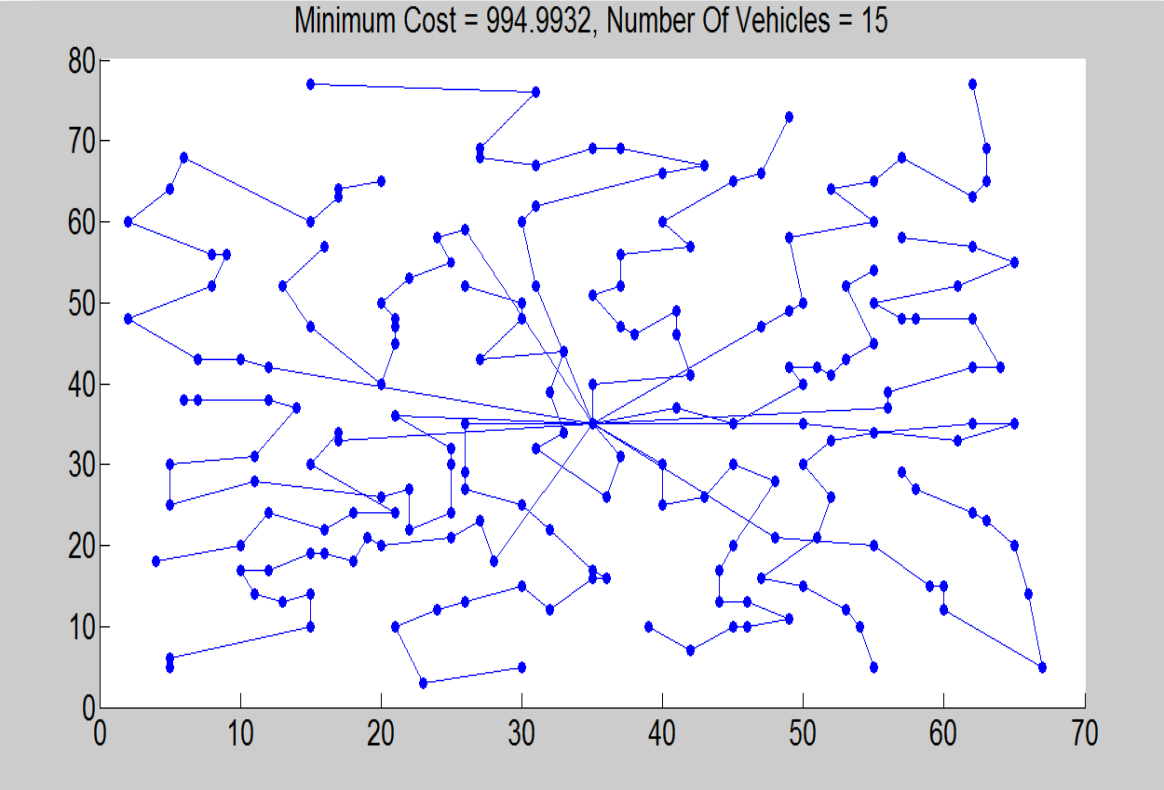



*Εικόνα 10: Αναπαράσταση της βέλτιστης λύσης του 4ου παραδείγματος*

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

*Πίνακας 10: Διαδρομές βέλτιστης λύσης 4ου παραδείγματος*

**Παράδειγμα C5**

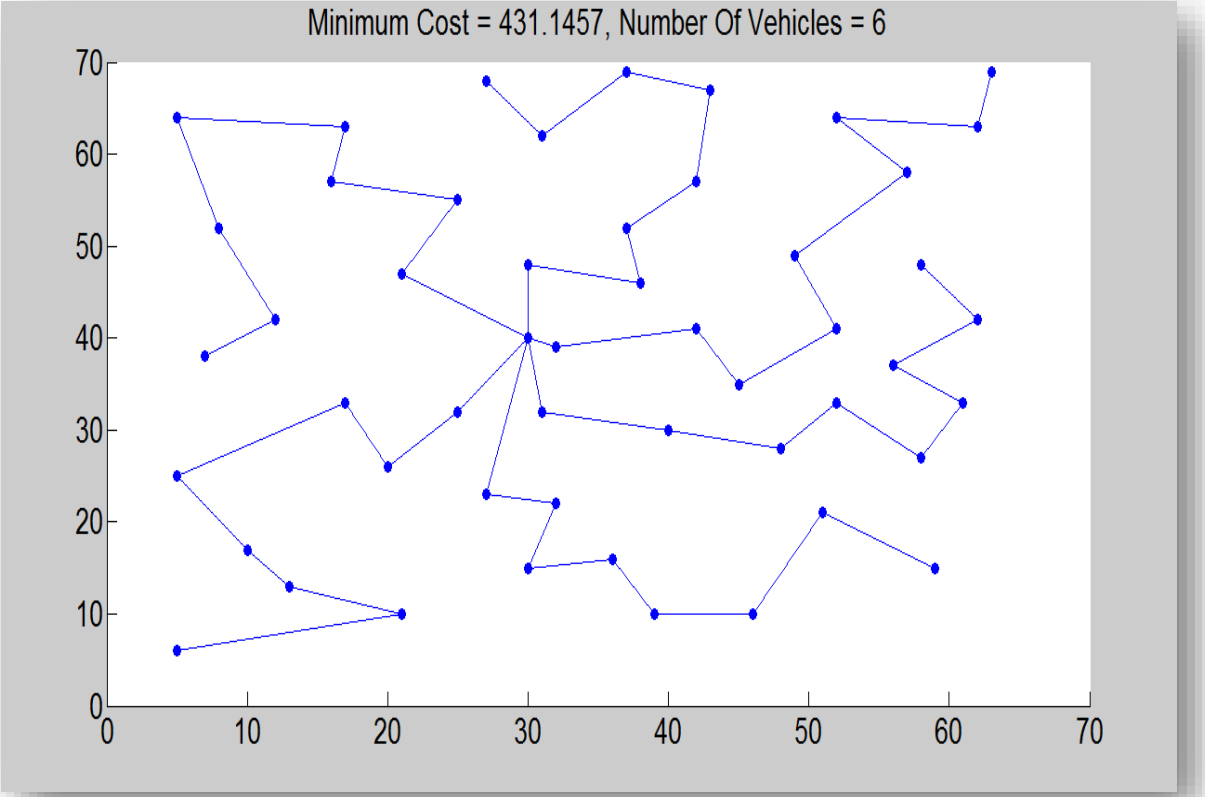



*Εικόνα 11: Αναπαράσταση της βέλτιστης λύσης του 5ου παραδείγματος*

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

*Πίνακας 11: Διαδρομές βέλτιστης λύσης 5ου παραδείγματος*

**Παράδειγμα C6**

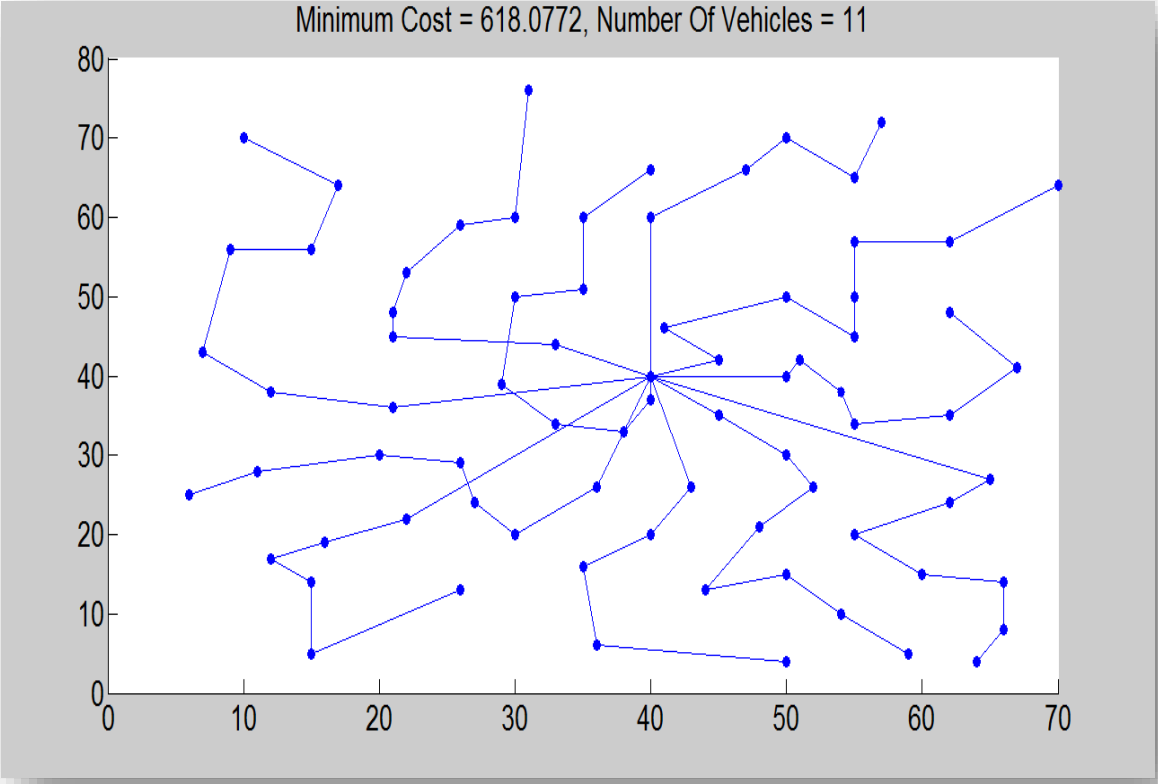



*Εικόνα 12: Αναπαράσταση της βέλτιστης λύσης του 6ου παραδείγματος*

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

*Πίνακας 12: Διαδρομές βέλτιστης λύσης 6ου παραδείγματος*

**Παράδειγμα C7**

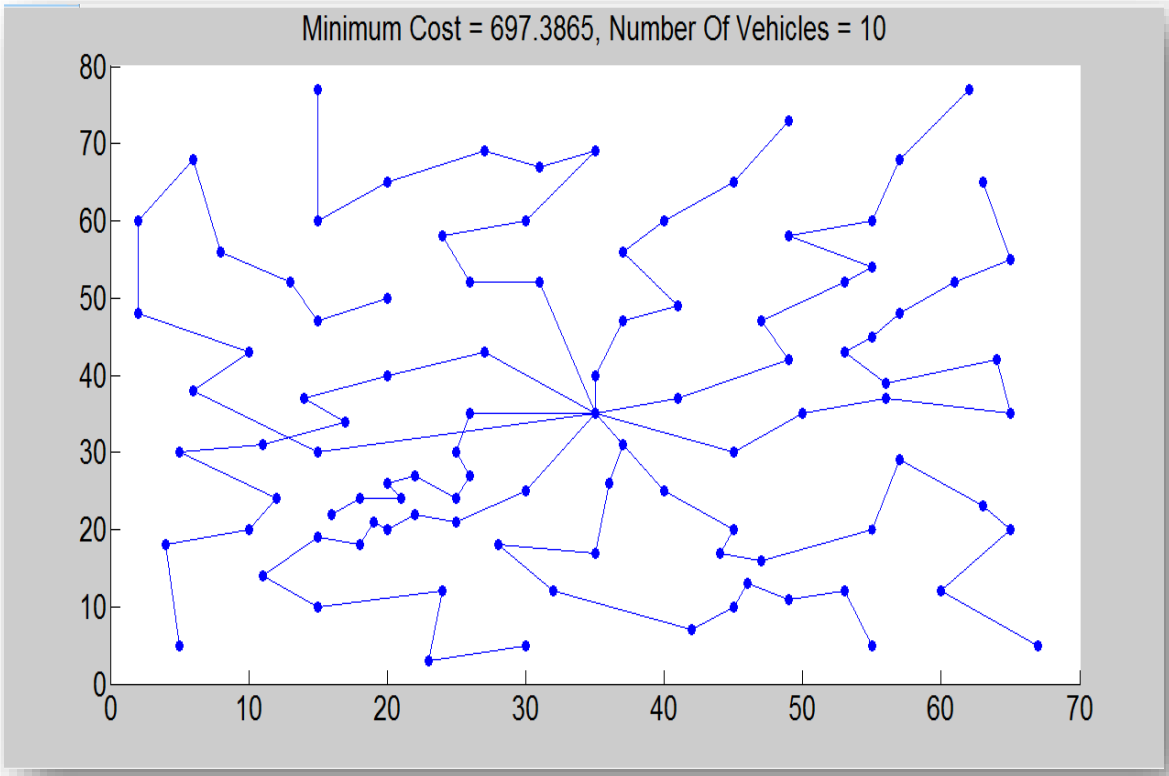



*Εικόνα 13: Αναπαράσταση της βέλτιστης λύσης του 7ου παραδείγματος*

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

*Πίνακας 13: Διαδρομές βέλτιστης λύσης 7ου παραδείγματος*

**Παράδειγμα C8**

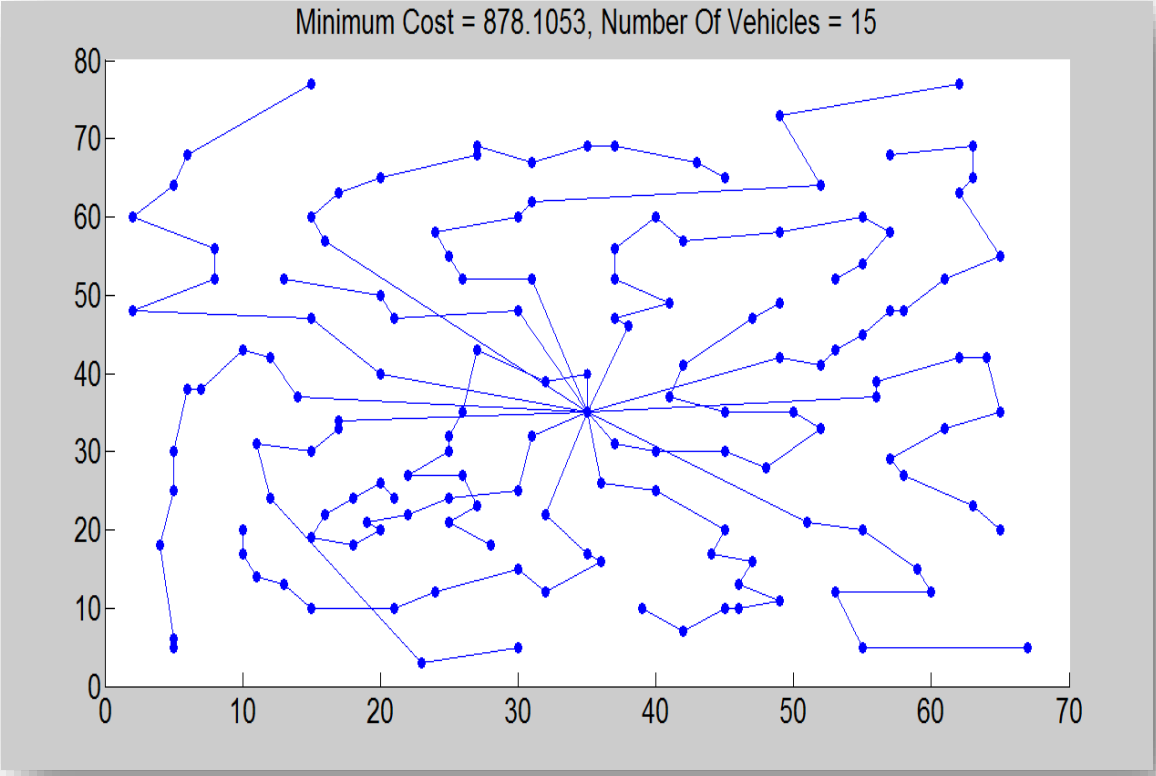



*Εικόνα 14: Αναπαράσταση της βέλτιστης λύσης του 8ου παραδείγματος*

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

*Πίνακας 14: Διαδρομές βέλτιστης λύσης 8ου παραδείγματος*

**Παράδειγμα C9**

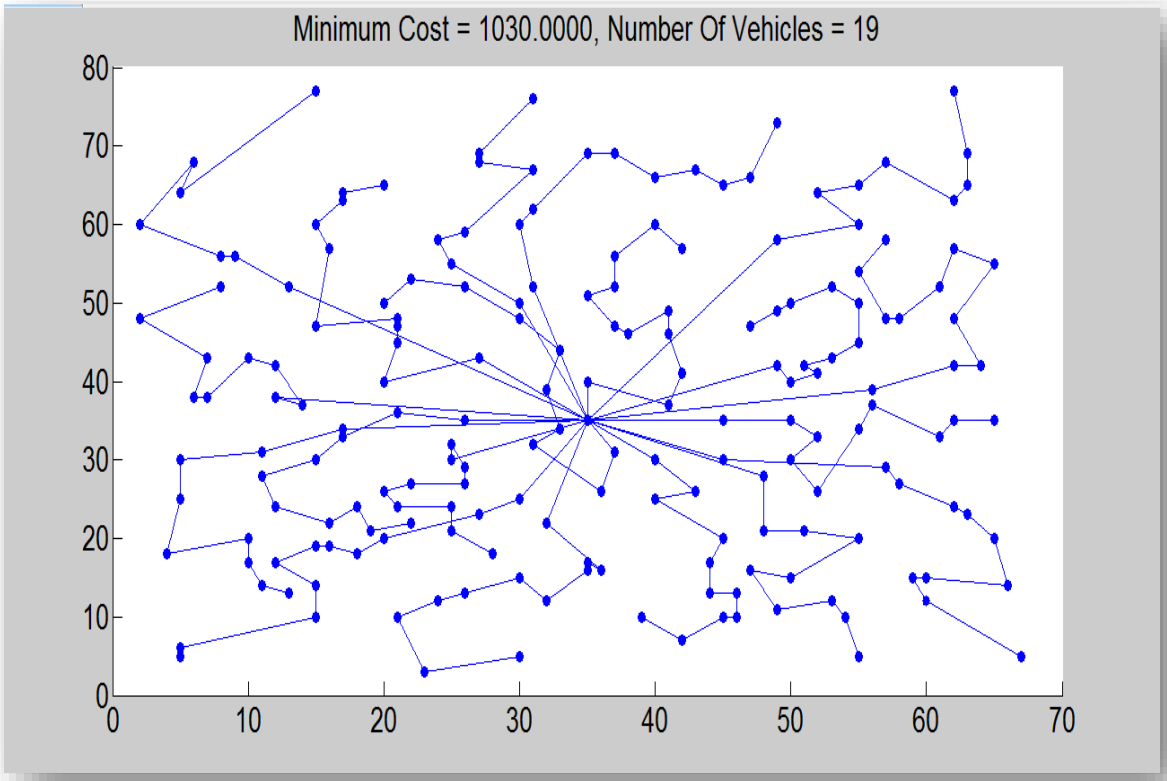



*Εικόνα 15: Αναπαράσταση της βέλτιστης λύσης του 9ου παραδείγματος*

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

*Πίνακας 15: Διαδρομές βέλτιστης λύσης 9ου παραδείγματος*

**Παράδειγμα C10**

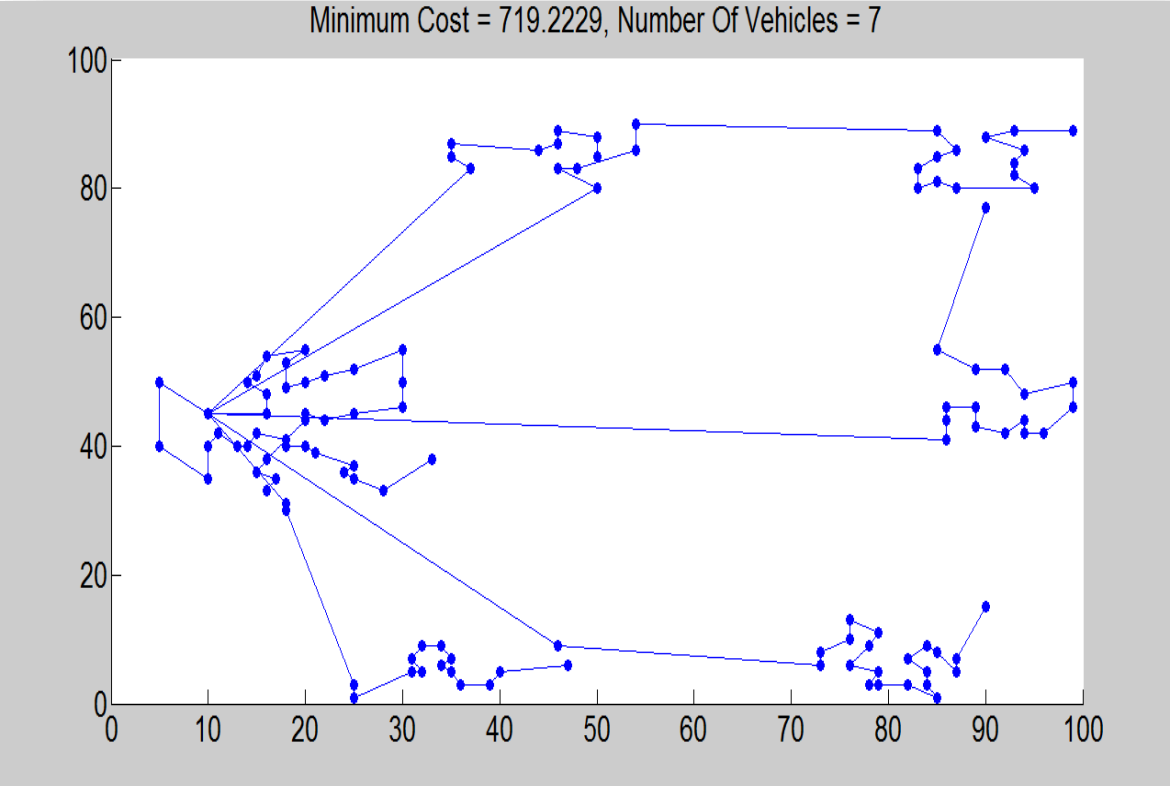



*Εικόνα 16: Αναπαράσταση της βέλτιστης λύσης του 10ου παραδείγματος*

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

*Πίνακας 16: Διαδρομές βέλτιστης λύσης 10ου παραδείγματος*

**Παράδειγμα C11**



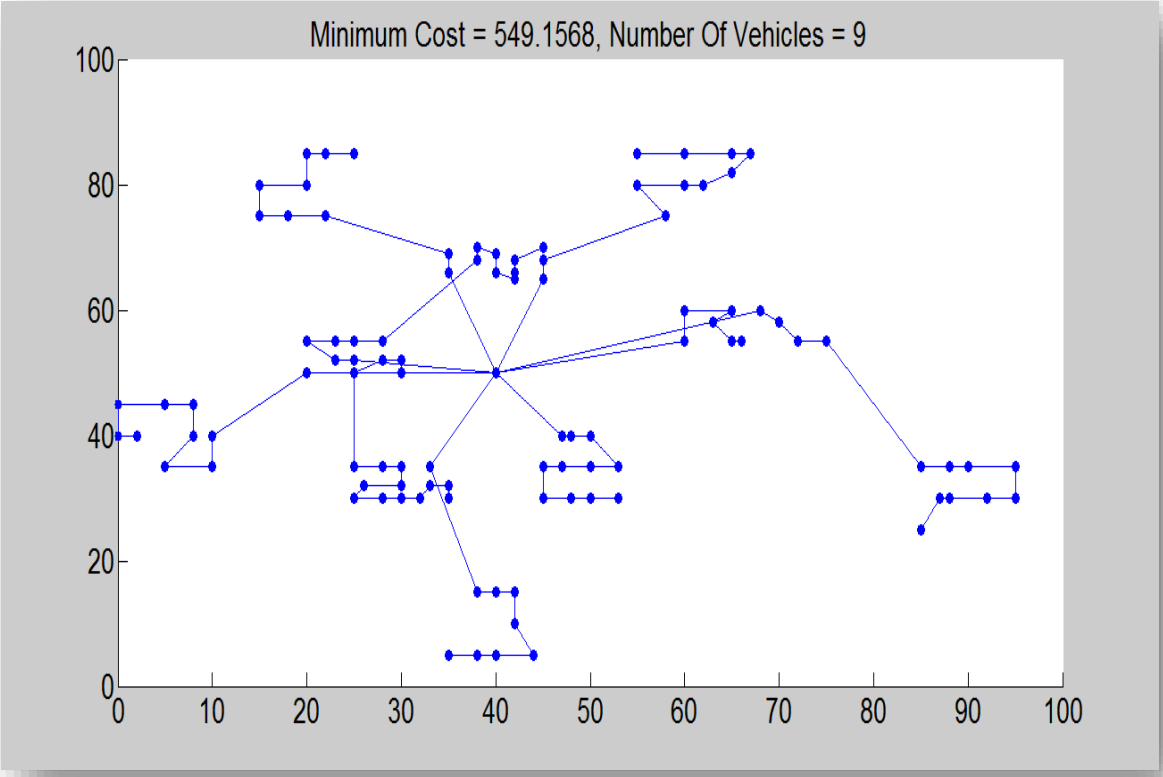

*Εικόνα 17: Αναπαράσταση της βέλτιστης λύσης του 11ου παραδείγματος*

| | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **1** | 121 | 120 | 82 | 83 | 89 | 112 | 87 | 88 | 93 | 90 | 92 |
| | 91 | 115 | 19 | 119 | 109 | 110 | | | | | |
| **1** | 96 | 103 | 106 | 107 | 108 | 104 | 105 | 102 | 100 | 101 | 117 |
| | 99 | 111 | 116 | 98 | 95 | 97 | 94 | 86 | 113 | 85 | 118 |
| **1** | 114 | 84 | 3 | 2 | 4 | 5 | 6 | 7 | 8 | 10 | 11 |
| | 12 | 16 | 15 | 14 | 13 | | | | | | |
| **1** | 9 | 17 | 18 | 21 | 22 | 27 | 24 | 20 | 26 | 23 | 25 |
| | 28 | 34 | 31 | 32 | 29 | 33 | 36 | 35 | 37 | 30 | |
| **1** | 38 | 39 | 40 | 43 | 42 | 45 | 48 | 47 | 50 | 51 | 52 |
| | 49 | 46 | 44 | 41 | 60 | | | | | | |

| | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **1** | 69 | 74 | 77 | 80 | 81 | 57 | 59 | 56 | 54 | 53 | 55 |
| | 58 | 66 | 62 | 63 | 65 | 61 | 64 | 67 | | | |
| **1** | 68 | 70 | 71 | 72 | 75 | 73 | 76 | 79 | 78 | | |

*Πίνακας 17: Διαδρομές βέλτιστης λύσης 11ου παραδείγματος*

**Παράδειγμα C12**

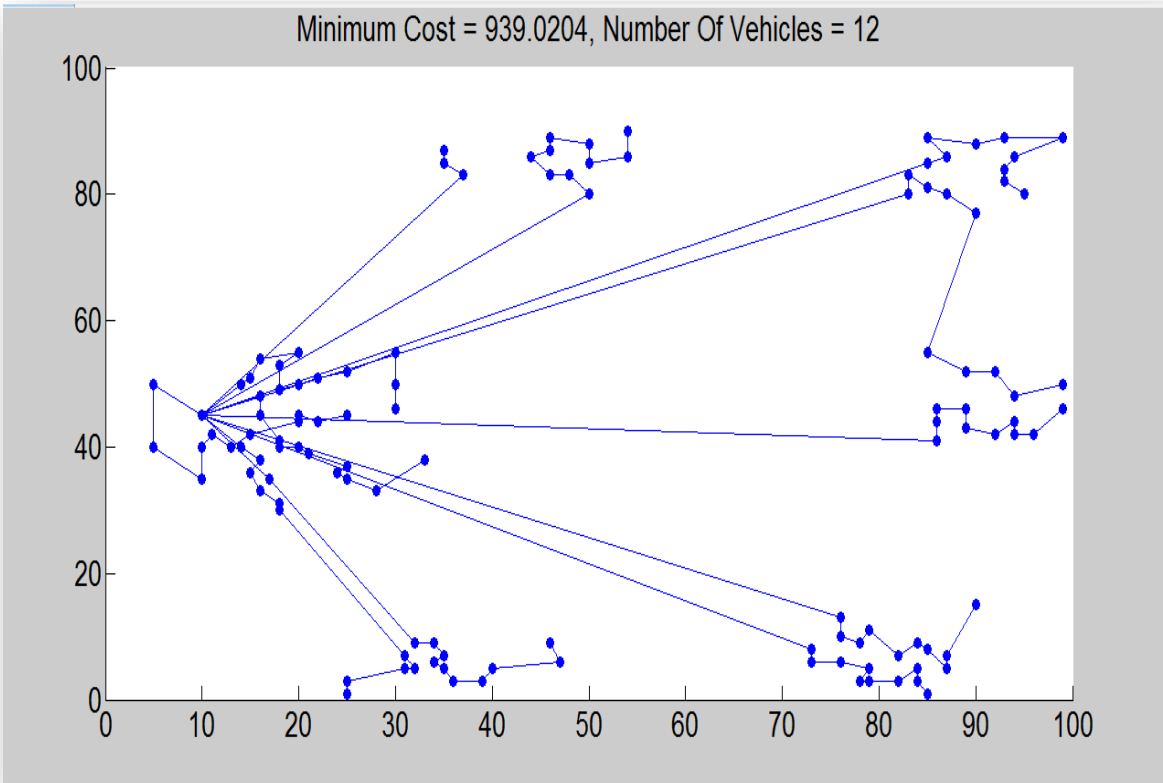



*Εικόνα 18: Αναπαράσταση της βέλτιστης λύσης του 12ου παραδείγματος*

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

*Πίνακας 18: Διαδρομές βέλτιστης λύσης 12ου παραδείγματος*

**Παράδειγμα C13**

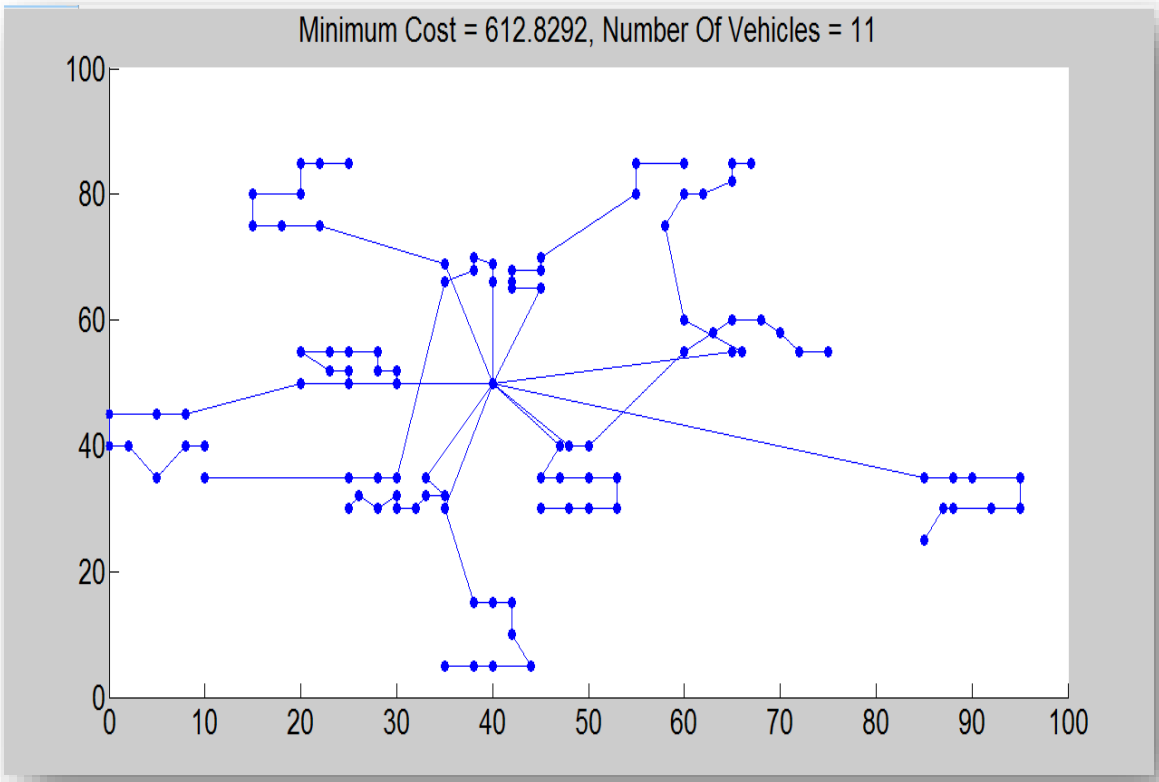



*Εικόνα 19: Αναπαράσταση της βέλτιστης λύσης του 13ου παραδείγματος*

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

*Πίνακας 19: Διαδρομές βέλτιστης λύσης 13ου παραδείγματος*

**Παράδειγμα C14**




*Εικόνα 20: Αναπαράσταση της βέλτιστης λύσης του 14ου παραδείγματος*

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

*Πίνακας 20: Διαδρομές βέλτιστης λύσης 14ου παραδείγματος*

## 3.2 Συμπεράσματα και σχόλια

Η παρούσα διπλωματική εργασία ασχολείται με το ανοιχτό πρόβλημα δρομολόγησης οχημάτων (OVRP) που επιλύθηκε με τον αλγόριθμο βελτιστοποίησης ζευγαρώματος μελισσών (HBMO). Για τον έλεγχο της αποτελεσματικότητας του αλγορίθμου χρησιμοποιήθηκαν 14 παραδείγματα, τα οποία εκτελέστηκαν στο προγραμματιστικό περιβάλλον της Matlab, των οποίων οι βέλτιστες τιμές κόστους είναι γνωστές. Μετά από ένα σύνολο εκτελέσεων, των οποίων τα αποτελέσματα είναι καταγεγραμμένα παραπάνω, θα μπορούσαμε να πούμε ότι ο αλγόριθμος έδωσε σχετικά ικανοποιητικά αποτελέσματα. Σε αυτό το σημείο πρέπει να διευκρινιστεί μία σημαντική διαφορά της παρούσας έρευνας συγκριτικά με άλλες. Η διαφορά αυτή είναι ότι η βελτιστοποίηση της λύσης (κόστους) μέσα στον κώδικα, που γινόταν με τις μεθόδους τοπικής αναζήτησης σε κάθε επανάληψη, εκτελούταν στο πρόβλημα του πλανόδιου πωλητή TSP κι όχι σε αυτό της δρομολόγησης οχημάτων OVRP (οι περισσότερες έρευνες εκτελούν τη βελτιστοποίηση στο πρόβλημα δρομολόγησης οχημάτων). Να υπενθυμίσουμε ότι από τη λύση του πλανόδιου πωλητή έπειτα προχωρούσαμε στη διαίρεση των διαδρομών (δρομολόγησης οχημάτων) κι αμέσως μετά στον υπολογισμό του κόστους. Αυτό είχε ως συνέπεια τη δυσκολία εύρεσης καλύτερης λύσης. Αυτό μπορεί να το συμπεράνει κανείς αν παρατηρήσει τα παραπάνω γραφήματα. Σε αρκετές διαδρομές δεν υπάρχει λογική συνοχή των τόξων αφού υπάρχουν κόμβοι οι οποίοι είναι ολοφάνερο ότι θα έπρεπε να ανήκουν σε διαφορετική διαδρομή. Παρόλα αυτά κάποιων αποτελεσμάτων οι αποκλίσεις ήταν πολύ μικρές - δεν ξεπέρασαν το 3% - καθώς υπήρχαν βέβαια και αποκλίσεις που ξεπέρασαν το 15%. Αν χωρίζαμε σε δύο κατηγορίες τα αποτελέσματα, αυτών που η απόκλιση δεν ξεπέρασε το 10% και αυτών που το ξεπέρασε θα μπορούσε κανείς να ισχυριστεί ότι τα παραδείγματα με τον μικρότερο πληθυσμό είχαν μικρότερη απόκλιση έναντι αυτών με μεγαλύτερο πληθυσμό.

ΒΙΒΛΙΟΓΡΑΦΙΑ

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

[2] Ιωάννης Μαρινάκης, Μαγδαληνή Μαρινάκη, Εξελικτικοί Αλγόριθμοι και Βελτιστοποίηση Συστημάτων Μεγάλης Κλίμακας

[3] Ιωάννης Μαρινάκης, Μαγδαληνή Μαρινάκη, Νικόλαος Ματσατσίνης, Κωνσταντίνος Ζοπουνίδης, Μεθευρετικοί και Εξελικτικοί Αλγόριθμοι σε Προβλήματα Διοικητικής Επιστήμης, εκδόσεις Κλειδάριθμος, 2011

[4] Douglas, M., Martha C. Cooper, Janus D. Pagh, 1998, "Supply Chain Management: Implementation Issues and Research Opportunities", The International journal of Logistics Management

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

[6] Παληοθόδωρος Ιωάννης, Ο αλγόριθμος των μελισσών και οι εφαρμογές του, Πανεπιστήμιο Πειραιώς, Φεβρουάριος 2012, Μεταπτυχιακή Διατριβή

[7] Παπαδόπουλος Αθανάσιος, "Το Πρόβλημα της Δρομολόγησης Στόλου Οχημάτων: Μελέτη Περίπτωσης", Πανεπιστήμιο Πατρών-Μεταπτυχιακό στη Διοίκηση Επιχειρήσεων (MBA), Σεπτέμβριος 2015, Μεταπτυχιακή Εργασία

[8] Χρήστος Δρόσος, Αποτίμηση τεχνολογιών μεθόδων και αλγορίθμων ελέγχου μη επανδρωμένων αεροσκαφών, Πανεπιστήμιο Πειραιώς, Οκτώβριος 2011, Μεταπτυχιακή Διατριβή

[9] https://el.wikipedia.org/wiki/%CE%9D%CE%BF%CE%B7%CE%BC%CE%BF%CF%83%CF%8D%CE%BD%CE%B7_%CF%83%CE%BC%CE%AE%CE%BD%CE%BF%CF%85%CF%82

[10]
https://el.wikiversity.org/wiki/%CE%95%CF%86%CE%BF%CE%B4%CE%B9%CE%B1%CF%83%CF%84%CE%B9%CE%BA%CE%AE_%CE%91%CE%BB%CF%85%CF%83%CE%AF%CE%B4%CE%B1_%CE%A3%CF%87%CE%B5%CE%B4%CE%B9%CE%B1%CF%83%CE%BC%CE%BF%CF%8D_%CE%A3%CF%85%CF%83%CF%84%CE%B7%CE%BC%CE%AC%CF%84%CF%89%CE%BD

[11]
http://www.terpconnect.umd.edu/~bgolden/recent_presentation_pdfs_links/2005_nov_ovrp_sanfran.pdf