**Πολυτεχνείο Κρήτης**

**Σχολή Μηχανικών Παραγωγής και Διοίκησης**

# Υλοποίηση και συγκριτική ανάλυση αλγορίθμων εμπνευσμένων από τη φύση για το πρόβλημα δρομολόγησης οχημάτων με περιορισμό χωρητικότητας και χρόνους εξυπηρέτησης πελατών

**Διπλωματική Εργασία**

**Παππάς Ξενοφών**

Επιβλέπων

Ιωάννης Μαρινάκης

**Χανιά, Οκτώβριος 2021**

**Πολυτεχνείο Κρήτης**

**Σχολή Μηχανικών Παραγωγής και Διοίκησης**

# Υλοποίηση και συγκριτική ανάλυση αλγορίθμων εμπνευσμένων από τη φύση για το πρόβλημα δρομολόγησης οχημάτων με περιορισμό χωρητικότητας και χρόνους εξυπηρέτησης πελατών

**Διπλωματική Εργασία**

**Παππάς Ξενοφών**

Εξεταστική Επιτροπή

Ιωάννης Μαρινάκης, Αναπληρωτής Καθηγητής (Επιβλέπων)

Νικόλαος Ματσατσίνης, Καθηγητής

Μαγδαληνή Μαρινάκη, ΕΔΙΠ

**Χανιά, Οκτώβριος 2021**

Πολυτεχνείο Κρήτης

**Technical University of Crete**

**School of Production Engineering and Management**

# Implementation and comparative analysis of nature inspired algorithms for the capacitated vehicle routing problem

**Diploma Thesis**

**Pappas Xenofon**

Supervisor

Yannis Marinakis

**Chania, October 2021**

Πολυτεχνείο Κρήτης

# Περίληψη

Σκοπός της τρέχουσας διπλωματικής είναι η υλοποίηση και συγκριτική ανάλυση αλγορίθμων εμπνευσμένων από τη φύση για το πρόβλημα δρομολόγησης οχημάτων με περιορισμό χωρητικότητας και χρόνους εξυπηρέτησης πελατών. Το Πρόβλημα Δρομολόγησης Οχημάτων αποτελεί ένα από τα πλέον σημαντικά προβλήματα της Συνδυαστικής Βελτιστοποίησης. Το πρόβλημα εξετάζει τη βέλτιστη δρομολόγηση οχημάτων κατά την παράδοση ή/και παραλαβή προϊόντων στους/από τους πελάτες, σε δεδομένη χρονική περίοδο, στα πλαίσια επιχειρησιακών δραστηριοτήτων. Κατά την εκπόνηση της παρούσας εργασίας υλοποιούνται αλγόριθμοι βασισμένοι στη συμπεριφορά, στους εξελικτικούς μηχανισμούς, καθώς και στις διεργασίες της φυσικής επιλογής. Συγκεκριμένα, οι αλγόριθμοι που υλοποιούνται επικεντρώνονται στη νοημοσύνη σμήνους. Τα συστήματα νοημοσύνης Σμήνους αποτελούνται από έναν πληθυσμό απλών οντοτήτων που αλληλοεπιδρούν τοπικά μεταξύ τους και με το περιβάλλον τους για να εκτελέσουν κάποια ενέργεια. Οι οντότητες λειτουργούν αυτόνομα αλλά εμφανίζουν μια συλλογική συμπεριφορά, έχοντας ελάχιστους βασικούς κανόνες. Στην συνέχεια, οι αλγόριθμοι εφαρμόζονται σε διάφορα σετ δεδομένων για το πρόβλημα δρομολόγησης, όπου ελέγχεται η αποτελεσματικότητά τους. Τέλος, μετά την υλοποίηση, συγκρίνονται τα αποτελέσματα που προέκυψαν για καθέναν από τους αλγορίθμους ώστε να προκύψει ο αποτελεσματικότερος για την επίλυση του προβλήματος.



# Λέξεις Κλειδιά

Αλγόριθμος Βελτιστοποίησης Σμήνους Σωματιδίων, Αλγόριθμος Βελτιστοποίησης Γκρίζου Λύκου, Αλγόριθμος Βελτιστοποίησης Φάλαινας, Αλγόριθμος Βελτιστοποίησης Πεταλούδας Μονάρχη, Αλγόριθμος Νυχτοπεταλούδας, Αλγόριθμοι Νοημοσύνης Σμήνους, Πρόβλημα Δρομολόγησης Οχημάτων.

## Abstract

The purpose of the current diploma thesis is the implementation and comparative analysis of nature inspired algorithms for the capacitated vehicle routing problem. The Vehicle Routing Problem is one of the most important problems of Combinatorial Optimization. The problem examines the optimal routing of vehicles during the delivery and / or pickup of products to / from customers, in a given period of time, in the context of business activities. During the elaboration of the present implementation, algorithms based on behavior, evolutionary mechanisms, as well as the processes of physical selection are implemented. Specifically, the algorithms implemented focus on swarm intelligence. Swarm Intelligence Systems consist of a population of simple entities that interact locally with each other and with their environment to perform an action. Entities operate autonomously but exhibit a collective behavior with minimal basic rules. The algorithms are then applied to various data sets for the routing problem, where their effectiveness is tested. Finally, after the implementation, the results obtained for each one of the algorithms are compared in order to obtain the most efficient one for solving the problem.



## Key Words

Particle Swarm Optimization Algorithm, Grey Wolf Optimizer, Whale Optimization Algorithm, Monarch Butterfly Optimization, Moth Flame Optimizer, Swarm Intelligence Algorithms, Capacitated Vehicle Routing Problem.

# Περιεχόμενα

# Ευχαριστίες

Αρχικά, θα ήθελα να ευχαριστήσω τον κ. Μαρινάκη για την εμπιστοσύνη και τη συνεργασία μας σε όλο αυτό το διάστημα εκπόνησης της διπλωματικής εργασίας. Επίσης, θα ήθελα να ευχαριστήσω την πιο σημαντική πηγή στήριξης όλων αυτών των χρόνων την οικογένειά μου. Τέλος, μαζί με τα εφόδια που αποκόμισα κέρδισα και μοναδικούς φίλους που αποτέλεσαν ένα σημαντικό κεφάλαιο στήριξης όλα αυτά τα χρόνια.

# 1. Εισαγωγή

Η εφοδιαστική αλυσίδα αποτελεί τη διαδικασία προγραμματισμού, υλοποίησης και ελέγχου της διακίνησης αγαθών. Αποτελεί ίσως τον πιο σημαντικό τομέα της σύγχρονης παγκοσμιοποιημένης οικονομίας καθώς διασφαλίζει την έγκαιρη και ασφαλή παράδοση των αγαθών. Με σκοπό τη διαχείριση της εφοδιαστικής αλυσίδας έχουν αναπτυχθεί προβλήματα που προσομοιάζουν τις απαιτήσεις του σχεδιασμού του εφοδιασμού. Το βασικότερο πρόβλημα για αυτό το σκοπό είναι το πρόβλημα δρομολόγησης οχημάτων.

Στην παρούσα εργασία παρουσιάζονται τα αποτελέσματα αλγορίθμων εμπνευσμένων από την φύση για την επίλυση του προβλήματος δρομολόγησης οχημάτων με περιορισμό χωρητικότητας και χρόνους εξυπηρέτησης. Οι αυξημένες ανάγκες παραγωγής και διανομής αγαθών έχουν φέρει στο προσκήνιο τη σημαντικότητα της λειτουργίας της εφοδιαστικής αλυσίδας. Οι χρόνοι μεταφορών μειώνονται και η βελτιστοποίηση των μέσων εφοδιασμού αποτελεί ένα από τα βασικά ζητήματα που απασχολούν τις σύγχρονες ανάγκες της οικονομικής και κοινωνικής ζωής. Τα προβλήματα δρομολόγησης έρχονται να καλύψουν αυτήν την ανάγκη επιλύοντας τις κυριότερες μορφές απαιτήσεων που ανακύπτουν.

Ο τρόπος επίλυσης του προβλήματος που παρουσιάζεται στην συγκεκριμένη εργασία είναι με χρήση αλγορίθμων εμπνευσμένων από τη φύση. Οι συγκεκριμένοι αλγόριθμοι αποτελούν μία σύγχρονη προσέγγιση στην επίλυση των προβλημάτων βελτιστοποίησης. Βασίζονται στην νοημοσύνη σμήνους και ανάλογα με τα χαρακτηριστικά τους σχεδιάζονται, ώστε να ακολουθούν τις φυσικές διεργασίες οντοτήτων της φύσης.

Στο πρώτο κεφάλαιο της εργασίας θα παρουσιαστεί το πρόβλημα δρομολόγησης καθώς και βασικές παραλλαγές του. Ακόμα θα παρουσιαστεί και η μαθηματική μοντελοποίηση των προβλημάτων. Στο δεύτερο κεφάλαιο θα παρουσιαστούν οι αλγόριθμοι που εφαρμόσθηκαν καθώς και πληροφορίες σχετικά με τον τρόπο εφαρμογής τους. Στο επόμενο κεφάλαιο γίνεται παρουσίαση των μεθόδων τοπικής αναζήτησης που χρησιμοποιήθηκαν για τη περαιτέρω βελτίωση των αποτελεσμάτων που παρήχθησαν από την βασική εφαρμογή των αλγορίθμων. Τέλος, στο τελευταίο κεφάλαιο γίνεται παρουσίαση των τελικών αποτελεσμάτων, των διαδρομών καθώς και παρατίθεται η συγκριτική ανάλυση για την ανάδειξη του αποτελεσματικότερου αλγόριθμου.

# 2. Πρόβλημα Δρομολόγησης Οχημάτων με Περιορισμό Χωρητικότητας και Χρόνους Εξυπηρέτησης

## 2.1 Το πρόβλημα δρομολόγησης οχημάτων

Το Πρόβλημα της Δρομολόγησης Οχημάτων (VRP – Vehicle Routing Problem) παρουσιάστηκε για πρώτη φορά από τους Dantzing & Ramser το έτος 1959, ενώ σήμερα αποτελεί ένα από τα πιο σημαντικά και εφαρμόσιμα προβλήματα διανομής της εφοδιαστικής αλυσίδας. Το πρόβλημα δρομολόγησης οχημάτων αποτελεί επέκταση του προβλήματος του πλανόδιου πωλητή.

Οι πρώτες προσπάθειες επίλυσης του προβλήματος αναπτύχθηκαν από τους Dantzing & Ramser οι οποίοι δημιούργησαν την πρώτη αλγοριθμική προσέγγιση επίλυσης τέτοιων προβλημάτων. Στη συνέχεια οι Clarke & Wright (1964) πρότειναν έναν ευρετικό αλγόριθμο εξοικονόμησης, ο οποίος παρουσιάζει σημαντικές βελτιώσεις σε σχέση με αυτόν των Dantzing & Ramser. Με την πάροδο των χρόνων έχουν αναπτυχθεί πολλές μέθοδοι επίλυσης του προβλήματος.

Ο βασικός στόχος του προβλήματος δρομολόγησης οχημάτων (VRP – Vehicle Routing Problem) είναι η κατάλληλη δρομολόγηση ενός στόλου οχημάτων έτσι ώστε να διανύει τη μικρότερη δυνατή απόσταση χωρίς να παραβιάζονται περιορισμοί που διαφέρουν από πρόβλημα σε πρόβλημα. Όλες οι διαδρομές του προβλήματος έχουν αρχή και τέλος το ίδιο σημείο αφετηρίας, το όχημα επισκέπτεται μία φορά ένα υποσύνολο πελατών και όλοι οι πελάτες καλύπτονται από το υποσύνολο των διαδρομών. Οι περιορισμοί που εντάσσονται στο πρόβλημα έχουν να κάνουν με διαφορετικούς παράγοντες που μπορεί να αντιμετωπίσει κάποιος στην εφοδιαστική αλυσίδα. Μερικά παραδείγματα είναι η ζήτηση των πελατών, η ικανοποίηση ζήτησης σε συγκεκριμένο χρόνο, η πολυπλοκότητα των προϊόντων και οι συνθήκες διατήρησής τους, η περιορισμένη χωρητικότητα στο όχημα κ.α.

## 2.2 Χαρακτηριστικά του προβλήματος

Τα τρία βασικά ‘’συστατικά ‘’ του προβλήματος δρομολόγησης είναι οι πελάτες, τα οχήματα και οι περιορισμοί. Αρχικά, οι πελάτες όντας οι θέσεις παράδοσης των προϊόντων έχουν τα εξής στοιχεία: Μια συγκεκριμένη γεωγραφική θέση εντός του πεδίου λύσεων, μία ορισμένη ζήτηση από προϊόντα (demand), ένα

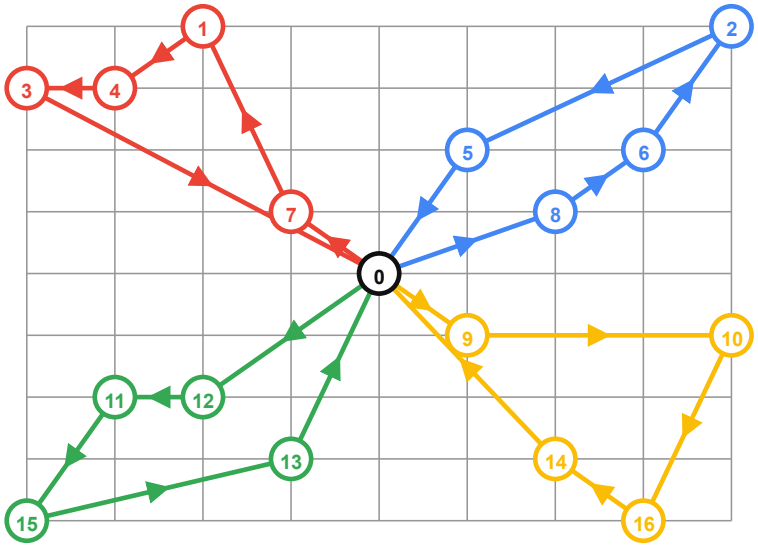
ορισμένο διάστημα εξυπηρέτησης το οποίο έχει να κάνει είτε με τη διαθεσιμότητά τους είτε και με τη φύση των αγαθών που τους παραδίδονται καθώς και ένα ορισμένο χρόνο που απαιτείται για να τους αποδοθούν τα προϊόντα. Όσο αφορά το δεύτερο συστατικό του προβλήματος, τα οχήματα, τα στοιχεία που θα πρέπει να έχουν για να οριστεί κατάλληλα το πρόβλημα είναι τα εξής: Ορισμένη θέση της αποθήκης και προσδιορισμός δυνατότητας ανεφοδιασμού από άλλη αποθήκη, η χωρητικότητα του οχήματος, προσδιορισμός αριθμού οχημάτων, εάν δεν είναι μόνο ένα, και προσδιορισμός χαρακτηριστικών τους, κόστος λειτουργίας του οχήματος. Τέλος, όσο αφορά τους περιορισμούς υπάρχουν πολλές παραλλαγές των χαρακτηριστικών παρόλα αυτά τα βασικά χαρακτηριστικά τους είναι: Το άνω όριο χρόνου που μπορεί ένα όχημα να βρίσκεται σε μία διαδρομή, η μη υπέρβαση ενός συγκεκριμένου ορίου χωρητικότητας του οχήματος, η σειρά εφοδιασμού των πελατών, η φύση της εξυπηρέτησης στον πελάτη και ειδικότερα αν αφορά διανομή ή παραλαβή προϊόντων κ.α.

## 2.3 Μοντελοποίηση του Προβλήματος

Η βασική ιδέα γύρω από το πρόβλημα δρομολόγησης οχημάτων είναι η επέκταση του προβλήματος του πλανόδιου πωλητή, βασισμένο στις ανάγκες της εφοδιαστικής αλυσίδας. Για τον ορισμό του απλού VRP θα πρέπει να ληφθούν υπόψιν όλα τα χαρακτηριστικά που διατρέχουν την εφοδιαστική αλυσίδα καθώς και οι ανάγκες – περιορισμοί σε τομείς διανομής – παράδοσης προϊόντων. Στην συγκεκριμένη ενότητα θα παρουσιαστούν τα χαρακτηριστικά του προβλήματος (2.2) ως μαθητικές σχέσεις. Αρχικά, για την μαθηματική μοντελοποίηση είναι αναγκαίο να αναφερθούν ως βασικά στοιχεία όλες οι παράμετροι του προβλήματος. Πιο συγκεκριμένα:

- Το οδικό δίκτυο σε ένα πραγματικό πρόβλημα, αναπαρίσταται με κόμβους και τόξα στο υπολογιστικό πρόβλημα δρομολόγησης. Κάθε τόξο αναπαριστά μία προσανατολισμένη διαδρομή και κάθε κόμβος αναπαριστά έναν πελάτη.
- Κάθε πελάτης – κόμβος απαιτεί συγκεκριμένη ποσότητα προϊόντων για να εξυπηρετηθεί (demand).
- Τα οχήματα σε κάθε πρόβλημα έχουν συγκεκριμένη χωρητικότητα προϊόντων.
- Κάθε όχημα του οποίου η χωρητικότητα φθάνει στο ανώτατο όριο θα πρέπει να επιστρέψει στην αποθήκη για ανεφοδιασμό. Κάθε διαδρομή ξεκινά και καταλήγει στην αποθήκη.

- Στόχος του προβλήματος είναι η δημιουργία διαδρομής ελάχιστης διανυόμενης απόστασης. Από τη διαδρομή θα πρέπει να εξυπηρετούνται όλοι οι πελάτες καθώς και να μην παραβιάζονται οι περιορισμοί.




**Γράφημα**: Απεικόνιση γραφήματος διαδρομής δρομολόγησης οχημάτων

Είναι δυνατό να δοθούν πρότυπα που εκφράζουν πιο έντονα τα επιμέρους στοιχεία του προβλήματος και των οποίων ο αριθμός μεταβλητών είναι περιορισμένος. Έτσι, το περιορισμένης χωρητικότητα πρόβλημα δρομολόγησης οχημάτων μπορεί να περιγραφεί με όρους θεωρίας γράφων ως ακολούθως. Έστω ότι έχουμε ένας πλήρες γράφημα $G = (V,A)$, όπου $V = \{ 0,\ldots,n \}$ είναι το σύνολο των κόμβων με τον κόμβο 0 να αντιστοιχεί στην αποθήκη και οι υπόλοιποι κόμβοι αντιστοιχούν στους πελάτες. Σε πολλές περιπτώσεις, ανάλογα με την μορφοποίηση του προβλήματος, στην αποθήκη αντιστοιχίζεται και ο κόμβος $n+1$.Ένα μη αρνητικό κόστος $c_{ij}$ συσχετίζεται με κάθε τόξο $(i,j)$ και αντιπροσωπεύει το κόστος ταξιδιού από τον κόμβο $i$ στον κόμβο $j$. Όταν τα τόξα είναι μη προσανατολισμένα τότε το πρόβλημα ονομάζεται το συμμετρικό περιορισμένης χωρητικότητας δρομολόγησης οχημάτων. Σε πολλές περιπτώσεις οι τιμές του πίνακα κόστους ικανοποιούν την τριγωνική ανισότητα

$$c_{ik} + c_{kj} \geq c_{ij} \; \forall \, i,j,k \in V \qquad \textbf{(2.3.1)}$$

Η βασική προτυποποίηση για το απλό πρόβλημα δρομολόγησης οχημάτων έχει παρουσιαστεί από τους Fisher & Jaikumar. Αρχικά, υποθέτουμε ότι όλες οι εφικτές διαδρομές με αρχή και σημείο αφετηρίας έχουν απαριθμηθεί και ότι ο

αριθμός είναι Μ. Δημιουργείται ένας πίνακας Α = [$\alpha_{i,j}$] διαστάσεων Μ x Ν με Ν τον αριθμό των πελατών.

$$x_{ijk} = \begin{cases} 1\text{ , εάν το όχημα k επισκέπτεται τον πελάτη j αμέσως μετά τον πελάτη} \\ 0\text{, αλλιώς} \end{cases} \quad \textbf{(2.3.2)}$$

$$y_{jk} = \begin{cases} 1\text{ ,εάν ο πελάτης i επισκέπτεται από το όχημα k} \\ 0\text{, αλλιώς} \end{cases} \quad \textbf{(2.3.3)}$$

$$\min \sum_{i,j} c_{ij} \sum_k x_{ijk} \quad \textbf{(2.3.4)}$$

$$\sum_k y_{ik} = \begin{cases} 1, & i = 2, \dots, n, \\ m, & i = 1 \end{cases} \quad \textbf{(2.3.5)}$$

$$\sum_i q_i y_{ij} \leq Q_k \ , \quad k = 1, \dots, m \quad \textbf{(2.3.6)}$$

$$\sum_j x_{ijk} = \sum_j x_{jik} = y_{ik} \quad i = 1, \dots, n \ \& k \ = 1, \dots, m \quad \textbf{(2.3.7)}$$

$$\sum_{i,j \in S} x_{ijk} \leq |S| - 1 \quad για\ όλα\ τα\ S \cup \{2, \dots, n\} \ \& \ k = 1, \dots, m \quad \textbf{(2.3.8)}$$

$$y_{ik} \in \{0,1\} \quad \textbf{(2.3.9)}$$

$$x_{ijk} \in \{0,1\} \quad \textbf{(2.3.10)}$$

**Γράφημα**: Μαθηματική Μοντελοποίηση του Προβλήματος Δρομολόγησης Οχημάτων

Ο πρώτος περιορισμός στην παραπάνω μοντελοποίηση δείχνει ότι κάθε πελάτης εκχωρείται σε ένα μόνο όχημα, εκτός της αποθήκης που την επισκέπτονται όλα τα οχήματα, ο δεύτερος περιορισμός αφορά το άνω όριο χωρητικότητας των

οχημάτων, ο τρίτος περιορισμός δείχνει ότι ένα όχημα που επισκέπτεται ένα πελάτη φεύγει από τον πελάτη. Στις περισσότερες περιπτώσεις προβλημάτων δρομολόγησης οχημάτων οι κόμβοι συσχετίζονται μεταξύ τους με σημεία στο επίπεδο δηλαδή τα δεδομένα είναι συντεταγμένες, και το κόστος $c_{ij}$ για κάθε τόξο *(i,j)* υπολογίζεται από την ευκλείδεια απόσταση ανάμεσα στα δύο σημεία που αντιστοιχούν στους κόμβους i και j. Σε αυτή την περίπτωση ο πίνακας κόστους είναι συμμετρικός και ικανοποιεί την τριγωνική ανισότητα, το πρόβλημα δρομολόγησης ονομάζεται ευκλείδειο περιορισμένης χωρητικότητας πρόβλημα δρομολόγησης οχημάτων. Με βάση και την μοντελοποίηση του προβλήματος στόχος είναι η εύρεση Κ απλών κύκλων, με κάθε έναν να αντιστοιχεί σε ένα όχημα, με το ελάχιστο δυνατό κόστος και καθορίζεται σαν το άθροισμα όλων των κοστών των τόξων.

## 2.4 Βασικά Προβλήματα Δρομολόγησης Οχημάτων

Κατά καιρούς έχουν αναπτυχθεί πολλές παραλλαγές για το πρόβλημα δρομολόγησης οχημάτων. Οι διάφορες παραλλαγές του προβλήματος έχουν να κάνουν με τις αυξανόμενες απαιτήσεις στο τομέα της εφοδιαστικής αλυσίδας. Κάποιες από τις πιο γνωστές παραλλαγές αποτελούν: Το πρόβλημα δρομολόγησης οχημάτων με περιορισμό χωρητικότητας και χρόνους εξυπηρέτησης, το πρόβλημα δρομολόγησης με πολλαπλές επιστροφές στην αποθήκη, το ανοιχτό πρόβλημα δρομολόγησης οχημάτων, το πρόβλημα δρομολόγησης για την εξυπηρέτηση πελατών μέσα σε δεδομένα χρονικά περιθώρια. Παρακάτω παρουσιάζονται μερικά από τα βασικά προβλήματα δρομολόγησης.

### 2.4.1 Το Πρόβλημα Δρομολόγησης Οχημάτων με Περιορισμό Χωρητικότητας και Χρόνους εξυπηρέτησης (Capacitated Vehicle Routing Problem)

Το πρόβλημα το οποίο πραγματεύεται και η παρούσα διπλωματική εργασία είναι το πρόβλημα δρομολόγησης οχημάτων με περιορισμό χωρητικότητας και χρόνους εξυπηρέτησης ( Capacitated Vehicle Routing Problem). Πρόκειται για ένα πιο ολοκληρωμένο παράδειγμα που προσθέτει στο πρόβλημα ένα ακόμα περιορισμό, τον περιορισμό του χρόνου εξυπηρέτησης για τους πελάτες και του μέγιστου χρόνου διαδρομής για τα οχήματα. Η διαφορά του με το απλό πρόβλημα είναι η ότι εκτός της ανώτατης χωρητικότητας που μπορεί να εξυπηρετήσει ένα όχημα σε κάθε

διαδρομή η συνολική του παραμονή στην διαδρομή δεν θα πρέπει να ξεπερνά ένα ανώτατο όριο (Max Route Duration). Ακόμα, μία παράμετρος που εισέρχεται στο πρόβλημα είναι ο χρόνος εξυπηρέτησης. Πιο συγκεκριμένα, εκτός από την διανυόμενη απόσταση από τον κόμβο i στον κόμβο j, τη ζήτηση για κάθε πελάτη προστίθεται και ο χρόνος που απαιτείται για την εξυπηρέτηση κάθε πελάτη. Η μοντελοποίηση του προβλήματος είναι ίδια με τη μοντελοποίηση του απλού προβλήματος με την προσθήκη περιορισμού για την μέγιστη παραμονή του οχήματος σε μία διαδρομή καθώς και τον χρόνο εξυπηρέτησης. Παρουσιάζεται η προσθήκη περιορισμού σύμφωνα με τη μοντελοποίηση του προβλήματος (CPVRP) από τον Golden.

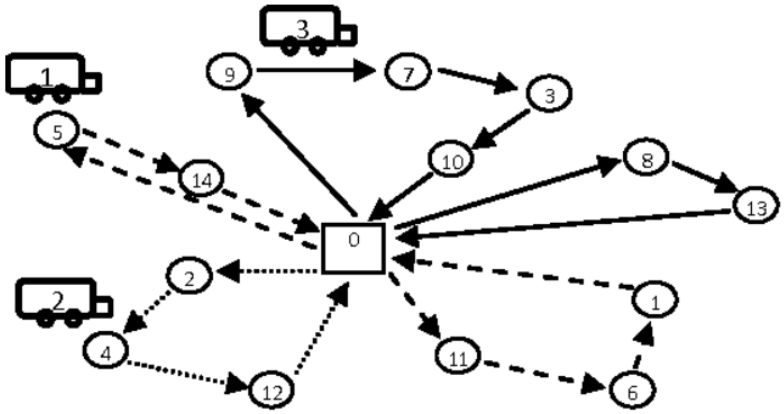
$$\sum_{i=1}^{n} t_i^u \sum_{j=1}^{n} x_{ij}^u + \sum_{i=1}^{n} \sum_{i=1}^{n} t_{ij}^u x_{ij}^u \leq T_u, \quad u = 1, \dots, NV \quad \textbf{(2.4.1.1)}$$

Με $T_u$ το μέγιστο χρόνο παραμονής του οχήματος στη διαδρομή, $t_{ij}^u$ το χρόνο μετάβασης από τον κόμβο i στον κόμβο j και $t_i^u$ το χρόνο που απαιτείται για την εξυπηρέτηση ενός πελάτη.

### 2.4.2 Το Πρόβλημα Δρομολόγησης Οχημάτων με Πολλαπλές Επιστροφές στην Αποθήκη (Multitrip Vehicle Routing Problem)

Στα περισσότερα προβλήματα δρομολόγησης οχημάτων , στα οχήματα επιτρέπεται να πραγματοποιήσουν μόνο μία απλή διαδρομή. Υπάρχουν όμως πολλές περιπτώσεις όπου τα οχήματα μπορούν να χρησιμοποιηθούν παραπάνω από μία φορά. Στο πρόβλημα δρομολόγησης με πολλαπλές επιστροφές υπάρχει μία κεντρική αποθήκη και ένας αριθμός οχημάτων τα οποία μπορούν να πραγματοποιούν συγκεκριμένο αριθμό διαδρομών. Για τη μοντελοποίηση του προβλήματος, δίνεται αρχικά ένα πλήρες μη προσανατολισμένο γράφημα $G = (V', E)$ όπου $V'$ είναι το σύνολο των κόμβων και $E$ είναι το σύνολο των τόξων. Τα χαρακτηριστικά του προβλήματος πολλαπλών επιστροφών δεν διαφέρουν από τα δύο προηγούμενα προβλήματα που παρουσιάστηκαν. Τα χαρακτηριστικά του θα πρέπει να εντάσσονται σε ένα πρόγραμμα ενός οχήματος ώστε να είναι ένα υποσύνολο από διαδρομές όπου η συνολική διάρκειά τους είναι μικρότερη ή ίση με το μέγιστο χρόνο που μπορεί να παραμείνει ένα όχημα στη διαδρομή. Πιο συγκεκριμένα, ένα όχημα μπορεί να πραγματοποιήσει πάνω από μία διαδρομές που δεν παραβιάζουν τη χωρητικότητα του οχήματος κατά τη διάρκεια μίας ημέρας αλλά η χρονική διάρκεια του συνόλου των διαδρομών δεν θα μπορεί να ξεπεράσει τη μέγιστη χρονική διάρκεια που μπορεί ένα όχημα να παραμείνει στη διαδρομή. Οι πελάτες

εξακολουθούν να εξυπηρετούνται αποκλειστικά από συγκεκριμένες αποθήκες, με τη διαφορά όμως ότι ο στόλος των οχημάτων που απασχολεί κάθε αποθήκη δεν είναι σταθερός, καθ' όλες τις περιόδους εξυπηρέτησης, αλλά μεταβάλλεται προκειμένου να καλύψει την ζήτηση των πελατών της.




**Γράφημα**: Το πρόβλημα δρομολόγησης οχημάτων με πολλαπλές επιστροφές στην αποθήκη.

Η μοντελοποίηση του προβλήματος προτάθηκε από τους Mingozzi, Roberti και Toth. Έστω $R$ ο δείκτης του συνόλου όλων των εφικτών διαδρομών στο γράφημα G και $R_i$ υποσύνολο του $R$, ο δείκτης του υποσυνόλου των διαδρομών που επισκέπτονται τον πελάτη $i \in V$. Σε κάθε διαδρομή $l \in R$ αντιστοιχίζεται ένα κόστος $d_l$ και μία χρονική διάρκεια $T_l$. Με R και με E(R) συμβολίζονται το σύνολο των πελατών και τα τόξα που πέρασαν κατά τη διάρκεια της διαδρομής $l$.

$$z = min \sum_{l \in R} d_l \sum_{j \in M} x_l^j \qquad \textbf{(2.4.3.1)}$$

$$\sum_{l \in R} \sum_{J \in R} x_l^j = 1\,, \quad \forall\, i \in V \qquad \textbf{(2.4.3.2)}$$

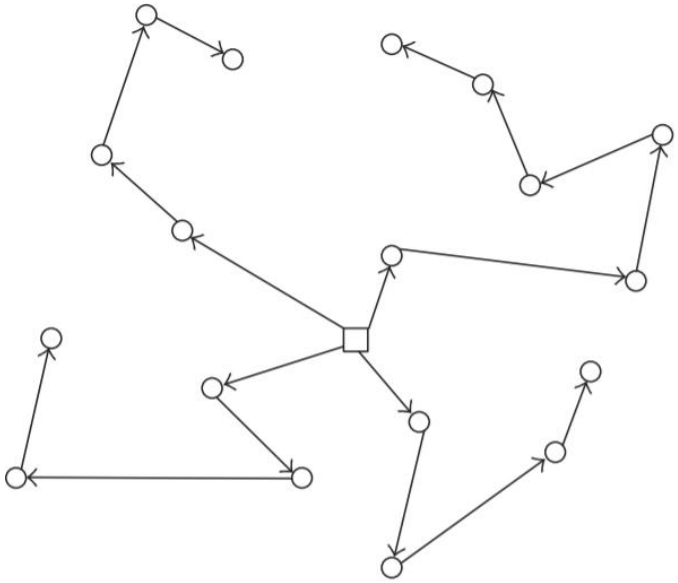
$$\sum_{l \in R} \tau_l x_l^j \leq \quad T, \quad \forall\, j \in M \qquad \textbf{(2.4.3.3)}$$

$$x_l^j \in \{\, 0\,,1\,\}, \quad \forall\, j \in M, \forall\, l \in M \qquad \textbf{(2.4.3.4)}$$

Ο πρώτος περιορισμός εξασφαλίζει ότι ο κάθε πελάτης επισκέπτεται ακριβώς μία φορά ο δεύτερος καθορίζει την εφικτότητα ενός προγράμματος σχετικά με τη χρονική διάρκεια μιας διαδρομής.

### 2.4.3 Το Ανοιχτό Πρόβλημα Δρομολόγησης Οχημάτων (Open Vehicle Routing Problem)

Το ανοιχτό πρόβλημα δρομολόγησης οχημάτων (Open Vehicle Routing Problem) είναι μία παραλλαγή του βασικού προβλήματος με τη διαφορά ότι τα οχήματα δεν επιστρέφουν στην αποθήκη. Η βασική εφαρμογή της παραλλαγής συναντάται σε περιπτώσεις όπου μία εταιρία δεν διαθέτει στόλο ώστε να εξυπηρετήσει τις μεταφορές οπότε αναγκάζεται να χρησιμοποιήσει άλλες εταιρίες διανομών ή να νοικιάσει στόλο. Στο παρακάτω σχήμα δίνεται ένα ανοιχτό πρόβλημα δρομολόγησης οχημάτων. Σε κάθε διαδρομή το κάθε όχημα μόλις ολοκληρώσει τον κύκλο εξυπηρέτησης των πελατών δεν επιστρέφει στην αποθήκη αλλά επιστρέφει στην εταιρία από την οποία ενοικιάσθηκε ή συνεχίζει να απασχολείται με άλλες εργασίες.



**Γράφημα**: Το Ανοιχτό Πρόβλημα Δρομολόγησης Οχημάτων.

Έστω ότι το $G = (V,E)$ είναι ένα γράφημα όπου $V = \{J_0, J_1, J_2, \ldots J_n\}$ είναι το σύνολο των κόμβων, με τον κόμβο $J_0$ να αντιστοιχεί στην αποθήκη και τους υπόλοιπους στους πελάτες. Κάθε πελάτης θα πρέπει να εξυπηρετηθεί από ακριβώς ένα από τα k οχήματα και το συνολικό μέγεθος των διανομών για τους πελάτες που έχουν ανατεθεί σε ένα όχημα δεν μπορεί να ξεπερνά την χωρητικότητα του οχήματος ($Q_k$). Κάθε πελάτης έχει μια ορισμένη ζήτηση καθώς και χρόνο εξυπηρέτησης με συμβολισμούς $q_{ij}$ και $st_{ij}$ αντίστοιχα. Ο χρόνος που χρειάζεται για να μετακινηθεί ένα όχημα από ένα κόμβο i στον κόμβο j συμβολίζεται με $cost_{ij}$, ενώ το όχημα δεν θα

πρέπει να ξεπερνά το άνω όριο παραμονής στην διαδρομή. Στόχος του προβλήματος είναι η δημιουργία διαδρομής ελάχιστης απόστασης με προϋπόθεση ότι όταν ολοκληρώσει τον κύκλο ένα όχημα δεν επιστρέφει στην αποθήκη.

$$min \sum_{i=1}^{n} \sum_{j=1}^{n} \sum_{k=1}^{K} c_{ij} x_{ij}^{k} \quad \textbf{(2.4.3.1)}$$

$$\sum_{i=1}^{n} \sum_{k=1}^{K} x_{ij}^{k} = 1\ ,\ j = 2, \dots, n1 \quad \textbf{(2.4.3.2)}$$

$$\sum_{j=2}^{n} \sum_{k=1}^{K} x_{ij}^{k} = 1,\ i = 2, \dots, n \quad \textbf{(2.4.3.3)}$$

$$\sum_{i=1}^{n} x_{ii_1} - \sum_{j=2}^{n} x_{i1j} = 0,\ k = 1, \dots, K\ , i_1 = 1, \dots, n \quad \textbf{(2.4.3.4)}$$

$$\sum_{j=2}^{n} q_i \sum_{i=1}^{n} x_{ij}^{k} \leq Q_k,\ k = 1, \dots, K \quad \textbf{(2.4.3.5)}$$

$$\sum_{j=2}^{n} t_i^k \sum_{i=1}^{n} x_{ij}^{k} + \sum_{j=2}^{n} \sum_{i=1}^{n} t_{ij}^{k} x_{ij}^{k} \leq T_{mk},\ k = 1, \dots, K \quad \textbf{(2.4.3.6)}$$

$$\sum_{j=2}^{n} x_{1j}^{k} \leq 1,\ k = 1, \dots, K \quad \textbf{(2.4.3.7)}$$

$$\sum_{i=2}^{n} x_{i1}^{k} = 0,\ k = 1, \dots K \quad \textbf{(2.4.3.8)}$$

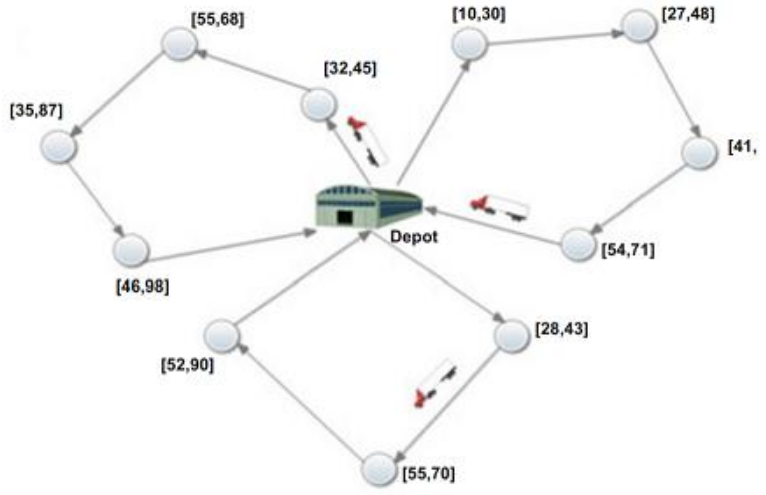
$$X \in S \quad \textbf{(2.4.3.9)}$$

$$x_{ij}^{k} = 0 \text{ ή } 1 \quad \textbf{(2.4.3.10)}$$

Η αντικειμενική συνάρτηση καθορίζει την ελαχιστοποίηση της διανυόμενης απόστασης. Οι περιορισμοί **2.4.3.2-3** καθορίζουν ότι ο κάθε πελάτης θα εξυπηρετηθεί από ένα όχημα. Ο περιορισμός **2.4.3.5** αποτελεί την μέγιστη επιτρεπόμενη χωρητικότητα κάθε οχήματος και οι εξισώσεις 2.4.3.6 αντιπροσωπεύουν τους χρονικούς περιορισμούς κάθε διαδρομής. Οι εξισώσεις **2.4.3.7-8** εξασφαλίζουν ότι η διαθεσιμότητα των οχημάτων δεν παραβιάζεται.

### 2.4.4 Το Πρόβλημα Δρομολόγησης για την Εξυπηρέτηση Πελατών μέσα σε Δεδομένα Χρονικά Περιθώρια ( Vehicle Routing Problem with Time Windows)

Το πρόβλημα δρομολόγησης οχημάτων με εξυπηρέτηση σε δεδομένα χρονικά περιθώρια αποτελεί επέκταση του βασικού προβλήματος δρομολόγησης. Πιο συγκεκριμένα, τα χαρακτηριστικά του προβλήματα ακολουθούν την δομή των κλασσικών προβλημάτων, το επιπλέον χαρακτηριστικό είναι ότι κάθε πελάτης έχει ένα συγκεκριμένο χρονικό περιθώριο για το οποίο θα είναι διαθέσιμος για εξυπηρέτηση. Έτσι, μία λύση είναι εφικτή όταν το όχημα έχει επισκεφθεί όλους του πελάτες, δεν παραβιάζεται σε καμία διαδρομή ο περιορισμός χωρητικότητας και η εξυπηρέτηση έχει γίνει εντός του χρονικού πλαισίου διαθεσιμότητας του πελάτη. Αναλυτικότερα, η εξυπηρέτηση για κάθε πελάτη πρέπει να ξεκινήσει εντός δεδομένων χρονικών πλαισίων που μπορεί ο πελάτης να εξυπηρετηθεί. Αν κάποιο όχημα φτάσει νωρίτερα από αυτό το περιθώριο σε κάποιον πελάτη στις περισσότερες περιπτώσεις είναι ανεκτή η αναμονή έως ότου το περιθώριο χρόνου ενεργοποιηθεί. Υπάρχουν δύο ειδών χρονικά παράθυρα, τα χαλαρά κατά τα οποία μία παράδοση μπορεί να συμβεί και εκτός του χρονικού πλαισίου που έχει τεθεί και τα σκληρά χρονικά παράθυρα όπου δεν επιτρέπεται η εξυπηρέτηση μετά το πέρας του χρονικού περιθωρίου.




**Γράφημα**: Το Πρόβλημα Δρομολόγησης για την Εξυπηρέτηση Πελατών μέσα σε Δεδομένα Χρονικά Περιθώρια.

Παρακάτω παρουσιάζεται η μοντελοποίηση του προβλήματος, όπου η αποθήκη συμβολίζεται με 0 και *n+1* και όλες οι διαδρομές είναι εφικτές αν ξεκινούν

και τελειώνουν στην αποθήκη. Ένα χρονικό περιθώριο συμβολίζεται με $\{\alpha_0, \beta_0\} = \{\alpha_{n+1}, \beta_{n+1}\} = \{E, L\}$, όπου τα E και L, να αναπαριστούν την ελάχιστη πιθανή αναχώρηση από την αποθήκη και η αργότερη δυνατή άφιξη.

$$min \ \sum_{i,j} c_{ij} \ \sum_{i,j} x_{ijk} \quad \textbf{(2.5.3.1)}$$

$$\sum_{k \in K} \sum_{j \in V} x_{ijk} \ = \ 1 \quad \textbf{(2.5.3.2)}$$

$$\sum_{i \in V-\{0\}} x_{0jk} \ = \ 1, \ \forall \, j \in N, k \in \ K \quad \textbf{(2.5.3.3)}$$

$$\sum_{i \in V-\{j\}} x_{ijk} \ - \sum_{i \in V-\{j\}} x_{ijk} \ = \ 0 \, , \forall \, j \in N, k \in \ K \quad \textbf{(2.5.3.4)}$$

$$\sum_{i \in V-\{n+1\}} x_{in+1k} \ = \ 1, \ k \in \ K \quad \textbf{(2.5.3.5)}$$

$$x_{ijk}(w_{ik} + s_i \ + \ t_{ij} \ - \ w_{jk}) \ \leq \ 0 \ , k \in \ K(i,j) \ \in \ A \quad \textbf{(2.5.3.6)}$$

$$\alpha_i \sum_{J \in V} x_{ijk} \leq \ w_{ik} \ \leq \ \beta_i \sum_{j \in V} x_{ijk} \ , \forall \, j \in N, k \in \ K \quad \textbf{(2.5.3.7)}$$

$$\mathrm{E} \ \leq \ w_{ik} \ \leq \ L \quad \textbf{(2.5.3.8)}$$

$$\sum_{i \in N} d_i \ \sum_{j \in V} x_{ijk} \ \leq \ C \ , \forall \, k \in \ K \quad \textbf{(2.5.3.9)}$$

$$x_{ijk} \ \geq \ 0 \ \forall \, k \in \ K, (i,j) \ \in \ A \quad \textbf{(2.5.3.10)}$$

$$x_{ijk} \ \in \ \{0,1\} \quad \textbf{(2.5.3.11)}$$

Η αντικειμενική συνάρτηση εκφράζει το συνολικό κόστος. Οι περιορισμοί **2.5.3.2** εκφράζουν την εξυπηρέτηση του κάθε πελάτη από ένα όχημα. Οι περιορισμοί **2.5.3.3 – 2.5.3.5** χαρακτηρίζουν την διαδρομή που θα ακολουθήσει κάθε όχημα. Οι περιορισμοί **2.5.3.6,8,10** εγγυώνται την εφικτότατα των διαδρομών με βάση τους χρονικούς περιορισμούς και τους περιορισμούς χωρητικότητας. Τέλος, ο περιορισμός **2.5.3.7** εκφράζει τους χρόνους αναχώρησης και άφιξης από και προς την αποθήκη.

# 3. Αλγόριθμοι Εμπνευσμένοι από τη Φύση

## 3.1 Εισαγωγή

Οι αλγόριθμοι που παρουσιάζονται εντάσσονται στην κατηγορία των αλγόριθμων εμπνευσμένων από τη φύση και γενικότερα στην κατηγορία των εξελικτικών αλγορίθμων. Η δομή τους βασίζεται στις φυσικές διεργασίες οργανισμών είτε αυτό αφορά την εξεύρεση τροφής, τη διαδικασία εύρεσης φωλιάς, τους μηχανισμούς πλοήγησης. Η λειτουργία των αλγορίθμων βασίζεται στην νοημοσύνη σμήνους. Η συγκεκριμένη λειτουργία ξεκινά με την ύπαρξη ενός αρχικού πληθυσμού και την διενέργεια συγκεκριμένων κινήσεων εντός του πεδίου λύσεων για την εξεύρεση βέλτιστων λύσεων σε προβλήματα. Οι κινήσεις αυτές προσομοιάζουν τη λειτουργία ενός σμήνους.

## 3.2 Εξελικτικοί Αλγόριθμοι

Οι εξελικτικοί αλγόριθμοι αποτελούν ένα συνεχώς αυξανόμενο πεδίο έρευνες για τους τομείς της βελτιστοποίησης. Η κατηγορία των αλγορίθμων αυτών έχει ως βάση έναν αρχικό πληθυσμό από εφικτές λύσεις του προβλήματος που μελετάται. Το επόμενο στοιχείο των αλγορίθμων αποτελεί η επαναληπτική εφαρμογή διαδικασιών για ένα συγκεκριμένο αριθμό γενιών. Περνώντας από γενιά σε γενιά, οι διαδικασίες που εφαρμόζονται έχουν ως στόχο την αναμόρφωση των αρχικών πληθυσμών ώστε να παραχθούν καλύτερες λύσεις από τις αρχικές.

Μία βασική περίπτωση εξελικτικών αλγορίθμων είναι οι γενετικοί. Όντας η πιο γνωστή μορφή εξελικτικών αλγορίθμων είναι χρήσιμο να εξετασθούν οι βασικές τους λειτουργίες μιας και αποτυπώνουν σε μεγάλο βαθμό τις διεργασίες που διέπουν και τους υπόλοιπους αλγορίθμους που εντάσσονται στην κατηγορία. Οι διεργασίες, αυτές, περιλαμβάνουν τον εξελικτικό προγραμματισμό, την στρατηγική βελτίωσης των ασθενέστερων μονάδων ενός πληθυσμού, την υλοποίηση της φυσικής επιλογής.

Οι εξελικτικοί αλγόριθμοι είναι ανίχνευσης – αναζήτησης με βάση τις φυσικές διεργασίες. Στα προβλήματα βελτιστοποίησης οι αλγόριθμοι στοχεύουν στην διαχρονική βελτίωση των πιθανών λύσεων με υπολογιστικά μοντέλα που προσομοιάζουν τις φυσικές διαδικασίες.

## 3.3 Αλγόριθμος Βελτιστοποίησης Σμήνους Σωματιδίων

Ο αλγόριθμος σμήνους σωματιδίων παρουσιάστηκε από τους Kennedy & Eberhart (1995). Αποτελεί μια παράλληλη αναζήτηση στο πεδίο λύσεων. Είναι βασισμένος στην συμπεριφορά ζωικών ειδών και προσπαθεί να μιμηθεί τη συμπεριφορά τους στην αναζήτηση τροφής, στην εύρεση καλύτερων φωλιών και άλλων βιολογικών διεργασιών. Ο αλγόριθμος αποτελείται από έναν αρχικό πληθυσμό υποψήφιων λύσεων. Κάθε λύση αποτελεί ένα σωματίδιο, κάθε σωματίδιο έχει δύο χαρακτηριστικά μεγέθη, τα οποία είναι η θέση και η ταχύτητα. Η θέση του σωματιδίου αντικατοπτρίζει την τοποθεσία του κάθε σωματιδίου στο πεδίο λύσεων, ενώ η ταχύτητα του αντικατοπτρίζει την κατεύθυνση κάθε σωματιδίου για το που θα πρέπει να κινηθεί. Τα βήματα του αλγορίθμου περιλαμβάνουν την επαναληπτική διαδικασία υπολογισμού ταχύτητας και θέσης για κάθε σωματίδιο του πληθυσμού με βάση χαρακτηριστικά που έχει από την προηγούμενη επανάληψη.

Τα χαρακτηριστικά που επηρεάζουν ταχύτητα και θέση του σωματιδίου αποτελούν η θέση του καλύτερου σωματιδίου, με βάση την αντικειμενική συνάρτηση, καθώς και η καλύτερη θέση που είχε το σωματιδίου στις προηγούμενες επαναλήψεις. Η καλύτερη θέση που είχε ένα σωματίδιο στην διάρκεια της επαναληπτικής διαδικασίας καλείται *pbest* (personal best), ενώ η τιμή του καλύτερου σωματιδίου καλείται *gbest* (global best). Ο λόγος που οι θέσεις και οι ταχύτητες του σμήνους επηρεάζονται από τα δύο παραπάνω χαρακτηριστικά είναι ότι στη φύση η τάση των μελών του σμήνους είναι να ακολουθούν το καλύτερο μέλος καθώς και να καταγράφουν τις καλύτερες προσωπικές τους θέσεις. Κάθε μέλος του πληθυσμού έχει τρείς πιθανές κινήσεις στο χώρο λύσεων. Η πρώτη είναι να ακολουθήσει μια δικιά του τυχαία πορεία, η δεύτερη είναι να αναζητά θέσεις μέσα στο χώρο λύσεων και να επιστρέφει στην καλύτερη θέση που είχε αν η επόμενη θέση που βρήκε είναι χειρότερη και τέλος να ακολουθεί γειτονικά σωματίδια. Κατά αυτό τον τρόπο, συνδυάζεται στην συμπεριφορά των σωματιδίων μια γνωστική όσο και μία κοινωνική φύση των μελών του πληθυσμού. Οι Kennedy & Eberhart πρότειναν τις σχέσεις (1) και (2) για την ενημέρωση των ταχυτήτων και των θέσεων των σωματιδίων. Στις σχέσεις που παρουσιάζονται ως $x_p(t)$ και $u_p(t)$ αποτελούν η θέση και η ταχύτητα του σωματιδίου για την επανάληψη $t$ αντίστοιχα. Οι συντελεστές *c1* και *c2* είναι δυο συντελεστές που προσομοιάζουν την φύση των σωματιδίων με τον τρόπο που προαναφέρθηκε. Πιο συγκεκριμένα, ο συντελεστής *c1* λειτουργεί ως γνωστικός

όρος και χρησιμοποιείται για να επαναφέρει το σωματίδιο σε παλαιότερες καλύτερες θέσεις που ήταν περισσότερο αποδοτικές, ενώ ο συντελεστής *c2* λειτουργεί ως κοινωνικός όρος και είναι αυτός που κατευθύνει το κάθε σωματίδιο προς το βέλτιστο.

$$u_p(t) = u_p(t\text{-}1) + c1 \cdot rand1 \cdot (pbest_p(t\text{-}1) - x_p(t\text{-}1)) + c2 \cdot rand2 \cdot (gbest_p(t\text{-}1) - x_p(t\text{-}1)) \quad \textbf{(3.1.1)}$$

$$x_p(t) = u_p(t\text{-}1) + u_p(t) \quad \textbf{(3.1.2)}$$

Το 1998 οι Shi & Eberhart μετέβαλλαν την διαδικασία υπολογισμού της ταχύτητας των σωματιδίων εισάγοντας έναν παράγοντα βάρους αδράνειας. Στόχος τους ήταν το σωματίδιο να κινηθεί με βάση την προηγούμενη του κίνηση. Με την εισαγωγή αυτού του συντελεστή αποφεύγονταν κινήσεις των σωματιδίων που ήταν εντελώς εκτός του τοπικού χώρου λύσεων. Ο συντελεστής αδράνειας μειώνεται σε σχέση με τις επαναλήψεις.

$$u_p(t) = w \cdot u_p(t\text{-}1) + c1 \cdot rand1 \cdot (pbest_p(t\text{-}1) - x_p(t\text{-}1)) + c2 \cdot rand2 \cdot (gbest_p(t\text{-}1) - x_p(t\text{-}1)) \quad \textbf{(3.1.3)}$$

$$w = w_{max} - \frac{w_{max} - w_{min}}{t_{max}} \cdot t$$

Το 1999 ο Clerc πρότεινε την ένταξη στον υπολογισμό των ταχυτήτων ενός συντελεστή περιορισμού. Σκοπός του ήταν να διασφαλιστεί η σύγκλιση του αλγορίθμου και η αποφυγή απομάκρυνσης των σωματιδίων από το ολικό βέλτιστο. Οι ταχύτητες των σωματιδίων υπολογίζονται με βάση την παρακάτω σχέση.

$$u_p(t) = K \cdot (u_p(t\text{-}1) + c1 \cdot rand1 \cdot (pbest_p(t\text{-}1) - x_p(t\text{-}1)) + c2 \cdot rand2 \cdot (gbest_p(t\text{-}1) - x_p(t\text{-}1))) \quad \textbf{(3.1.4)}$$

$$K = \frac{2}{2 - c - \sqrt{c^2 - 4 \cdot c}} \quad \text{με} \quad c1 + c2 > 4$$

Η ενημέρωση των καλύτερων θέσεων των σωματιδίων καθώς και του καλύτερου σωματιδίου για προβλήματα ελαχιστοποίησης γίνεται ως εξής:

$$\text{pbest}_{ij} = \begin{cases} X_{ij}^{\,t+1}, \text{ εάν } f(X_{ij}^{\,t+1}) < f(X_{ij}^{\,t}) \\ \text{pbest}_{ij}, \text{ αλλιώς} \end{cases} \quad \textbf{(3.1.5)}$$

Η βέλτιστη θέση όλου του σμήνους τη χρονική στιγμή t υπολογίζεται από την εξίσωση:

$$\text{gbest}_j \{\text{pbest}_{1j}, \text{pbest}_{2j}, \text{pbestNj} \,|f(\text{gbest}_j)|\} = \min\{f(\text{pbest}_{1j}), f(\text{pbest}_{2j}) \ldots, f(\text{pbest}_{Nj})\} \quad \textbf{(3.1.6)}$$

**Algorithm:** Particle Swarm Optimization

1. *Initialize Population of Particles*
2. ***Do***
3. ***For*** *each Particle p in position* $x_p$ *do*
4. ***If*** ($x_p$ *is better than* $pbest_p$) *then*
5. $Pbest_p \leftarrow x_p$
6. ***end_if***
7. ***end for***
8. *Define* $gbest_p$ *as the best particle found in population*
9. ***For*** *each particle p do*
10. *Update* $u_p$, $x_p$
11. $U_p \leftarrow$ *Compute Velocity* ($x_p$, $pbest_p$, $gbest_p$)
12. $X_p \leftarrow$ *Update Position* ($x_p$, $U_p$)
13. ***End for***
14. ***While*** *(a stopping condition is not satisfied)*

**Γράφημα**: Λειτουργία αλγορίθμου Βελτιστοποίησης Σμήνους Σωματιδίων

Οι μεταβλητές επιτάχυνσης $c1$ και $c2$, όπως παρουσιάζονται στις παραπάνω σχέσεις, καθορίζουν το πόσο μακριά μπορεί να κινηθεί ένα σωματίδιο κατά τη διάρκεια μιας επανάληψης. Χαμηλές τιμές επιτρέπουν στα σωματίδια να αποκλίνουν από τη στοχευμένη περιοχή πριν βρεθούν κοντά σε τοπικό ελάχιστο, ενώ πολύ υψηλές τιμές έχουν ως αποτέλεσμα τα σωματίδια να συγκλίνουν προς τις στοχευμένες περιοχές. Τυπικά και οι δύο αυτές μεταβλητές παίρνουν την τιμή 2.0, παρόλα αυτά για διάφορα προβλήματα βελτιστοποίησης προτιμώνται τιμές μικρότερες. Έχουν παρουσιαστεί διάφοροι τρόποι αρχικοποίησης των συντελεστών επιτάχυνσης. Σε κάποια προβλήματα οι συντελεστές εξ' αρχής παίρνουν την τιμή 2.0, ενώ έχουν παρουσιαστεί τρόποι υπολογισμού με μεταβολή των τιμών των συντελεστών κατά τη διάρκεια των επαναλήψεων.

$$c_1 = c_{1,min} + \frac{c_{1,max} - c_{1,min}}{t_{max}} \cdot t$$

$$c_2 = c_{2,min} + \frac{c_{2,max} - c_{2,min}}{t_{max}} \cdot t$$

Ανάλογα με τις τιμές των συντελεστών μπορεί να γίνει αντιληπτή η κίνηση των σωματιδίων και ποια από τις τρείς κατευθύνσεις λαμβάνει το σωματίδιο. Πιο συγκεκριμένα, αν $c1 = c2 = 0$ τότε το σωματίδιο ακολουθεί αυτόνομη πορεία, εάν $c1 > 0$ και $c2 = 0$ τότε το σωματίδιο κινείται στην κατεύθυνση των προηγούμενων

κινήσεων του δίνοντας έμφαση στην γνωστική φύση του, αντίθετα αν *c1 = 0* και *c2 > 0* τότε το σωματίδιο δίνει έμφαση στην κοινωνική του συμπεριφορά λαμβάνοντας υπόψιν το βέλτιστο σωματίδιο του σμήνους.

## 3.4 Αλγόριθμος Βελτιστοποίησης Γκρίζου Λύκου

Ο αλγόριθμος βελτιστοποίησης γκρίζου λύκου παρουσιάστηκε από τον Seyedali Mirjalili (2013). Ο αλγόριθμος είναι εμπνευσμένος από τον τρόπο με τον οποίο μία αγέλη λύκων προσεγγίζει ένα θήραμα. Μια αγέλη λύκων συνήθως αποτελείται από 5-12 μέλη. Η δομή της αγέλης είναι αυστηρά καθορισμένη, βασισμένη στην δυναμική κάθε μέλους.

Κατά αυτόν τον τρόπο, ο αρχηγός της αγέλης καλείται alpha wolf. Ο αρχηγός αποτελεί τον κυρίαρχο της αγέλης μιας και οι αποφάσεις του είναι δεσμευτικές για τα υπόλοιπα μέλη της. Στο δεύτερο επίπεδο ιεραρχίας υπάρχουν οι beta wolves. Πρόκειται για ομάδα λύκων που δρουν συνεπικουρικά με τον alpha. Ο ρόλος τους έχει να κάνει με την παροχή βοήθειας στον αρχηγό της αγέλης τόσο στην λήψη αποφάσεων όσο και στην μεταφορά μηνυμάτων μεταξύ των ανώτερων και κατώτερων επιπέδων της ιεραρχίας. Αποτελούν υποψήφιοι αρχηγοί της αγέλης καθώς σε περίπτωση θανάτου του αρχηγού κάποιος από την ομάδα παίρνει το ρόλο του. Ο beta wolf θα πρέπει να δείχνει σεβασμό προς τον αρχηγό της αγέλης, αλλά και να ηγείται των μελών των κατώτερων επιπέδων. Στο επόμενο επίπεδο ιεραρχίας συναντάται η ομάδα λύκων delta wolf. Πρόκειται για μία ενδιάμεση ομάδα που ρόλο έχει τη σύνδεση των ανώτερων και κατώτερων επιπέδων. Συνήθως, ο ρόλος τους είναι η επιτήρηση του χώρου κίνησης της αγέλης, η εξεύρεση θηραμάτων κ.α. Στην ομάδα συγκαταλέγονται και τα μεγαλύτερα σε ηλικία μέλη της ομάδας. Το τελικό επίπεδο ιεραρχίας είναι αυτό των omega wolves. Μπορεί η σημαντικότητά τους να μην γίνεται αρχικώς αντιληπτή παρόλα αυτά διαδραματίζουν βασικό ρόλο στην συνεκτικότητα της αγέλης. Έχει παρατηρηθεί ότι μάχες μεταξύ των omega επηρεάζουν συνολικά τη συνοχή της αγέλης δημιουργώντας προβλήματα στα ανώτερα μέλη της.

Μετά από την ιεραρχία που αποτελεί βασική δομή στην λειτουργία του αλγορίθμου η επόμενη φυσική διεργασία των λύκων, από την οποία είναι εμπνευσμένες οι σχέσεις υπολογισμών, είναι η διαδικασία εύρεσης θηράματος. Η διεργασία αποτελεί μια πολύπλοκη ροή ενεργειών και απαιτεί συνεργασία μεταξύ των μελών της αγέλης. Τα στάδιά της είναι τα εξής: Παρακολούθηση, κυνήγι και

προσέγγιση του θηράματος στο αρχικό στάδιο, περικύκλωση και παρενόχληση του θηράματος μέχρι αυτό να σταματήσει να κινείται στο δεύτερο στάδιο και επίθεση στο τελικό στάδιο. Η μοντελοποίηση του αλγορίθμου συνδυάζει τις δύο παραπάνω διεργασίες δηλαδή τόσο την ιεραρχία του πληθυσμού όσο και την διαδικασία εύρεσης και επίθεσης στο θήραμα. Αρχικά, ταξινομούνται τα μέλη του αρχικού πληθυσμού ανάλογα με την τιμή της αντικειμενικής συνάρτησης του καθενός. Κατά αυτόν τον τρόπο, το καλύτερο μέλος του πληθυσμού γίνεται ο alpha wolf, το δεύτερο καλύτερο beta wolf, το τρίτο καλύτερο delta wolf και τέλος τα υπόλοιπα μέλη omega wolves. Η επόμενη διεργασία αποτελεί η περικύκλωση του θηράματος. Η μαθηματική μοντελοποίηση της συγκεκριμένης κίνησης επιτυγχάνεται με την ένταξη συντελεστών, ώστε να προσδίδουν στη θέση του μέλους κυκλική κίνηση γύρω από την υποψήφια λύση. Παρακάτω παρουσιάζονται οι σχέσεις καθώς και ο τρόπος υπολογισμού των συντελεστών.

$$\vec{A} = 2 \cdot \vec{\alpha} \cdot \overrightarrow{r_1} - \vec{\alpha} \quad \textbf{(3.2.1)}$$

$$\vec{C} = 2 \cdot \overrightarrow{r_2} \quad \textbf{(3.2.2)}$$

$$\overrightarrow{D} = | \vec{C} \cdot \overrightarrow{X_P(t)} - \overrightarrow{X(t)} | \quad \textbf{(3.2.3)}$$

$$\overrightarrow{X(t+1)} = \overrightarrow{X_P(t)} - \vec{A} \cdot \vec{D} \quad \textbf{(3.2.4)}$$

Το διάνυσμα $\vec{\alpha}$ μειώνεται γραμμικά από το 2 στο 0 από την πρώτη έως την τελευταία επανάληψη και οι τιμές $\overrightarrow{r_1}$, $\overrightarrow{r_2}$ αποτελούν διανύσματα με τυχαίες τιμές από το 0 έως το 1. Οι παραπάνω σχέσεις, όπως προαναφέρθηκε, προσδίδουν την κυκλική κίνηση των μελών του πληθυσμού γύρω από τον χώρο της υποψήφιας λύσης. Στο επόμενο στάδιο του αλγορίθμου εντάσσεται η διαδικασία που περιλαμβάνει το κυνήγι του θηράματος. Στην μοντελοποίηση των κινήσεων λαμβάνεται υπόψιν ότι στο συγκεκριμένο στάδιο τα μέλη του πληθυσμού ακολουθούν τις αποφάσεις του alpha wolf. Με αυτόν τον τρόπο οι θέσεις των μελών ανανεώνονται σε σχέση με την θέση του αρχηγού, όπως επίσης και με τις προηγούμενες θέσεις των μελών των υψηλών βαθμίδων του πληθυσμού. Για κάθε μία από τις βαθμίδες ιεραρχίας οι παραπάνω σχέσεις λαμβάνουν τη μορφή:

$$\overrightarrow{D_\alpha} = \left| \overrightarrow{C_1} \cdot \overrightarrow{X_\alpha} - \vec{X} \right|, \overrightarrow{D_\beta} = | \overrightarrow{C_2} \cdot \overrightarrow{X_\beta} - \vec{X} |, \overrightarrow{D_\delta} = | \overrightarrow{C_3} \cdot \overrightarrow{X_\delta} - \vec{X} | \quad \textbf{(3.2.5)}$$

$$\overrightarrow{X_1} = \overrightarrow{X_a} - \overrightarrow{A_1} \cdot \left(\overrightarrow{D_\alpha}\right), \overrightarrow{X_2} = \overrightarrow{X_B} - \overrightarrow{A_2} \cdot \left(\overrightarrow{D_\beta}\right), \overrightarrow{X_3} = \overrightarrow{X_\delta} - \overrightarrow{A_3} \cdot \left(\overrightarrow{D_\delta}\right) \quad \textbf{(3.2.6)}$$

$$\overrightarrow{X_{(t+1)}} = \frac{\left(\overrightarrow{X_1} + \overrightarrow{X_2} + \overrightarrow{X_3}\right)}{3} \quad \textbf{(3.2.7)}$$

**Algorithm:** Grey Wolf Optimizer

*Initialize Population of Wolves*
*Initial α, A, C and D*
*Calculate fitness for every Wolf*
$X_a$ *= best search agent*
$X_b$ *= second best search agent*
$X_\delta$ *= third best search agent*
***While*** *t < max number of iterations*
    ***For*** *each search agent*
        *Update the position of search agents using above equation*
    ***End for***
    *Update α, A, C and D*
    *Calculate the fitness of all search agents*
    *Update* $X_a$, $X_b$, $X_\delta$
    *t = t + 1*
***end while***
*return* $X_a$

**Γράφημα**: Λειτουργία αλγορίθμου Βελτιστοποίησης Γκρίζου Λύκου

Οι κινήσεις του σμήνους επηρεάζονται από τους συντελεστές A και C των παραπάνω σχέσεων. Κατά αυτόν τον τρόπο οι τιμές αυτών προσομοιάζουν την τελική κίνηση των μελών της αγέλης. Αναλυτικότερα, οι τιμές του A κυμαίνονται μεταξύ -1 και 1. Όταν η τιμή του |A| είναι μεγαλύτερή του 1 τότε η κίνηση που ακολουθεί το σμήνος είναι να αλλάζει την πορεία του σε άλλη κατεύθυνση για την εξεύρεση καλύτερης λύσης. Αντίθετα, όταν η τιμή του A πλησιάζει στο 0 τότε η κίνηση που ακολουθεί το σμήνος είναι η επιθετική προσέγγιση του σημείου. Ο συντελεστής C αποτελεί βοήθημα στον αλγόριθμο, ώστε να καταφέρνει να ξεπερνά τοπικά ελάχιστα. Πιο συγκεκριμένα, όταν το C > 1 δίνεται έμφαση περαιτέρω αναζήτησης στο σημείο που βρίσκεται το σμήνος ενώ όταν το C < 1 το σμήνος κινείται εκτός του σημείου όπου βρίσκονταν. Η ύπαρξη του συντελεστή μπορεί να κατευθύνει το σμήνος στην επικέντρωση των μελών σε ένα σημείο στο πεδίο λύσεων και από την άλλη την υποχώρηση εάν η λύση που μελετάται δεν έχει ουσιαστικό ενδιαφέρον αναζήτησης. Συνεπώς, η λειτουργία του αλγορίθμου βασίζεται σε τεχνικές αναζήτησης στο σύνολο του πεδίου τιμών και μετέπειτα με την κατάλληλη προσαρμογή των συντελεστών στην στοχευμένη αναζήτηση λύσεων κοντά στην περιοχή που επιλέγεται.

## 3.5 Αλγόριθμος Βελτιστοποίησης Φάλαινας

Ο αλγόριθμος βελτιστοποίησης φάλαινας είναι μεθευρετικός αλγόριθμος που προτάθηκε από τους Seyedali Mirjalili και Andrew Lewis (2016). Ο αλγόριθμος βασίζεται στις φυσικές διεργασίες των φαλαινών. Μία φάλαινα μπορεί να φτάσει σε μέγεθος έως και 30 μέτρα και να ζυγίζει μέχρι και 180 τόνους. Υπάρχουν 7 διαφορετικά είδη φαλαινών τα περισσότερα εκ των οποίων αποτελούν κυνηγοί. Θεωρούνται ιδιαίτερα έξυπνα είδη καθώς αναπτύσσουν περίπλοκες αλυσίδες σχέσεων μεταξύ τους καθώς και αναπτύσσουν πολύπλοκες φυσικές δυνατότητες. Η μορφολογία του εγκεφάλου τους τους επιτρέπει να αναπτύσσουν συναισθήματα, να κρίνουν, να επικοινωνούν αλλά και να αλληλοεπιδρούν με το περιβάλλον στο οποίο ζουν. Η ιδιαίτερα πολύπλοκη διαδικασία συλλογής και κατανάλωσης της τροφής τους αποτέλεσε το βασικό ερέθισμα ανάπτυξης του αλγορίθμου. Πιο συγκεκριμένα, οι φάλαινες καταδύονται περίπου 12 μέτρα κάτω από την επιφάνεια της θάλασσας, καταγράφουν μια σπειροειδή κίνηση γύρω από το θήραμα δημιουργώντας του συνεχή σύγχυση για τον τρόπο που θα κινηθούν. Στη συνέχεια, αφήνοντας όσο περισσότερα από τα ψάρια να μαζευτούν μέσα στο σπειροειδή ιστό που δημιουργούν ανοίγουν το στόμα τους καταπίνοντας όσο μεγαλύτερη ποσότητα ψαριών γίνεται.

Με βάση την παραπάνω περιγραφή οι δύο ερευνητές ανέπτυξαν τις σχέσεις για τις θέσεις και τις κινήσεις των φαλαινών. Η αρχική θέση του θηράματος θεωρείται η τρέχουσα βέλτιστη λύση. Οι θέσεις των φαλαινών ανανεώνεται σε σχέση με την θέση του καλύτερου κυνηγού.

$$\vec{D} = | \vec{C} \cdot \overrightarrow{X^*(t)} - \overrightarrow{X(t)} | \qquad \textbf{(3.4.1)}$$

$$\overrightarrow{X(t+1)} = \overrightarrow{X^*(t)} - \vec{A} \cdot \vec{D} \qquad \textbf{(3.4.2)}$$

Με τις μεταβλητές $\vec{A}, \vec{D}$ να αποτελούν διανύσματα συντελεστών κίνησης και η μεταβλητή $\overrightarrow{X^*(t)}$ να αποτελεί τη θέση του καλύτερου κυνηγού. Οι συντελεστές κίνησης $\vec{A}, \vec{D}$ υπολογίζονται ως εξής:

$$\vec{A} = 2 \cdot \vec{\alpha} \cdot \vec{r} - \vec{\alpha} \qquad \textbf{(3.4.3)}$$

$$\vec{C} = 2 \cdot \vec{r} \qquad \textbf{(3.4.4)}$$

Η μεταβλητή $\vec{\alpha}$ μειώνεται γραμμικά από το 2 στο 0 καθ' όλη τη διάρκεια των επαναλήψεων και η μεταβλητή $\vec{r}$ αποτελεί διάνυσμα με τυχαίες τιμές μεταξύ [0,1]. Η επόμενη διεργασία που θα έπρεπε να μοντελοποιηθεί είναι η αυτή της δημιουργίας ιστού φυσαλίδων, ώστε τελικά η φάλαινα να ξεκινήσει να καταπίνει τα

περικυκλωμένα ψάρια. Η αρχική κίνηση περικύκλωσης μειούμενης διαμέτρου επιτυγχάνεται μειώνοντας γραμμικά την τιμή του συντελεστή $\vec{a}$ όπως προαναφέρθηκε. Η σπειροειδής κίνηση των φαλαινών επιτυγχάνεται όταν συνδυάζεται η κίνηση της εκάστοτε φάλαινας με την καλύτερη φάλαινα του πληθυσμού. Η σπειροειδής κίνηση μοντελοποιείται ως εξής:

$$\overrightarrow{X(t+1)} = \overrightarrow{D'} \cdot e^{bl} \cdot \cos(2\pi l) + \overrightarrow{X^*(t)} \qquad \textbf{(3.4.5)}$$

Με τη μεταβλητή $\overrightarrow{D'}$ αν αναπαριστά τη απόσταση της εκάστοτε φάλαινας από την καλύτερη φάλαινα του πληθυσμού $\overrightarrow{D'} = |\overrightarrow{X^*(t)} - \overrightarrow{X(t)}|$, $b$ αποτελεί σταθερά και ο $l$ αναπαριστά τυχαίο αριθμό στο διάστημα [-1,1]. Καθώς η σπειροειδής κίνηση των φαλαινών συμβαίνει προς δύο αντίθετες κατευθύνσεις τα μέλη του πληθυσμού διαμοιράζονται, ώστε να περικυκλώσουν πλήρως τα ψάρια. Για να διαμοιραστούν τα μέλη του πληθυσμού επιλέγεται η εισαγωγή γεννήτριας αριθμών για την αναπαράσταση πιθανότητας, Έτσι, αν ο αριθμός είναι μεγαλύτερος ή ίσος του 0.5 επιλέγεται η σπειροειδής κίνηση σε σχέση το καλύτερο μέλος του πληθυσμού (3.4.5), ενώ αν είναι μικρότερος του 0.5 τότε επιλέγεται η απλή μετακίνηση σε σχέση με το καλύτερο μέλος του πληθυσμού (3.4.2).

$$\overrightarrow{X(t+1)} = \begin{cases} \overrightarrow{X^*(t)} - \vec{A} \cdot \vec{D} \text{ , } p < 0.5 \\ \\ \overrightarrow{D'} \cdot e^{bl} \cdot \cos(2\pi l) + \overrightarrow{X^*(t)} \text{ , } p >= 0.5 \end{cases} \qquad \textbf{(3.4.6)}$$

Το τελευταίο στάδιο μοντελοποίησης της κίνησης των φαλαινών έχει να κάνει με την αναζήτηση τροφής. Η κίνηση εισάγεται καθώς οι φάλαινες αναπτύσσουν το πεδίο αναζήτησης προς εύρεση καλύτερης τροφής.

$$\vec{D} = |\vec{C} \cdot \overrightarrow{X_{rand}} - \overrightarrow{X(t)}| \qquad \textbf{(3.4.7)}$$

$$\overrightarrow{X(t+1)} = \overrightarrow{X_{rand}} - \vec{A} \cdot \vec{D} \qquad \textbf{(3.4.8)}$$

Για αυτό το λόγο η κίνηση στο χώρο λύσεων δεν γίνεται σε σχέση με το καλύτερο μέλος του πληθυσμού, αλλά σε σχέση με ένα τυχαίο μέλος του πληθυσμού. Στόχος της συγκεκριμένης κίνησης είναι να εξετασθούν νέες θέσεις προς αποφυγή τοπικού ελαχίστου. Σημαντικές θεωρούνται για την εξεύρεση λύσεων οι τιμές που λαμβάνουν οι συντελεστές καθώς θα πρέπει να εξασφαλίζουν τόσο την εξερεύνηση στο σύνολο του πεδίου τιμών αλλά και η αξιοποίηση λύσεων που μπορούν να οδηγήσουν σε ολικό βέλτιστο.

**Algorithm:** Whale Optimization Algorithm

*Initialize Population of Wolves*
*Calculate fitness for every agent*
*$X^*$ = Best Search Agent*
***While*** *t < max number of iterations*
    *For each search agent*
        ***If1*** *(p < 0.5)*
            ***If*** *(|A| < 1)*
                *Update Position Using Equation* ***(3.4.1-2)***
            ***Else if2*** *(|A| < 1)*
                *Select Random Search Agent* $\overrightarrow{X_{rand}}$
                *Update Position Using Equation* ***(3.4.7-8)***
            ***end if2***
        ***Else if1*** *(p >= 0.5)*
            *Update Position Using Equation* ***(3.4.5)***
        ***End if1***
    ***End for***
    *Calculate Fitness of new population*
    *Update $X^*$ = Best Search Agent*
    *t = t + 1*
***end while***
*return $X^*$*

**Γράφημα**: Λειτουργία αλγορίθμου Βελτιστοποίησης Φάλαινας

## 3.6 Αλγόριθμος Βελτιστοποίησης Πεταλούδας Μονάρχη

Ο αλγόριθμος βελτιστοποίησης πεταλούδας μονάρχη προτάθηκε από τους Gai-Ge Wang, Suash Deb και Zhihua Cui το 2015. Ο αλγόριθμος βασίζεται στην διαδικασία μετανάστευσης των πεταλούδων μεταξύ δύο φωλιών. Η πεταλούδα μονάρχης συναντάται σε μέρη της βόρειας Αμερικής. Έχει χαρακτηριστικό πορτοκαλί με μαύρο χρώμα και η μορφολογία των φτερών τους διαφέρει από αρσενικό σε θηλυκό ως αναγνωριστικό του φύλλου τους. Το συγκεκριμένο είδος πεταλούδας διαθέτει ισχυρή δυναμική μιας και μπορεί να πετάει για πολλές ώρες καλύπτοντας μεγάλες αποστάσεις. Η χαρακτηριστική τους φυσική διεργασία είναι η δυνατότητα μετανάστευσης από τόπο σε τόπο για την εξασφάλιση καλύτερων καιρικών συνθηκών. Η συγκεκριμένη διεργασία συμβαίνει για τους μήνες του

χρόνου που η θερμοκρασία είναι ψυχρή, την άνοιξη οι θηλυκές πεταλούδες αφήνουν τα αυγά τους σε ασφαλή μέρη. Καθώς η αλγοριθμική διαδικασία μιμείται τις φυσικές διεργασίες του είδους οι κινήσεις που θα πρέπει να εξασφαλιστούν είναι: Μετακίνηση από μία περιοχή σε μία άλλη, δημιουργία τέκνων μέσω μετακίνησης από τη μία περιοχή στην άλλη, εξασφάλιση ότι τα ισχυρότερα μέλη του πληθυσμού θα περάσουν αναλλοίωτα στις επόμενες γενεές. Η αρχική διαδικασία μετανάστευσης μεταξύ δύο περιοχών υλοποιείται δημιουργώντας δύο πληθυσμιακές ομάδες και ανταλλάσσοντας μέλη μεταξύ αυτών των ομάδων. Οι δύο πληθυσμιακές ομάδες καλούνται Land1 και Land2, η μετανάστευση μεταξύ των δύο αυτών περιοχών συμβαίνει ως εξής: Την άνοιξη οι πεταλούδες μετακινούνται από την περιοχή 1 προς την περιοχή 2, ενώ τον Σεπτέμβρη ακολουθούν την αντίθετη κίνηση. Κατά αυτόν τον τρόπο, η παραμονή των πεταλούδων στην περιοχή 1 συμβαίνει για 7 μήνες και αντίστοιχα 5 στην περιοχή 2. Ο αριθμός των μελών των δύο πληθυσμιακών ομάδων υπολογίζεται με βάση τις παρακάτω σχέσεις.

$$NP_1 = ceil\,(p \cdot NP) \qquad NP_2 = NP - NP_1 \qquad \textbf{(3.5.1)}$$

Με $NP_1$ να αποτελεί τον αριθμό μελών της πρώτης πληθυσμιακής ομάδα, αντίστοιχα $NP_2$ για την δεύτερη και $NP$ ο αριθμός μελών του συνολικού πληθυσμού. Ο συντελεστής $p$ αποτελεί το λόγο μετανάστευσης, όπου με βάση την παρατήρηση των πεταλούδων παίρνει τιμή 5/12 μιας και η διεργασία συμβαίνει 5 μήνες το χρόνο. Η συνάρτηση $ceil$ στρογγυλοποιεί τον αριθμό $NP_1$ στον αμέσως επόμενο ακέραιο. Η διαδικασία μετανάστευσης αναπαρίσταται ως εξής:

$$x^{t+1}{}_{i,k} = x^t_{r1,k} \qquad \textbf{(3.5.2)}$$

Ο όρος $x^{t+1}{}_{i,k}$ υποδηλώνει το στοιχείο με αριθμό k του συνολικού πληθυσμού για την επανάληψη t+1 που λαμβάνει την τιμή του στοιχείου k για ένα τυχαίο μέλος r1 της πληθυσμιακής ομάδας 1 για την επανάληψη t. Η παραπάνω μετακίνησης συμβαίνει όταν η τιμή ενός συντελεστή r είναι μικρότερη από την τιμή του λόγου μετανάστευσης $p$. Ο συντελεστής r υπολογίζεται σε σχέση με συντελεστή χρονικής διάρκειας ενός έτους σε μήνες δηλαδή 12 μήνες (peri = 1.2):

$$r = rand \cdot peri \qquad \textbf{(3.5.3)}$$

Αντίθετα όταν η τιμή του συντελεστή είναι μεγαλύτερη του λόγου μετανάστευσης το τυχαίο μέλος λαμβάνεται από την πληθυσμιακή ομάδα 2.

$$x^{t+1}{}_{i,k} = x^t_{r2,k} \qquad \textbf{(3.5.4)}$$

Η μεταβολή του λόγου μετανάστευσης μπορεί να μεταβάλλει σημαντικά την διαδικασία καθώς όταν λαμβάνεται μεγάλη τιμή του δείκτη δύναται έμφαση στην επιλογή μελών από την πρώτη πληθυσμιακή ομάδα, ενώ όταν ο δείκτης είναι μικρός τότε δύναται έμφαση στην δεύτερη πληθυσμιακή ομάδα. Εκτός της διαδικασίας μετανάστευσης για την αλγοριθμική διαδικασία υιοθετείται και διαδικασία μετακίνησης. Η συγκεκριμένη κίνηση έχει να κάνει με την διεργασία μετακίνησης των πεταλούδων από τη μία περιοχή στην άλλη καλύπτοντας μεγάλες αποστάσεις. Η κίνηση που λαμβάνει υπόψιν τη μετακίνηση του μέλους του πληθυσμού σε σχέση με το καλύτερο μέλος.

$$x^{t+1}{}_{j,k} = x^{t}_{best,k} \qquad \textbf{(3.5.5)}$$

Ο όρος $x^{t+1}{}_{j,k}$ υποδηλώνει το στοιχείο με αριθμό j του συνολικού πληθυσμού για την επανάληψη t+1 που λαμβάνει την τιμή του στοιχείου k για το καλύτερο μέλος best της πληθυσμιακής ομάδας για την επανάληψη t. Η διαδικασία ακολουθεί ίδια δομή με την παραπάνω. Μιας και η κίνηση 3.5.4 συμβαίνει όταν ένας τυχαίος αριθμός είναι μεγαλύτερο από τον λόγο μετανάστευσης ενώ σε αντίθετη περίπτωση η διαδικασία μετακίνησης λαμβάνει τυχαίο μέλος από τη δεύτερη πληθυσμιακή ομάδα ($r_3 \in \{1, 2, 3 \dots \dots \dots NP_2\}$).

$$x^{t+1}{}_{j,k} = x^{t}_{r3,k} \qquad \textbf{(3.5.6)}$$

Τέλος, στον αλγόριθμο έχει προστεθεί και σχέση μεταβολής της θέσης ενός μέλους, ώστε να υπάρχει και τρόπος μετακίνησης διαφορετικού τύπου από την συμβατική μετανάστευση όπως στις προηγούμενες σχέσεις. Η κίνηση συμβαίνει όταν ένας τυχαίος αριθμός είναι μεγαλύτερος από έναν συντελεστή προσαρμογής BAR.

$$x^{t+1}{}_{j,k} = x^{t+1}_{j,k} + a \cdot (dx_k - 0.5) \qquad \textbf{(3.5.7)}$$

$$dx = Levy\left(x^{t}_{j}\right) \qquad \textbf{(3.5.8)}$$

$$a = \frac{Smax}{t^2} \qquad \textbf{(3.5.9)}$$

Στη σχέση 3.5.6 ο όρος $dx_k$ αναπαριστά το βήμα αναζήτησης για την επόμενη θέση και ακολουθεί τη κατανομή Levy. Ο όρος α στην σχέση 3.5.8 αναπαριστά συντελεστή βαρύτητας με Smax το μέγιστο βήμα αναζήτησης. Όσο μεγαλύτερο είναι το βήμα αναζήτησης τόσο πιο πολύ διευρύνεται ο χώρος αναζήτησης λύσεων.

**Algorithm:** Monarch Butterfly Optimization

*1. Initialize the population, calculate monarch butterfly members in*
*2. Land 1, Land 2 ($NP_1$, $NP_2$). Set Smax, BAR, peri and p.*
*3. Calculate fitness of individuals*
*4.* ***While*** *t < max number of generations*
*5.* *Sort individuals based on their fitness*
*6.* *Divide population using equation* ***3.5.1*** *in Land 1, Land 2*
*7.* ***For*** *i = 1 to* $NP_1$
*8.* ***For*** *k = 1 to NP*
*9.* *Generate random number*
*10.* *r = rand * peri*
*11.* ***If*** *r =< p then*
*12* *Randomly Select butterfly from Land 1*
*13.* *Migrate using operator* ***3.5.2***
*14.* ***Else***
*15.* *Randomly Select butter from Land 2*
*16.* *Migrate using operator* ***3.5.4***
*17.* ***Endif***
*18.* ***End for***
*19.* ***End for***
*20.* ***For*** *i = 1 to* $NP_2$
*21.* ***For*** *k = 1 to NP*
*22.* *Generate random number*
*23.* ***If*** *rand =< p then*
*24.* *Migrate using operator* ***3.5.4***
*25.* ***Else***
*26.* *Randomly Select butter from Land 2*
*27.* *Migrate using operator* ***3.5.5***
*28.* ***If*** *rand < BAR*
*29.* *Adjust Butterfly position using Eq* ***3.5.6***
*30.* ***End if***
*31.* ***End if***
*32.* ***End for, End for***
*34.* *Combine Land 1 – Land 2 to a whole population*
*35.* *Calculate fitness of new population*
*36.* *t = t + 1*
*37.* ***end while***

**Γράφημα**: Λειτουργία αλγορίθμου Βελτιστοποίησης Πεταλούδας Μονάρχη

## 3.7 Αλγόριθμος Νυχτοπεταλούδας

Ο αλγόριθμος νυχτοπεταλούδας προτάθηκε από τον Seyedali Mirjalili (2015). Ο αλγόριθμος βασίζεται στον τρόπο πλοήγησης των νυχτοπεταλούδων. Οι νυχτοπεταλούδες εντάσσονται στον πληθυσμό των πεταλούδων ο οποίος αριθμεί περίπου 160,000 παρόμοια είδη. Οι βασικές τους λειτουργίες είναι η δημιουργία απογόνων και η ενηλικίωση. Η διεργασία από την οποία είναι εμπνευσμένος ο αλγόριθμος αποτελεί το περίπλοκο σύστημα πλοήγησής τους μιας και χρησιμοποιούν το φως του φεγγαριού για να πλοηγηθούν.

Πιο συγκεκριμένα, η πλοήγηση τους λαμβάνει ως σταθερό σημείο το φως του φεγγαριού καθώς αποτελεί μια δυνατή πηγή φωτός. Το ενδιαφέρον χαρακτηριστικό της πορείας τους είναι η περίπτωση εγκλωβισμού του φωτός λόγω της αυξημένης δόμησης στις πόλεις. Ο εγκλωβισμός του φωτός μαζί με την παρουσία των τεχνητών φώτων μπερδεύει τις νυχτοπεταλούδες αναγκάζοντάς τες σε μια σπειροειδή κίνηση γύρω από την πηγή του φωτός.

Στην περίπτωση της μαθηματικής μοντελοποίησης της παραπάνω συμπεριφοράς η κάθε λύση αναπαριστά μία πηγή φωτός. Οι νυχτοπεταλούδες προσεγγίζουν την πιο φωτεινή πηγή κάθε επανάληψης δημιουργώντας ένας σμήνος που κινείται με σπειροειδή μορφή γύρω από την λύση. Αν μετά από κάποια επανάληψη προκύψει φωτεινότερη πηγή, δηλαδή καλύτερη λύση, οι πεταλούδες πετούν στην επόμενη θέση προσεγγίζοντάς την επαναλαμβάνοντας την αναζήτηση. Για την μαθηματική μοντελοποίηση της σπειροειδούς κίνησης επιλέχθηκε η λογαριθμική σπείρα.

$$S(M_i, F_j) = D_i \cdot e^{bt} \cdot \cos(2\pi t) + F_j \qquad \textbf{(3.6.1)}$$

Με $S(M_i, F_j)$ να αναπαριστά τη θέση για τη νυχτοπεταλούδα $M_i$ σε σχέση με τη φλόγα $F_j$. Με τη φλόγα $F_j$ να αποτελεί την αναπαράσταση του αρχικού πληθυσμού ταξινομημένο με βάση την τιμή της αντικειμενικής συνάρτησης. Κατά αυτόν τον τρόπο, η πρώτη φλόγα είναι η πεταλούδα με την καλύτερη τιμή αντικειμενικής συνάρτησης για το πρόβλημα βελτιστοποίησης. Όπου $D_i$ η απόσταση της νυχτοπεταλούδας από τη φλόγα, b σταθερά ρύθμισης του μεγέθους της σπείρας. Ο υπολογισμός της απόσταση της νυχτοπεταλούδας από τη φλόγα:

$$D_i = \left| F_j - M_i \right| \qquad \textbf{(3.6.2)}$$

Ο όρος t αναπαριστά τυχαίους αριθμούς στο διάστημα [r,1] με r γραμμικά μειούμενο αριθμός στο διάστημα [-1,-2]. Ο αριθμός των φλογών μειώνεται με το πέρας των

επαναλήψεων για την αποφυγή εγκλωβισμών σε τοπικό ελάχιστο, καθώς και για την διερεύνηση του πεδίου αναζήτησης προς εύρεση καλύτερων λύσεων.

$$Flame_{number} = round\left(N - l \cdot \frac{N-1}{T}\right) \quad \textbf{(3.6.3)}$$

Στην παραπάνω σχέση ο όρος $l$ αναπαριστά την τρέχουσα επανάληψη, N ο μέγιστος αριθμός φλογών και T ο συνολικός αριθμός επαναλήψεων.

**Algorithm:** Moth Flame Optimization

```
1. Initialize Population of Moths
2. Calculate fitness for every agent
3. Flames = Sorted Population
4. While t < max number of iterations
5.     For i = 1: n
6.         For j = 1: d
7.             Update r, t
8.             Calculate Distance using Eq (3.6.2) to the
9.             corresponding moth
10.            Update M (i, j) using Eq. (3.6.1) and (3.6.3)
11.        End For
12     End For
13.    Calculate Fitness of New Population
14.    Flames = Sorted Population
15.    t = t + 1
16. end while
```

**Γράφημα**: Λειτουργία αλγορίθμου Βελτιστοποίησης Νυχτοπεταλούδας

# 4. Αλγόριθμοι Τοπικής Αναζήτησης

## 4.1 Εισαγωγή

Η τοπική αναζήτηση βασίζεται στην αρχαιότερη μέθοδο βελτιστοποίησης, στη μέθοδο δοκιμής και σφάλματος. Οι αλγόριθμοι τοπικής αναζήτησης έχουν εφαρμοστεί σε διάφορα προβλήματα βελτιστοποίησης παρουσιάζοντας επιτυχημένη εφαρμογή. Στην περίπτωση των προβλημάτων δρομολόγησης οχημάτων έχουν αναπτυχθεί διάφοροι αλγόριθμοι τοπικής αναζήτησης που σχετίζονται είτε με στοχευμένη αναζήτηση σε μια συγκεκριμένη γειτονιά λύσεων, είτε ευρύτερο σε όλο το πεδίο λύσεων. Στη συγκεκριμένη εργασία έχουν εφαρμοσθεί αλγόριθμοι τοπικής αναζήτησης σε δύο επίπεδα λύσεων. Πιο συγκεκριμένα, οι δύο κατηγορίες αποτελούν: Κινήσεις σε μία διαδρομή και κινήσεις σε περισσότερες διαδρομές. Η συγκεκριμένη μέθοδος αυξάνει σημαντικά τις πιθανότητες βελτίωσης των αρχικών λύσεων που προκύπτουν καθαρά από την εφαρμογή των αλγορίθμων. Η εφαρμογή των αλγορίθμων τοπικής αναζήτησης γίνεται σε συγκεκριμένο αριθμό υποψήφιων λύσεων λόγω του μεγάλου υπολογιστικού κόστους που θα είχε η ολιστική εφαρμογή σε όλο τον αρχικό πληθυσμό. Παρακάτω παρουσιάζονται οι αλγόριθμοι και οι κινήσεις τοπικής αναζήτησης που εφαρμόσθηκαν.

## 4.2 1-1 Swap, 2 – 2 Swap

Η πρώτη κίνηση που εντάχθηκε στην τοπική αναζήτηση είναι η εναλλαγή κόμβων. Η διαδικασία εφαρμογής της συγκεκριμένης κίνησης είναι η εξής: Επιλέγεται, αρχικά, με τυχαίο τρόπο ένας αριθμός που αντιστοιχεί στη θέση ενός πελάτη εντός μίας διαδρομής, μετέπειτα επιλέγεται ακόμα ένας αριθμός, που δεν μπορεί να είναι ίδιος με τον προηγούμενο, που αντιστοιχεί σε μία άλλη θέση ενός πελάτη εντός μίας διαδρομής και τέλος εφαρμόζεται εναλλαγή αυτών των δύο κόμβων.

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 1 | 4 | 3 | 7 | 5 | 6 | 2 | 1 |

Έστω η παραπάνω διαδρομή από και προς την αποθήκη. Μέσω γεννήτριας τιμών προκύπτει rand1 = 3 και rand2 = 7. Οι θέσεις με βάση τις τιμές που προέκυψαν από

την γεννήτρια τιμών αντιστοιχούν στη θέση path(3) = 3 και path(7) = 2, εφαρμόζεται εναλλαγή μεταξύ των δύο τιμών.

| 1 | 4 | 3 | 7 | 5 | 6 | 2 | 1 |
|---|---|---|---|---|---|---|---|

| 1 | 4 | 2 | 7 | 5 | 6 | 3 | 1 |
|---|---|---|---|---|---|---|---|

**Γράφημα**: Κίνηση 1-1 swap εντός μίας διαδρομής

Η κίνηση 1 – 1 Swap έχει ίδια δομή με την κίνηση 2 – 2 Swap αλλά αντί να ανταλλάσσονται δύο κόμβοι, η κίνηση συμβαίνει για δύο τόξα. Η συγκεκριμένη κίνηση πολλές φορές οδηγεί στο συνδυασμό διαφορετικών κόμβων εισαγωγής καταλήγοντας σε καλύτερα αποτελέσματα. Παρακάτω παρουσιάζεται ενδεικτικά με πρώτο τυχαίο κόμβο επιλογής εκείνον στην θέση 3 και δεύτερο στη θέση 5.

| 1 | 4 | 3 | 7 | 5 | 6 | 2 | 1 |
|---|---|---|---|---|---|---|---|

| 1 | 4 | 6 | 2 | 5 | 3 | 7 | 1 |
|---|---|---|---|---|---|---|---|

**Γράφημα**: Κίνηση 2 - 2 swap εντός μίας διαδρομής

## 4.3 2 – 0 Relocate, 1 – 0 Relocate

Η κίνηση 2-0 Relocate περιλαμβάνει την διαγραφή και επανατοποθέτηση ενός τόξου σε άλλη θέση εντός μιας διαδρομής. Η διαδικασία εφαρμογής της συγκεκριμένης κίνησης είναι η εξής: Αρχικά, με τυχαίο τρόπο επιλέγεται ένας αριθμός ο οποίος αντικατοπτρίζει μία θέση εντός της διαδρομής, μαζί του αποθηκεύτε και ο επόμενος σε αυτόν αριθμό ώστε να δημιουργηθεί ένα τόξο, μετέπειτα επιλέγεται και πάλι τυχαία ένας αριθμός οποίος αντιστοιχεί στη θέση όπου θα επανατοποθετηθεί το αρχικό τόξο.

| 1 | 4 | 3 | 7 | 5 | 6 | 2 | 1 |
|---|---|---|---|---|---|---|---|

Έστω η παραπάνω διαδρομή από και προς την αποθήκη. Μέσω γεννήτριας τιμών προκύπτει rand1 = 2 και η επόμενη σε αυτή θέση αντιστοιχεί στη θέση 3. Με βάση τις τιμές που προέκυψαν οι αντίστοιχοι κόμβοι είναι οι εξής: path(2) = 4 και path(2+1) = 3.

Και πάλι με γεννήτρια τιμών προκύπτει rand2 = 6, κατά συνέπεια το τόξο 4-3 θα επανατοποθετηθεί στην θέση 6 του διανύσματος.

| 1 | 4 | 3 | 7 | 5 | 6 | 2 | 1 |
|---|---|---|---|---|---|---|---|

| 1 | 7 | 5 | 6 | 2 | 4 | 3 | 1 |
|---|---|---|---|---|---|---|---|

**Γράφημα**: Κίνηση 2-0 relocate εντός μίας διαδρομής

Η κίνηση 1 – 0 Relocate έχει ίδια δομή με την κίνηση 2 – 0 αλλά αντί να διαγράφεται και να επανατοποθετείται ένα τόξο η κίνηση συμβαίνει για ένα κόμβο. Ο λόγος που εντάσσεται η συγκεκριμένη κίνηση είναι διότι πολλές φορές η μετάθεση ενός κόμβου σε άλλη θέση μπορεί να τον εντάξει σε μια διαδρομή και εν τέλει να οδηγήσει σε μείωση της απόστασης που θα διανύσει το όχημα. Παρακάτω, παρουσιάζεται ενδεικτικά με τυχαίο κόμβο επιλογής εκείνον στην θέση 3 και θέση επανατοποθέτησης τη θέση 6.

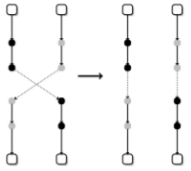
| 1 | 4 | 3 | 7 | 5 | 6 | 2 | 1 |
|---|---|---|---|---|---|---|---|

| 1 | 4 | 7 | 5 | 6 | 3 | 2 | 1 |
|---|---|---|---|---|---|---|---|

**Γράφημα**: Κίνηση 1-0 relocate εντός μίας διαδρομής

## 4.4 Μέθοδος 2 – Opt

Ο αλγόριθμος 2 – opt είναι ο πιο γνωστός αλγόριθμος τοπικής αναζήτησης για προβλήματα δρομολόγησης. Η κεντρική ιδέα του αλγορίθμου περιλαμβάνει την διαγραφή 2 τόξων και την επανασύνδεση δύο νέων μονοπατιών ούτως ώστε να δημιουργηθεί μία νέα διαδρομή. Το πλεονέκτημα του αλγορίθμου 2 – opt σε σχέση με τις προηγούμενες κινήσεις τοπικής αναζήτησης έχει να κάνει με την απόρριψη νέων μονοπατιών που οδηγούν σε χειρότερες λύσεις. Πιο συγκεκριμένα, η κίνηση 2 opt ελέγχεται ώστε να διαπιστωθεί αν το '' κέρδος '' του νέου μονοπατιού είναι μεγαλύτερο από την μη μετατροπή.

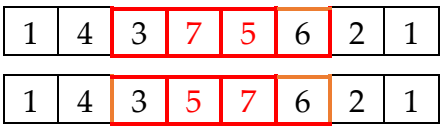


**Γράφημα**: Κίνηση 2 – Opt, *Πηγή: Knowledge-guided local search Florian Arnold, Kenneth Sorensen*

**Algorithm:** 2 Opt Algorithm

```
1.    for i in 0 ... N-1
2.    X1 = tour[i]
3.    X2 = tour[(i+1)]
4.    for j in 0 ... N-1
5.    Y1 = tour[j]
6.          Y2 = tour[(j+1]
7.          If distance [(X1, Y1) + (X2, Y2) < (X1, X2) + (Y1, Y2)]
8.           Exchange (X2, Y1)
9.          End
10.   End
11.   End
```

**Γράφημα**: Λειτουργία του αλγορίθμου 2 – Opt




**Γράφημα**: Κίνηση 2 - Opt εντός μίας διαδρομής

## 4.5 Τοπική Αναζήτηση σε Πολλαπλές Διαδρομές

Με τον συγκεκριμένο τρόπο τοπικής αναζήτησης επιλέγονται δύο διαδρομές από την αρχική λύση. Οι διαδρομές καταστρέφονται και εντός τους επιλέγεται με βάση μίας γεννήτριας τιμών μία από τις κινήσεις τοπικής αναζήτησης 1-1 Swap, 2 – 2 Swap, 2 – 0 Relocate, 1 – 0 Relocate. Αφού πραγματοποιηθεί η κίνηση και επανατοποθετηθούν οι διαδρομές στην αρχική, ώστε να δημιουργηθεί μία ολοκληρωμένη λύση ελέγχεται αν η κίνηση τοπικής αναζήτησης οδηγήσει σε καλύτερη λύση. Αν, τελικώς, η λύση είναι καλύτερη στις νέες διαδρομές εφαρμόζεται τοπική αναζήτηση 3 – Opt, σε άλλη περίπτωση συνεχίζεται η αναζήτηση και πάλι με εφαρμογή κινήσεων τοπικής αναζήτησης ώσπου να ολοκληρωθούν οι απαιτούμενες επαναλήψεις. Αξίζει να σημειωθεί ότι σε κάθε επανάληψη μεταξύ των διαδρομών επιλέγεται η εφαρμογή μίας και μόνο αλλαγής από μία από τις κινήσεις. Ο λόγος

που επιλέγεται μία αλλαγή έχει να κάνει με το γεγονός ότι πολλαπλές κινήσεις μπορούν να καταστρέψουν μια καλύτερη λύση που προέκυψε σε προηγούμενες.

**Procedure:** Inter Route Local Search

```
1.   While iterations < max iterations
2.       p = rand
3.       If p < 0.25
4.           New_Path = 1 – 1 Swap (Path)
5.       Else if p < 0.5 & p > 0.25
6.           New_Path = 2 – 2 Swap (Path)
7.       Else if p < 0.75 & p > 0.5
8.           New_Path = 2 – 0 Relocate (Path)
9.       Else if p < 1 & p > 0.75
10.          New_Path = 1 – 0 Relocate (Path)
11.  End While
12.  Distance = fitness_score (Path)
13.  New_Distance = fitness_score(New_Path)
14.  If New_Distance < Distance
15.      New_Path = Opt2 (New_Path)
16.      Path = New_Path
17.      Iterations = 1
18.  Else
19.      Iterations = Iterations + 1
20.  End If
```

**Γράφημα**: Λειτουργία του αλγορίθμου Τοπικής Αναζήτησης σε Πολλαπλές Διαδρομές

## 4.6 Τοπική Αναζήτηση σε μια Διαδρομή

Η διαδικασία τοπικής αναζήτησης εντός μίας διαδρομής αποτελεί την πιο βασική κίνηση τοπικής αναζήτησης που μπορεί να υλοποιηθεί σε ένα πρόβλημα δρομολόγησης. Με την συγκεκριμένη κίνηση δεν διαταράσσονται οι κόμβοι που εντάσσονται σε μία διαδρομή αλλά τροποποιείται κατάλληλα η αλληλουχία επίσκεψής τους ούτως ώστε να υιοθετηθεί η σειρά εκείνη που τελικά θα αποδώσει την μικρότερη απόσταση. Πιο συγκεκριμένα, στην μέθοδο αυτή δεν έχουν ενταχθεί κινήσεις τοπικής αναζήτησης όπως στην προηγούμενη παρά μόνο η τοπική αναζήτηση 2 – Opt. Μιας και οι κόμβοι εντός μιας διαδρομής είναι λιγότεροι σε σχέση με πολλαπλές ο αλγόριθμος προσδίδει ικανοποιητική βελτίωση σε κάθε διαδρομή

και με μικρό υπολογιστικό κόστος. Παρακάτω παρουσιάζεται συνοπτικά η λειτουργία της μεθόδου.

**Procedure:** Intra Route Local Search

```
1.    For Each Route
2.        New_Route = Opt2 (Route)
3.        New_Path = [ New_Path, New_Route]
4.    End For
```

**Γράφημα**: Λειτουργία του αλγορίθμου Τοπικής Αναζήτησης σε μία Διαδρομή

# 5. Μοντελοποίηση Προβλήματος στο Προγραμματιστικό Περιβάλλον

## 5.1 Εισαγωγή Δεδομένων

Αρχικά, για το πρόβλημα δρομολόγησης οχημάτων με περιορισμό χωρητικότητας και χρόνους εξυπηρέτησης πελατών θα πρέπει να οριστούν οι αποστάσεις ανάμεσα στους πελάτες. Στα σετ δεδομένων που εφαρμόσθηκαν οι αλγόριθμοι, οι πελάτες αναπαρίστανται από θέσεις στο χώρο με συντεταγμένες Χ,Υ. Ακόμα, κάθε πελάτης έχει τιμή απαιτούμενων μονάδων εξυπηρέτησης για τον περιορισμό χωρητικότητας καθώς και συγκεκριμένη τιμή χρόνου που απαιτείται για να εξυπηρετηθεί. Οι αποστάσεις μεταξύ των πελατών υπολογίζονται με βάση την ευκλείδεια απόσταση των σημείων ο τύπος της οποίας παρουσιάζεται παρακάτω:

$$distance(i,j) = sqrt\left(\left(coord(i,1) - coord(j,1)\right)^2 + \left(coord(i,2) - coord(j,2)\right)^2\right) \quad \textbf{(5.1)}$$

Η παραπάνω σχέση εφαρμόζεται για κάθε σημείο με συντεταγμένες Χ,Υ οι τιμές των οποίων εντάσσονται σε έναν πίνακα distance. Ενδεικτικά παρουσιάζεται η μορφή των συντεταγμένων 10 σημείων από πρώτο σετ δεδομένων για το πρόβλημα δρομολόγησης.

| | |
|---|---|
| 30 | 40 |
| 37 | 52 |
| 49 | 49 |
| 52 | 64 |
| 20 | 26 |
| 40 | 30 |
| 21 | 47 |
| 17 | 63 |
| 31 | 62 |
| 52 | 33 |

Κατά συνέπεια η σχέση **5.1** εφαρμόζεται επαναληπτικά για κάθε σημείο λαμβάνοντας τον παραπάνω πίνακα συντεταγμένων με την πρώτη στήλη να αντιστοιχεί στα σημεία για τον Χ άξονα και η δεύτερη στήλη για τον Υ. Σε κάθε σετ υπάρχουν ακόμα στα δεδομένα εισαγωγής ένας αριθμός n που αντιστοιχεί στον αριθμό των πελατών, ένα διάνυσμα n x 1 που αντιστοιχεί στις απαιτητές μονάδες προϊόντων για την εξυπηρέτηση κάθε πελάτη, καθώς και ένα διάνυσμα για τον χρόνο εξυπηρέτησης με n τιμές και 0 για τον κόμβο αφετηρίας. Για την αποθήκη,

δηλαδή τον κόμβο αρχής, η ζήτηση έχει τιμή 0 όπως και η τιμή για του χρόνου εξυπηρέτησης.

## 5.2 Διαδικασία Εφαρμογής Αλγορίθμων – Αναπαράσταση Λύσεων

Η διαδικασία εφαρμογής αλγορίθμων στην συγκεκριμένη αναφορά είχε τρία στάδια. Στο πρώτο στάδιο, μιας και οι αλγόριθμοι που εφαρμόστηκαν έχουν χαρακτηριστικά σμήνους, δημιουργούνται τυχαίες λύσεις ώστε να αποτελέσουν τον αρχικό πληθυσμό του σμήνους. Έχει επιλεχθεί ως μέγεθος αρχικού πληθυσμού τα 50 μέλη. Οι λύσεις που συγκαταλέγονται στον αρχικό πληθυσμό έχουν εξαχθεί με τυχαίο τρόπο. Συγκεκριμένα, παρατίθενται τυχαία πελάτες και μετέπειτα υπολογίζεται η τιμή – συνολική απόσταση της αντικειμενικής συνάρτησης. Το επόμενο στάδιο περιλαμβάνει την επαναληπτική εφαρμογή των εξισώσεων θέσης του σμήνους για τον εκάστοτε αλγόριθμο. Τα αποτελέσματα που έχουν προκύψει και θα παρουσιαστούν παρακάτω έχουν εξαχθεί από εφαρμογή των αλγορίθμων για 50 και 100 επαναλήψεις. Στο τρίτο στάδιο, περιλαμβάνεται η εφαρμογή των μεθόδων τοπικής αναζήτησης με βάση τις διαδικασίες που παρουσιάστηκαν στο προηγούμενο κεφάλαιο. Πιο συγκεκριμένα, εφαρμόζεται αρχικά η διαδικασία τοπικής αναζήτησης για πολλαπλές διαδρομές και μετέπειτα για κάθε διαδρομή ξεχωριστά. Οι μέθοδοι τοπικής αναζήτησης εφαρμόζονται συνολικά για όλα τα μέλη του πληθυσμού. Επιλέχθηκε η συνολική εφαρμογή μετά από εξέταση διαφόρων τακτικών σε περιορισμένη εφαρμογή μελών του πληθυσμού δεν παρήχθησαν καλύτερα αποτελέσματα. Οι αλγόριθμοι νοημοσύνης σμήνους έχουν ως χαρακτηριστικό την αναπαράσταση των μελών του πληθυσμού ως θέσεις εντός του πεδίου λύσεων. Οι συγκεκριμένοι αλγόριθμοι, κατά συνέπεια, εφαρμόζονται κυρίως σε συνεχή προβλήματα βελτιστοποίησης. Η εφαρμογή των αλγορίθμων σε διακριτά προβλήματα, όπως το πρόβλημα δρομολόγησης οχημάτων, απαιτεί διαδικασίες αναπαράστασης – μετατροπής των λύσεων από συνεχής σε διακριτές και αντιστρόφως. Στη συγκεκριμένη παρουσίαση η μέθοδος που επιλέχθηκε για τη μετατροπή από διακριτές σε συνεχής τιμές είναι η διαίρεση των λύσεων με τον αριθμό πελατών. Η διαδικασία ακολουθήθηκε παρουσιάζεται παρακάτω για μία τυχαία διαδρομή 7 πελατών.

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 1 | 4 | 3 | 7 | 5 | 6 | 2 | 1 |

Έστω η παραπάνω αρχική διαδρομή, κάθε πελάτης της διαδρομής θα διαιρεθεί με τον αριθμό πελατών.

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 0.142 | 0.571 | 0.428 | 1 | 0.714 | 0.857 | 0.285 | 0.142 |

Με την μετατροπή του αρχικού διανύσματος σε συνεχές μπορούν να εφαρμοστούν οι εξισώσεις θέσεων για κάθε αλγόριθμο. Καθώς προκύπτουν νέες τιμές για κάθε θέση του διανύσματος το διάνυσμα θα μετατραπεί εκ νέου με την μεγαλύτερη τιμή να αντιστοιχεί στον τελευταίο πελάτη και την μικρότερη τιμή στον πρώτο πελάτη. Έστω, ότι οι τιμές του διανύσματος μεταβάλλονται με τυχαίο τρόπο ως εξής:

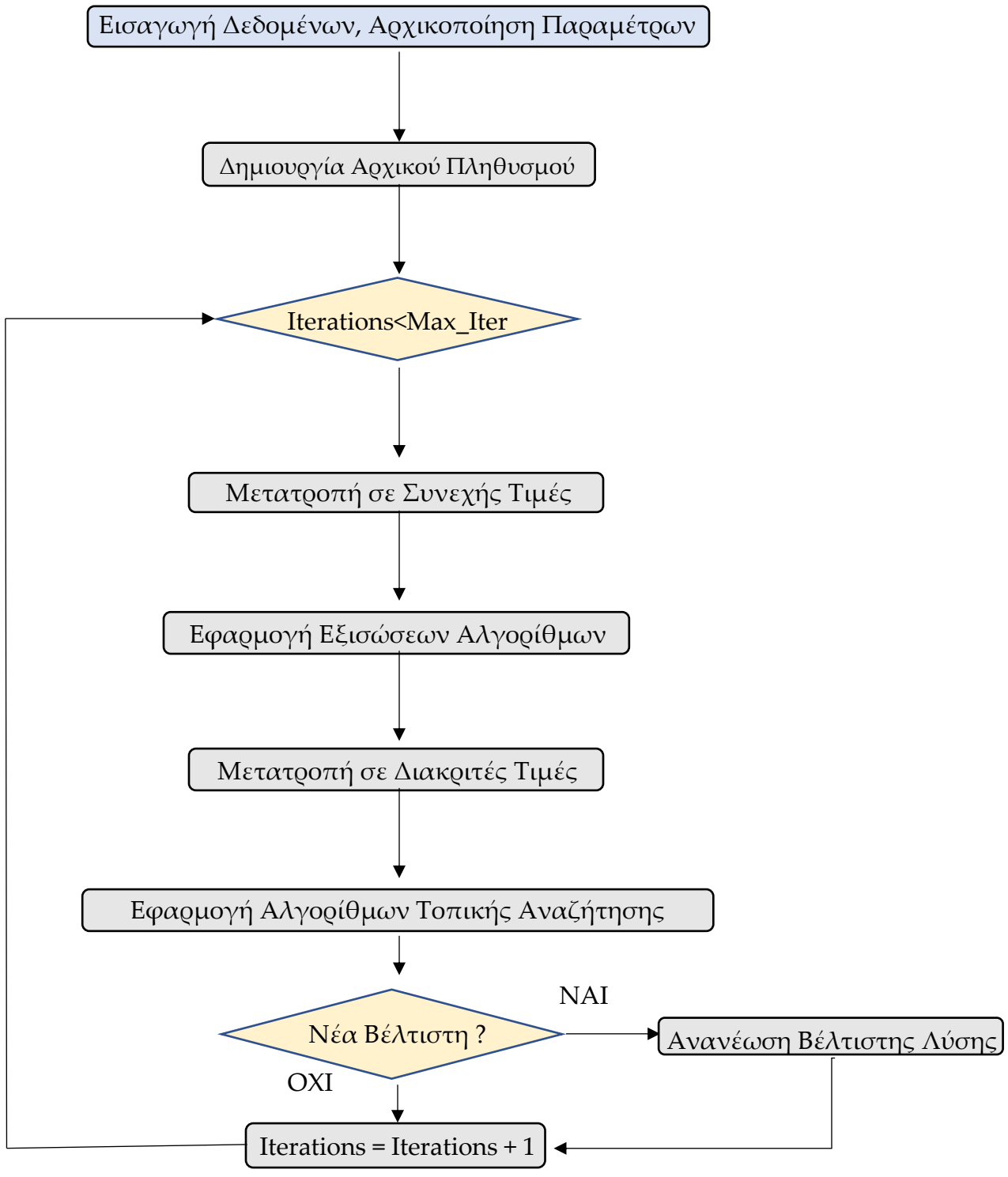
| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 0.41 | 0.667 | 0.981 | 0.760 | 0.225 | 0.198 | 0.528 | 0.41 |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 1 | 5 | 7 | 6 | 3 | 2 | 4 | 1 |

Στο παραπάνω διάνυσμα διακριτών τιμών μπορούν να εφαρμοστούν οι αλγόριθμοι τοπικής αναζήτησης για τη συνέχιση της διαδικασίας. Η διαδικασία επαναλαμβάνεται για κάθε επανάληψη μέχρι το μέγιστο αριθμό επαναλήψεων.

## 5.3 Χαρακτηριστικά Διαδικασίας Εφαρμογής Αλγορίθμων

Ο προγραμματισμός των αλγορίθμων έγινε σε περιβάλλον MATLAB. Οι εφαρμογές των αλγορίθμων έγιναν σε υπολογιστή με επεξεργαστή Inter® Core TM i5-8265U CPU @ 1.60 GHz. Η εγκατεστημένη μνήμη του υπολογιστή είναι 8 GB. Οι αλγόριθμοι εφαρμόστηκαν για 50 και 100 επαναλήψεις. Τα αποτελέσματα που παρουσιάζονται παρακάτω προέκυψαν εφαρμόζοντας 5 δοκιμές για 50 και 100 επαναλήψεις αντίστοιχα. Οι αλγόριθμοι εφαρμόστηκαν στα 14 βασικά προβλήματα δρομολόγησης (Benchmark Instances). Οι αλγόριθμοι τοπικής αναζήτησης εφαρμόζονταν σειριακά για το σύνολο του πληθυσμού και για κάθε επανάληψη. Ο αριθμό επαναλήψεων των αλγορίθμων τοπικής αναζήτησης είναι ίσος με 1000 επαναλήψεις. Εφαρμόστηκαν δύο είδη τοπικής αναζήτησης με τη μορφή που παρουσιάστηκαν στα προηγούμενα κεφάλαια Τα βέλτιστα αποτελέσματα κάθε επανάληψης αποθηκεύονταν και με το πέρας των συνολικών επαναλήψεων προέκυπτε η καλύτερη.




**Γράφημα:** Διάγραμμα Ροής Διαδικασίας Εφαρμογής Αλγορίθμων

# 6. Ταυτότητα Δεδομένων – Υπολογιστικά Αποτελέσματα

## 6.1 Ταυτότητα Δεδομένων

Η αξιολόγηση των αλγορίθμων που παρουσιάστηκαν στα προηγούμενα κεφάλαια θα ερευνηθεί από την εφαρμογή τους στα 14 προβλήματα αναφοράς που προτάθηκαν από του Christoforides, Mingozzi και Toth (1979) για το δρομολόγησης οχημάτων με περιορισμό χωρητικότητας και χρόνους εξυπηρέτησης πελατών. Πρόκειται για 14 προβλήματα με εύρος χωρητικότητας από 160 μονάδες έως 200, η τιμή των χρόνων εξυπηρέτησης κυμαίνεται από 0 έως 90 χρονικές μονάδες. Η δομή των προβλημάτων είναι η εξέταση επτά βασικών προβλημάτων με άπειρο χρόνο παραμονής σε κάθε διαδρομή και τα ίδια εφτά με ορισμένο χρόνο μέγιστης παραμονής στον πελάτη. Ο αριθμός κόμβων στα προβλήματα κυμαίνεται από 51 μέχρι και 200 κόμβους. Αρχικά, στη στήλη N παρουσιάζεται ο αριθμός των πελατών, στη στήλη Q παρουσιάζεται η μέγιστη χωρητικότητα των οχημάτων, η στήλη Max Duration αναφέρεται στη μέγιστη χρονική διάρκεια παραμονής του οχήματος σε μία διαδρομή, η στήλη Service Time αναπαριστά το χρόνο εξυπηρέτηση για κάθε πελάτη και τέλος η στήλη Best αναπαριστά τη βέλτιστη λύση.

| Instance | N | Q | Max Duration | Service Time | Best |
|---|---|---|---|---|---|
| **CMT1** | 50 | 160 | ∞ | 0 | 524.61 |
| **CMT2** | 75 | 140 | ∞ | 0 | 835.26 |
| **CMT3** | 100 | 200 | ∞ | 0 | 826.14 |
| **CMT4** | 150 | 200 | ∞ | 0 | 1028.42 |
| **CMT5** | 199 | 200 | ∞ | 0 | 1291.29 |
| **CMT6** | 50 | 160 | 200 | 10 | 555.43 |
| **CMT7** | 75 | 140 | 160 | 10 | 909.68 |
| **CMT8** | 150 | 200 | 230 | 10 | 865.94 |
| **CMT9** | 150 | 200 | 200 | 10 | 1162.55 |
| **CMT10** | 199 | 200 | 200 | 10 | 1395.85 |
| **CMT11** | 120 | 200 | ∞ | 0 | 1042.11 |
| **CMT12** | 100 | 200 | ∞ | 0 | 819.56 |
| **CMT13** | 120 | 200 | 720 | 50 | 1541.14 |
| **CMT14** | 100 | 200 | 1040 | 90 | 866.37 |

**Πίνακας:** Δεδομένα Περιπτώσεων Christofides, Mingozzi and Toth

## 6.2 Υπολογιστικά Αποτελέσματα

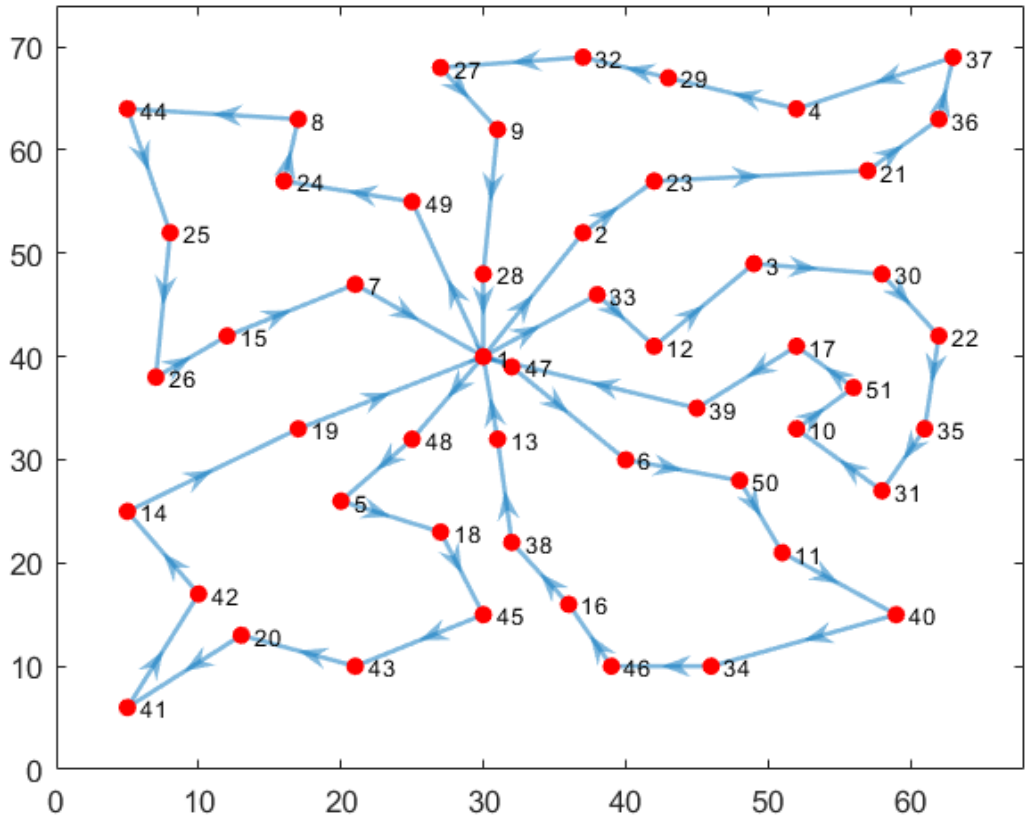
| Iter = 50 | Particle Swarm | | Grey Wolf | | Monarch Butterfly | | Whale | | Moth Flame | |
|---|---|---|---|---|---|---|---|---|---|---|
| **Instance** | **Avg** | **Best** | **Avg** | **Best** | **Avg** | **Best** | **Avg** | **Best** | **Avg** | **Best** |
| **CMT1** | 530.25 | **525.88** | 561.51 | 528.73 | 563.35 | 547.13 | 566.06 | 559.43 | 595.33 | 583.51 |
| **CMT2** | 872.41 | 864.88 | 874.66 | **861.85** | 888.02 | 878.25 | 909.15 | 886.27 | 913.73 | 896.20 |
| **CMT3** | 944.85 | 906.46 | 910.78 | **901.71** | 1014.6 | 999.51 | 922.45 | 907.06 | 951.87 | 942.66 |
| **CMT4** | 1259.4 | 1198.3 | 1132.2 | **1107.2** | 1180.4 | 1171.7 | 1203.7 | 1199.5 | 1204.9 | 1200.8 |
| **CMT5** | 1442.1 | 1402.4 | 1412.1 | **1387.8** | 1396.4 | 1388.2 | 1484.9 | 1465.0 | 1510.3 | 1495.9 |
| **CMT6** | 599.05 | 588.96 | 603.17 | **588.48** | 664.57 | 645.32 | 656.35 | 631.37 | 674.73 | 661.07 |
| **CMT7** | 982.10 | 976.09 | 971.98 | 961.40 | 1029.3 | 994.04 | 978.10 | **949.33** | 1076.3 | 1058.1 |
| **CMT8** | 1083.2 | 1001.3 | 1062.1 | **983.20** | 1154.9 | 1113.5 | 1299.8 | 1247.8 | 1229.5 | 1205.7 |
| **CMT9** | 1294.7 | **1238.2** | 1402.4 | 1366.1 | 1317.9 | 1276.8 | 1495.4 | 1471.2 | 1395.0 | 1312.9 |
| **CMT10** | 1698.1 | 1657.8 | 1648.0 | **1628.1** | 1765.6 | 1697.8 | 1680.2 | 1667.8 | 1708.6 | 1697.6 |
| **CMT11** | 1113.7 | 1112.6 | 1076.2 | **1070.3** | 1179.4 | 1165.0 | 1113.8 | 1112.2 | 1223.6 | 1163.9 |
| **CMT12** | 1035.62 | 1021.50 | 997.44 | 981.21 | 1050.0 | 1041.9 | 1083.4 | 1046.2 | 989.81 | **957.91** |
| **CMT13** | 1794.2 | 1782.4 | 1789.7 | 1773.8 | 1722.7 | **1702.3** | 1808.4 | 1781.8 | 1763.9 | 1722.1 |
| **CMT14** | 1082.1 | 1049.4 | 1077.1 | **1028.1** | 1241.3 | 1236.8 | 1191.5 | 1182.7 | 1117.2 | 1088.1 |

**Πίνακας:** Αποτελέσματα Αλγορίθμων για 50 Επαναλήψεις

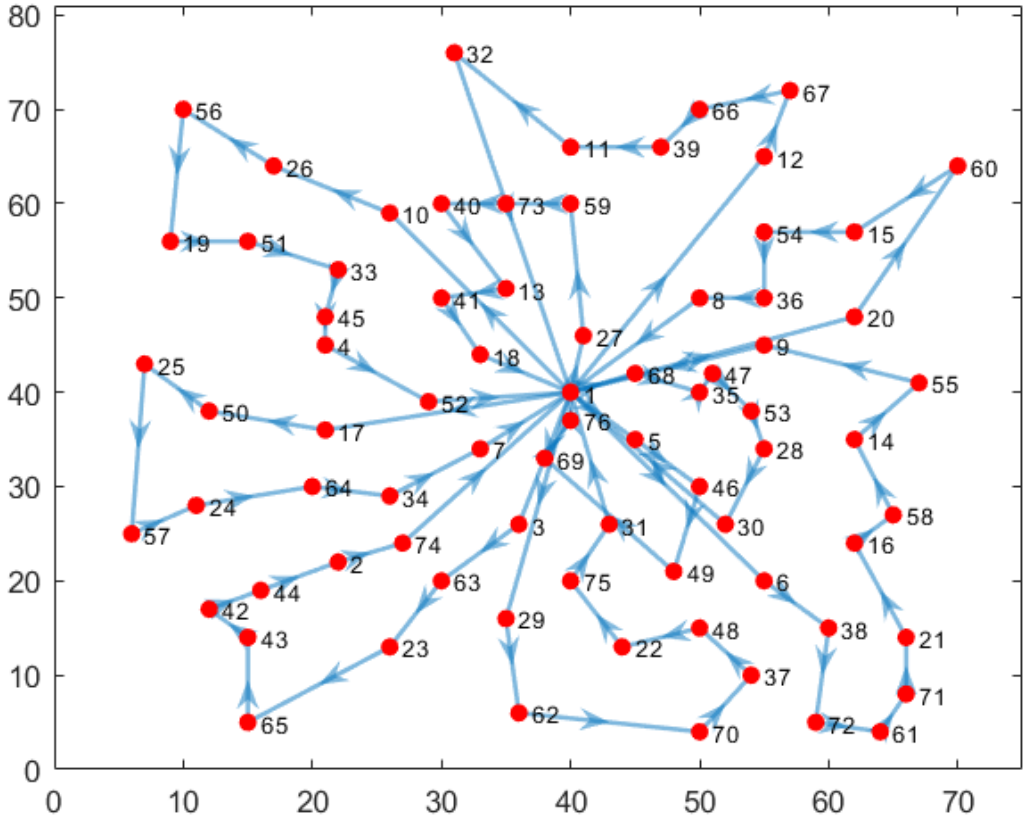
| Iter = 100 | Particle Swarm | | Grey Wolf | | Monarch Butterfly | | Whale | | Moth Flame | |
|---|---|---|---|---|---|---|---|---|---|---|
| **Instance** | **Avg** | **Best** | **Avg** | **Best** | **Avg** | **Best** | **Avg** | **Best** | **Avg** | **Best** |
| **CMT1** | 526.14 | **524.91** | 536.55 | 525.11 | 559.42 | 530.13 | 564.10 | 561.19 | 591.14 | 587.38 |
| **CMT2** | 861.41 | 857.55 | 861.85 | **850.18** | 886.13 | 851.81 | 876.16 | 853.51 | 886.89 | 856.90 |
| **CMT3** | 899.11 | 872.58 | 870.78 | **851.44** | 985.77 | 939.94 | 916.17 | 900.94 | 939.55 | 902.86 |
| **CMT4** | 1178.4 | 1128.6 | 1128.4 | **1094.2** | 1164.3 | 1151.2 | 1171.3 | 1162.1 | 1198.4 | 1188.1 |
| **CMT5** | 1412.7 | 1391.0 | 1387.6 | 1345.2 | 1337.3 | **1336.1** | 1467.3 | 1441.1 | 1457.2 | 1412.1 |
| **CMT6** | 591.13 | 575.31 | 582.27 | **568.22** | 655.55 | 624.21 | 621.74 | 601.45 | 645.63 | 609.80 |
| **CMT7** | 933.16 | **912.40** | 954.20 | 940.49 | 987.71 | 956.61 | 973.32 | 950.29 | 1012.6 | 1001.9 |
| **CMT8** | 965.33 | **916.36** | 966.22 | 948.24 | 1127.2 | 1088.1 | 1220.1 | 1185.2 | 1207.5 | 1191.3 |
| **CMT9** | 1276.8 | **1258.1** | 1352.5 | 1336.1 | 1311.7 | 1309.9 | 1426.2 | 1384.8 | 1355.2 | 1328.8 |
| **CMT10** | 1645.2 | 1596.1 | 1599.7 | **1583.6** | 1687.2 | 1633.4 | 1676.0 | 1661.8 | 1701.7 | 1677.9 |
| **CMT11** | 1107.8 | 1097.8 | 1067.6 | **1057.6** | 1113.3 | 1072.5 | 1107.9 | 1103.1 | 1174.2 | 1144.2 |
| **CMT12** | 989.13 | 987.81 | 966.63 | **909.42** | 1032.4 | 920.28 | 993.51 | 974.61 | 925.10 | 915.34 |
| **CMT13** | 1728.3 | 1698.6 | 1734.1 | 1711.3 | 1709.4 | **1671.2** | 1789.3 | 1740.3 | 1737.4 | 1698.7 |
| **CMT14** | 1039.3 | **974.67** | 1054.8 | 1030.6 | 1189.0 | 1174,7 | 1163.9 | 1023.3 | 1079.8 | 1000.4 |

**Πίνακας:** Αποτελέσματα Αλγορίθμων για 100 Επαναλήψεις

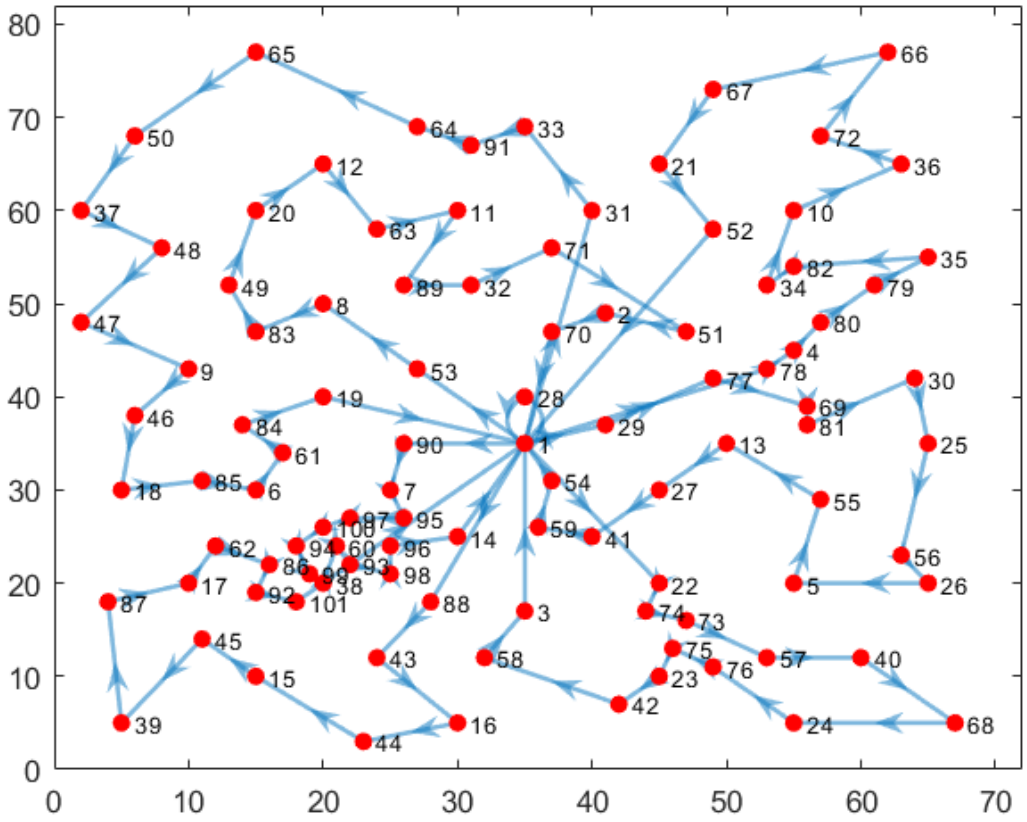
## 6.3 Παρουσίαση Καλύτερων Διαδρομών



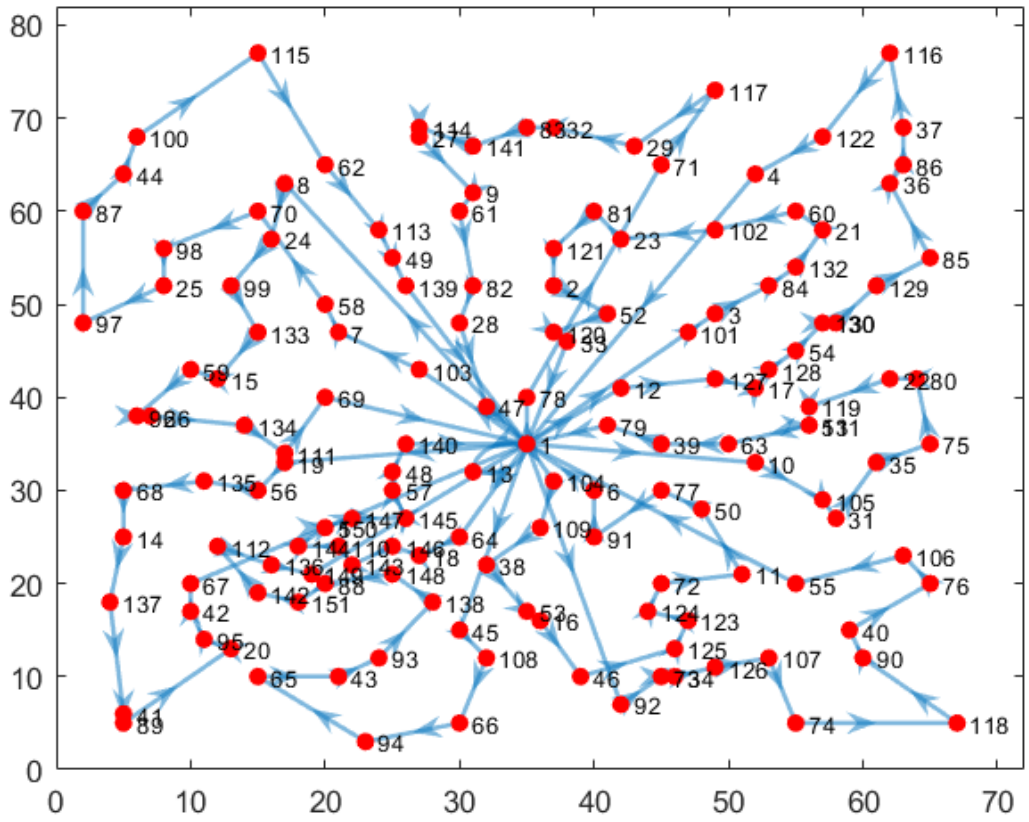

| CMT1 | |
|---|---|
| **Route 1** | 1 2 23 21 36 37 4 29 32 27 9 28 |
| **Route 2** | 1 33 12 3 30 22 35 31 10 51 17 39 |
| **Route 3** | 1 47 6 50 11 40 34 46 16 38 13 |
| **Route 4** | 1 48 5 18 45 43 20 41 42 14 19 |
| **Route 5** | 1 49 24 8 44 25 26 15 7 1 |
| **Distance:** 524.91 | |



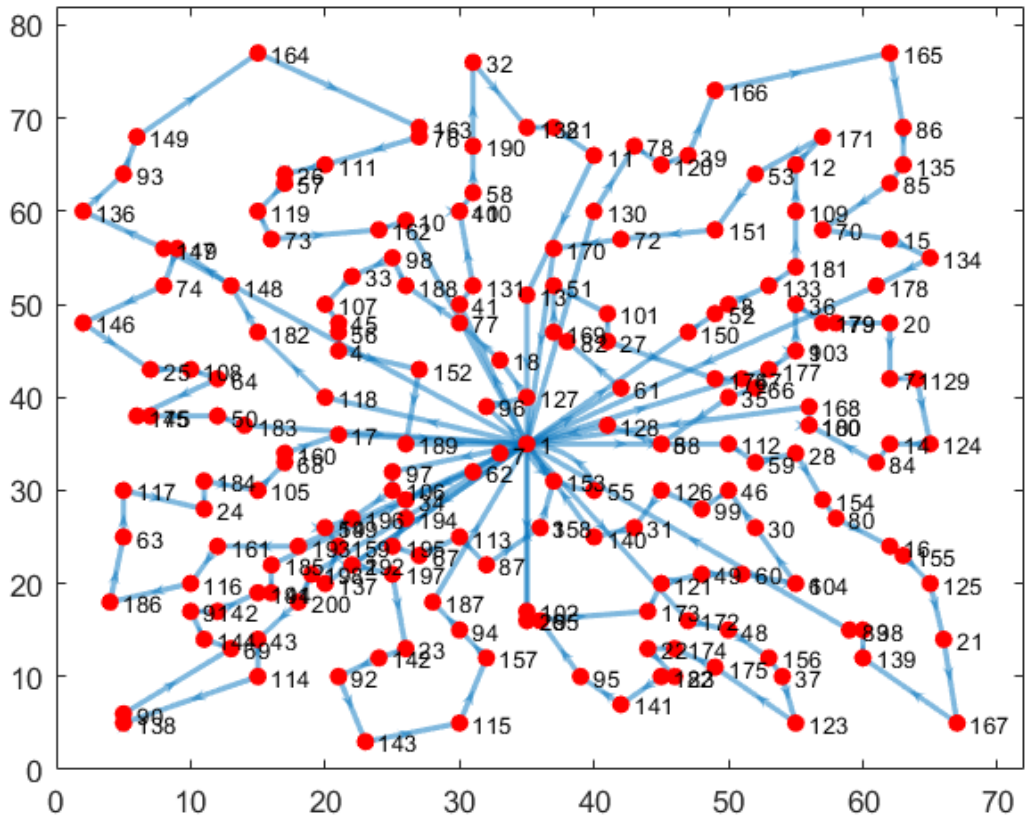

| CMT2 | |
|---|---|
| **Route 1** | 1 68 35 47 53 28 30 |
| **Route 2** | 1 6 38 72 61 71 21 16 58 14 55 9 |
| **Route 3** | 1 20 60 15 54 36 8 |
| **Route 4** | 1 12 67 66 39 11 32 |
| **Route 5** | 1 27 59 73 40 13 41 18 |
| **Route 6** | 1 10 26 56 19 51 33 45 4 52 |
| **Route 7** | 1 17 50 25 57 24 64 34 7 |
| **Route 8** | 1 3 63 23 65 43 42 44 2 74 |
| **Route 9** | 1 29 62 70 37 48 22 75 31 |
| **Route 10** | 1 5 46 49 69 76 1 |
| **Distance:** 850.19 | |



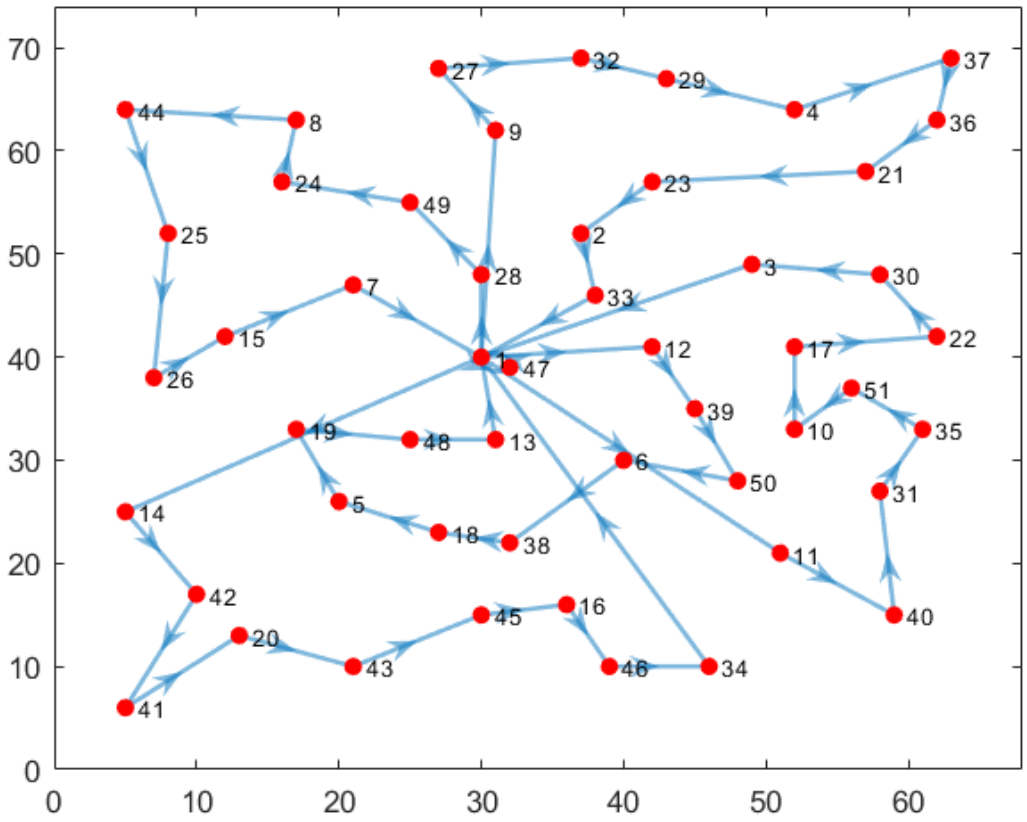

| CMT3 | |
|---|---|
| **Route 1** | 1 90 7 95 97 100 94 99 38 60 93 98 96 14 |
| **Route 2** | 1 88 43 16 44 15 45 39 87 17 62 86 92 101 |
| **Route 3** | 1 22 74 73 57 40 68 24 76 75 23 42 58 3 |
| **Route 4** | 1 29 77 69 81 30 25 56 26 5 55 13 27 41 59 54 |
| **Route 5** | 1 78 4 80 79 35 82 34 10 36 72 66 67 21 52 |
| **Route 6** | 1 31 33 91 64 65 50 37 48 47 9 46 18 85 6 61 84 19 |
| **Route 7** | 1 53 8 83 49 20 12 63 11 89 32 71 51 2 70 |
| **Route 8** | 1 28 1 |
| **Distance:** 851.44 | |



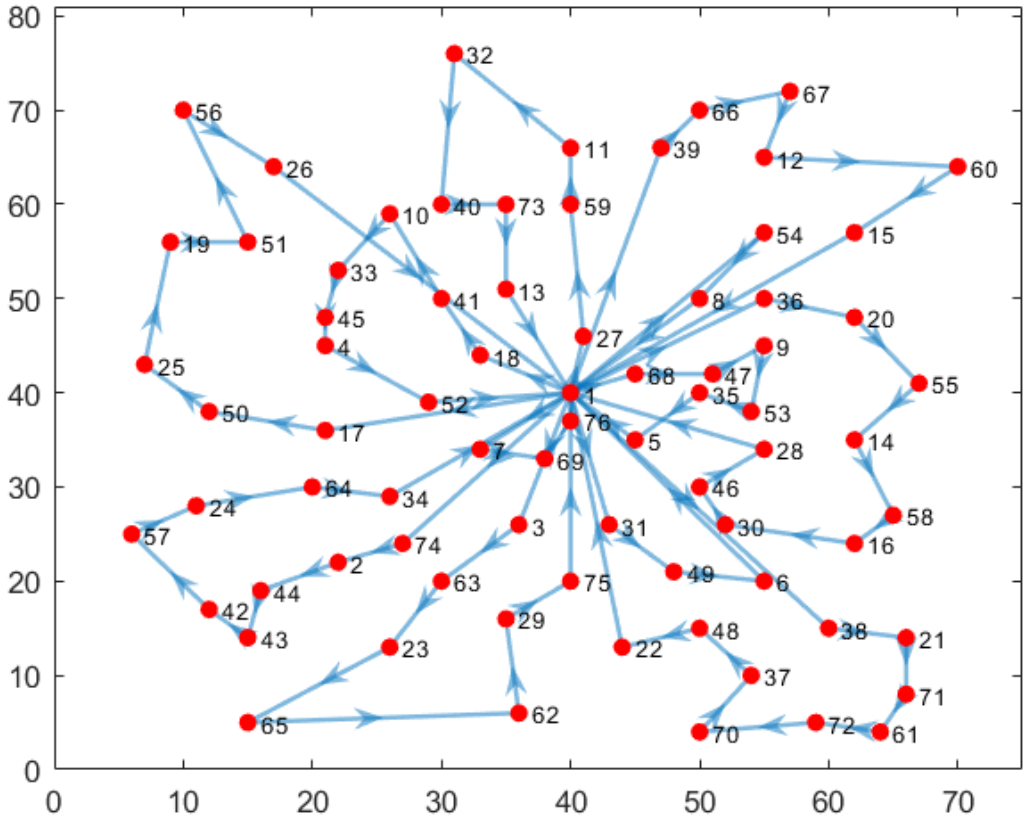
| CMT4 | |
|---|---|
| **Route 1** | 1 104 109 38 53 16 46 125 123 124 72 11 50 77 91 6 |
| **Route 2** | 1 92 73 34 126 107 74 118 90 40 76 106 55 |
| **Route 3** | 1 10 105 31 35 75 80 22 119 51 131 63 39 79 |
| **Route 4** | 1 12 127 17 128 54 130 30 129 85 36 86 37 116 122 4 |
| **Route 5** | 1 101 3 84 132 21 60 102 23 81 121 2 52 120 33 |
| **Route 6** | 1 78 71 117 29 32 83 141 114 27 9 61 82 28 47 |
| **Route 7** | 1 103 7 58 70 98 25 97 87 44 100 115 62 113 49 139 |
| **Route 8** | 1 8 24 99 133 15 59 96 26 134 111 69 |
| **Route 9** | 1 19 56 135 68 14 137 41 89 20 95 42 67 |
| **Route 10** | 1 45 108 66 94 65 43 93 138 148 88 151 142 112 136 149 |
| **Route 11** | 1 64 18 146 143 110 144 150 5 147 145 57 48 140 |
| **Route 12** | 1 13 1 |
| **Distance:** 1094.2 | |



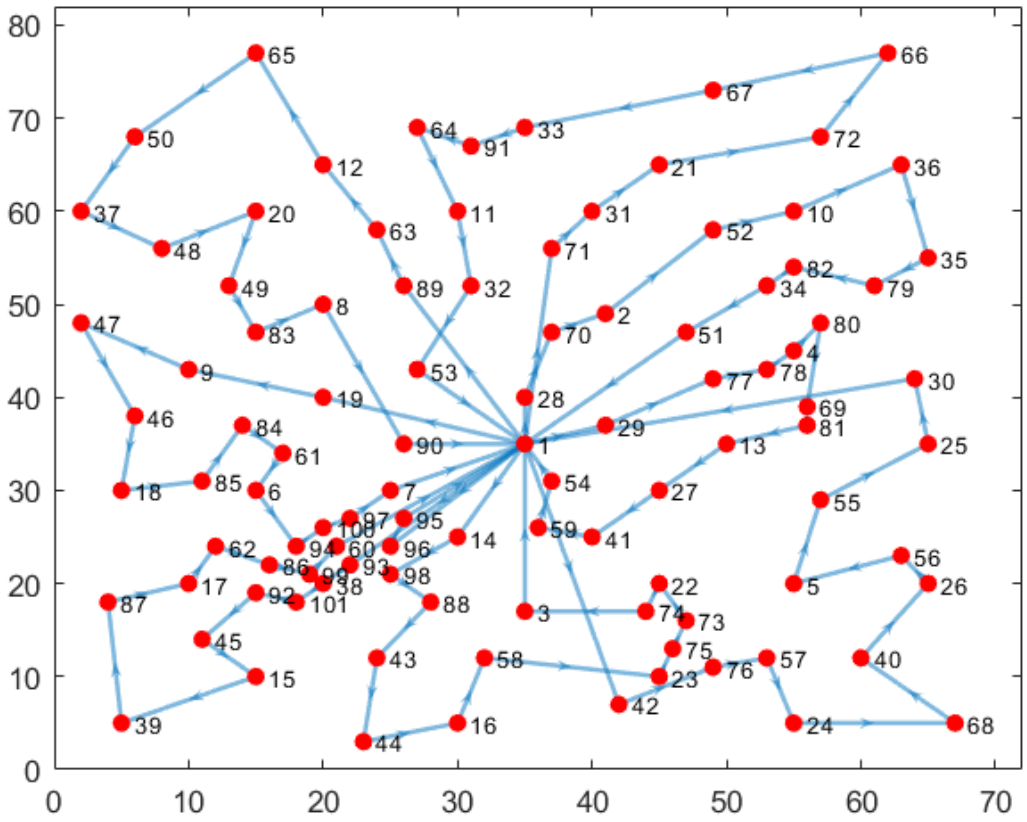
| CMT5 | |
|---|---|
| **Route 1** | 1 96 127 18 77 188 98 33 107 45 56 4 152 189 |
| **Route 2** | 1 97 106 196 54 199 159 137 192 2 197 |
| **Route 3** | 1 23 142 92 143 115 157 94 187 |
| **Route 4** | 1 198 200 43 114 138 90 69 144 91 42 191 44 185 |
| **Route 5** | 1 118 182 148 19 74 146 25 108 64 145 75 50 183 |
| **Route 6** | 1 147 136 93 149 164 163 76 111 26 57 119 73 162 10 |
| **Route 7** | 1 41 131 110 40 58 190 32 132 81 11 13 |
| **Route 8** | 1 150 52 8 133 181 109 12 171 53 151 72 170 |
| **Route 9** | 1 177 9 103 36 179 79 20 71 129 124 14 84 180 100 168 |
| **Route 10** | 1 112 59 28 154 80 16 155 125 21 167 139 38 89 |
| **Route 11** | 1 172 48 156 37 123 175 174 22 83 122 141 95 65 102 |
| **Route 12** | 1 29 173 121 49 60 6 104 30 46 99 126 31 140 |
| **Route 13** | 1 7 62 34 194 195 67 113 87 158 3 153 |
| **Route 14** | 1 55 1 |
| **Distance:** 1336.1 | |



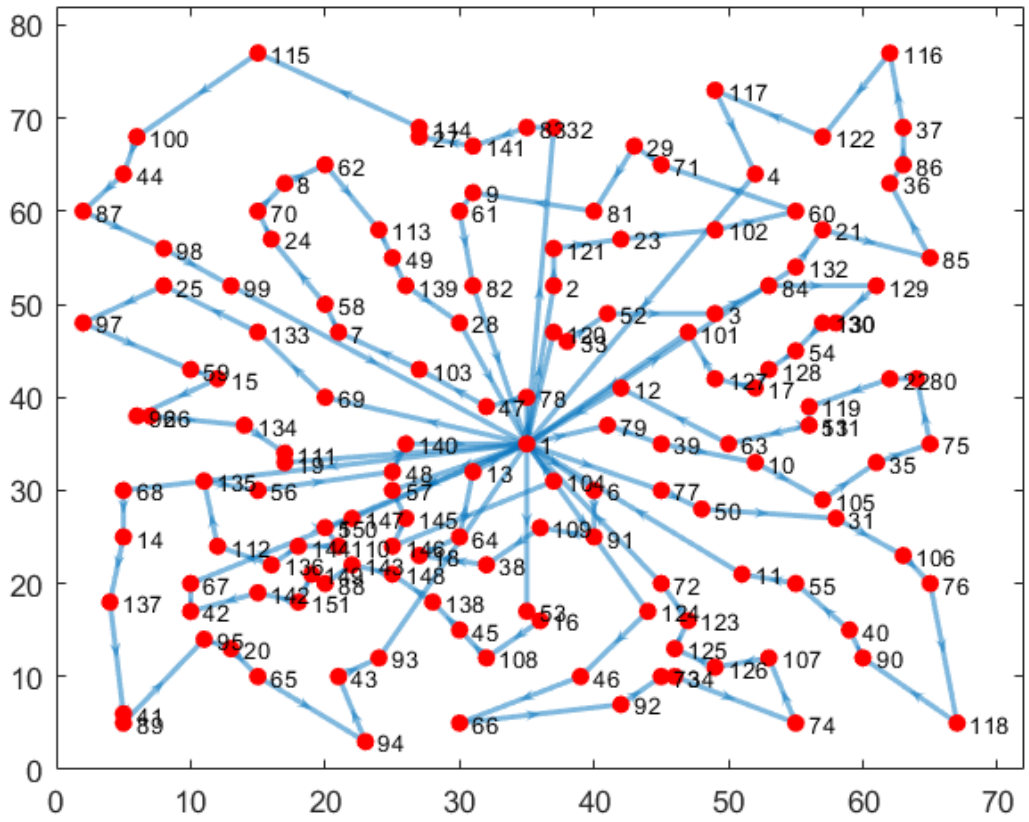

| CMT6 | |
|---|---|
| **Route 1** | 1 28 49 24 8 44 25 26 15 7 |
| **Route 2** | 1 14 42 41 20 43 45 16 46 34 |
| **Route 3** | 1 11 40 31 35 51 10 17 22 30 3 |
| **Route 4** | 1 9 27 32 29 4 37 36 21 23 2 33 |
| **Route 5** | 1 12 39 50 6 38 18 5 19 48 13 |
| **Route 6** | 1 47 1 |
| **Distance:** 568.22 | |



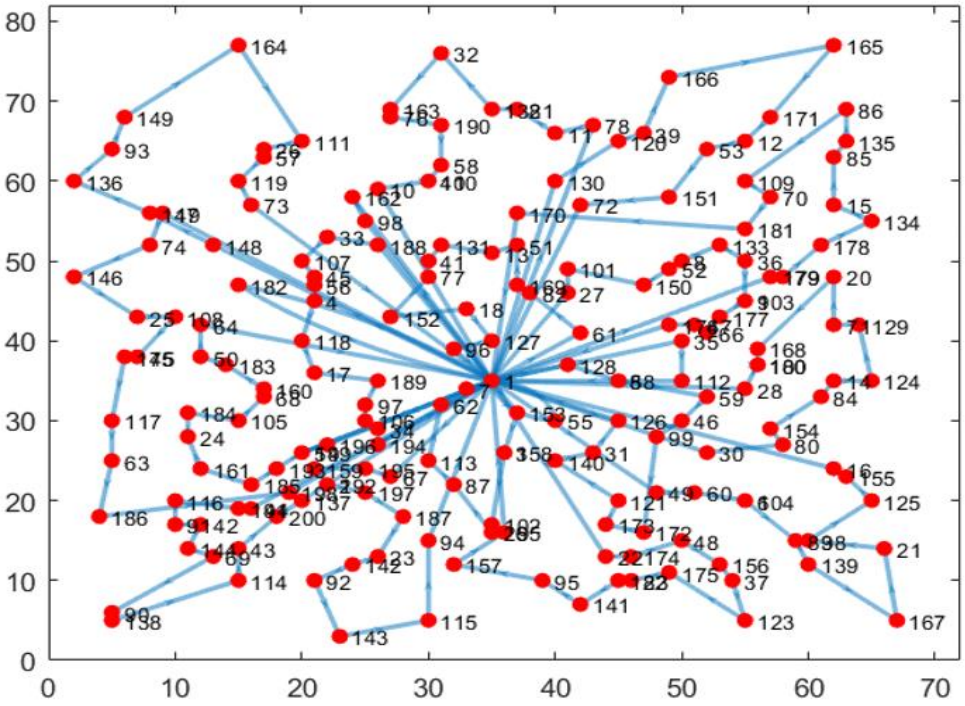

| CMT7 | |
|---|---|
| **Route 1** | 1 27 59 11 32 40 73 13 |
| **Route 2** | 1 39 66 67 12 60 15 |
| **Route 3** | 1 54 8 |
| **Route 4** | 1 36 20 55 14 58 16 30 46 28 |
| **Route 5** | 1 68 47 9 53 35 5 |
| **Route 6** | 1 31 49 6 1 38 21 71 61 72 70 37 48 22 |
| **Route 7** | 1 3 63 23 65 62 29 75 |
| **Route 8** | 1 74 2 44 43 42 57 24 64 34 |
| **Route 9** | 1 17 50 25 19 51 56 26 |
| **Route 10** | 1 18 41 10 33 45 4 52 |
| **Route 11** | 1 7 69 76 1 |
| **Distance:** 912.40 | |



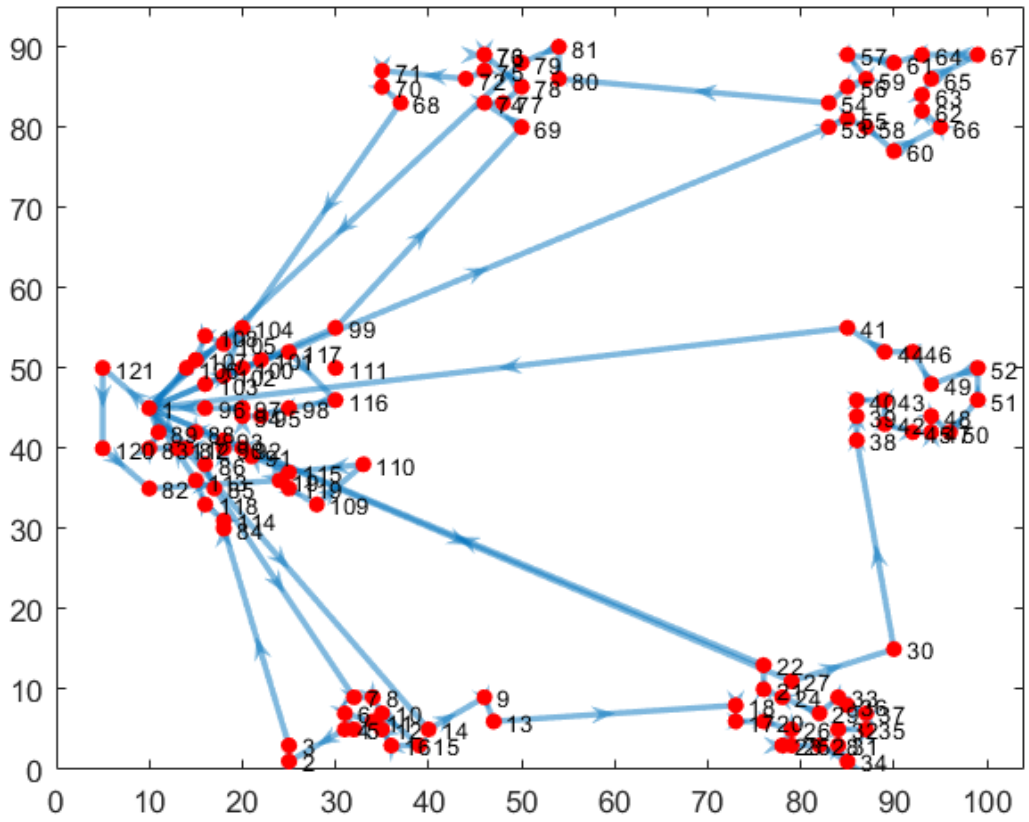

| CMT8 | |
|---|---|
| **Route 1** | 1 28 70 2 52 10 36 35 79 82 34 51 |
| **Route 2** | 1 71 31 21 72 66 67 33 91 64 11 32 53 |
| **Route 3** | 1 89 63 12 65 50 37 48 20 49 83 8 90 |
| **Route 4** | 1 19 9 47 46 18 85 84 61 6 94 100 97 7 |
| **Route 5** | 1 93 38 101 92 45 15 39 87 17 62 86 99 60 |
| **Route 6** | 1 96 95 |
| **Route 7** | 1 14 98 88 43 44 16 58 23 75 73 22 74 3 |
| **Route 8** | 1 42 76 57 24 68 40 26 56 5 55 25 30 |
| **Route 9** | 1 29 77 78 4 80 69 81 13 27 41 59 54 1 |
| **Distance:** 916.36 | |



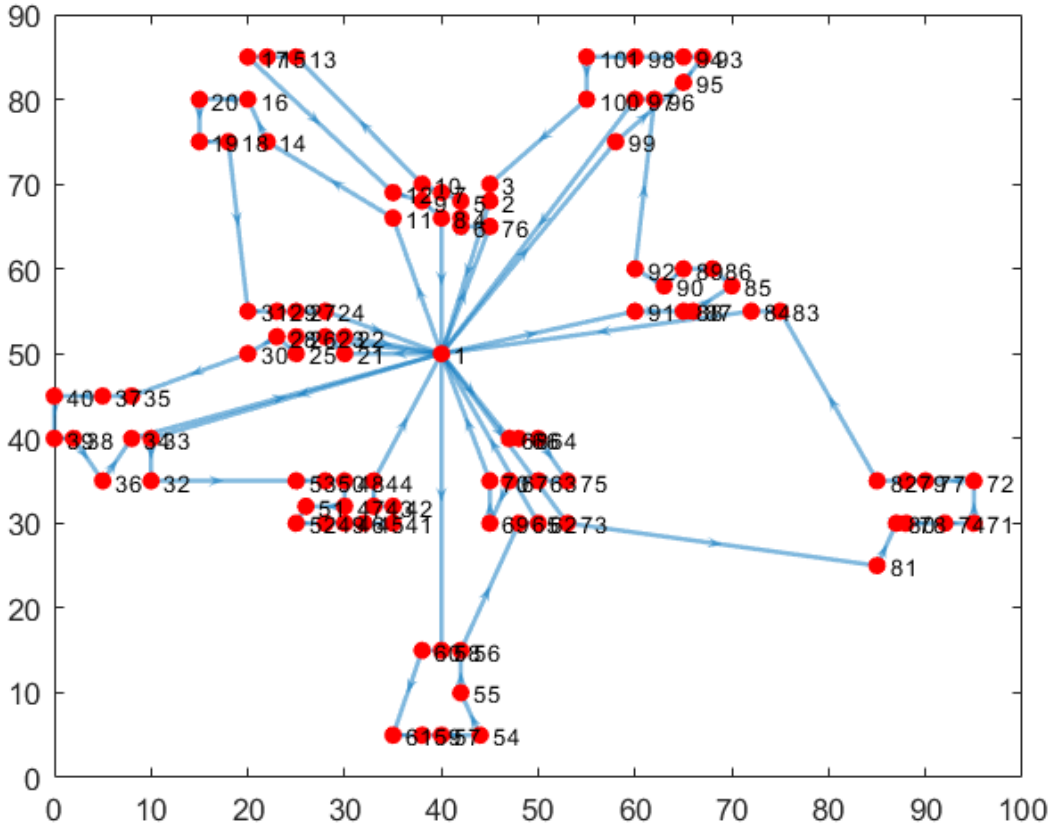
| CMT9 | |
|---|---|
| **Route 1** | 1 77 50 31 106 76 118 90 40 55 11 |
| **Route 2** | 1 79 39 10 105 35 75 80 22 119 131 51 63 12 |
| **Route 3** | 1 78 47 103 7 58 24 70 8 62 113 49 139 28 |
| **Route 4** | 1 120 33 52 3 84 129 30 130 54 128 17 127 101 |
| **Route 5** | 1 132 21 85 36 86 37 116 122 117 4 |
| **Route 6** | 1 2 121 23 102 60 71 29 81 9 61 82 |
| **Route 7** | 1 32 83 141 27 114 115 100 44 87 98 99 |
| **Route 8** | 1 69 133 25 97 59 15 96 26 134 111 19 |
| **Route 9** | 1 68 14 137 41 89 95 20 65 94 43 93 |
| **Route 10** | 1 124 46 66 92 73 34 74 107 126 125 123 72 |
| **Route 11** | 1 53 16 108 45 138 148 143 88 149 151 142 42 67 147 |
| **Route 12** | 1 150 5 110 144 136 112 135 56 48 140 |
| **Route 13** | 1 57 145 146 104 |
| **Route 14** | 1 6 91 109 38 18 64 13 1 |
| **Distance:** 1238.2 | |



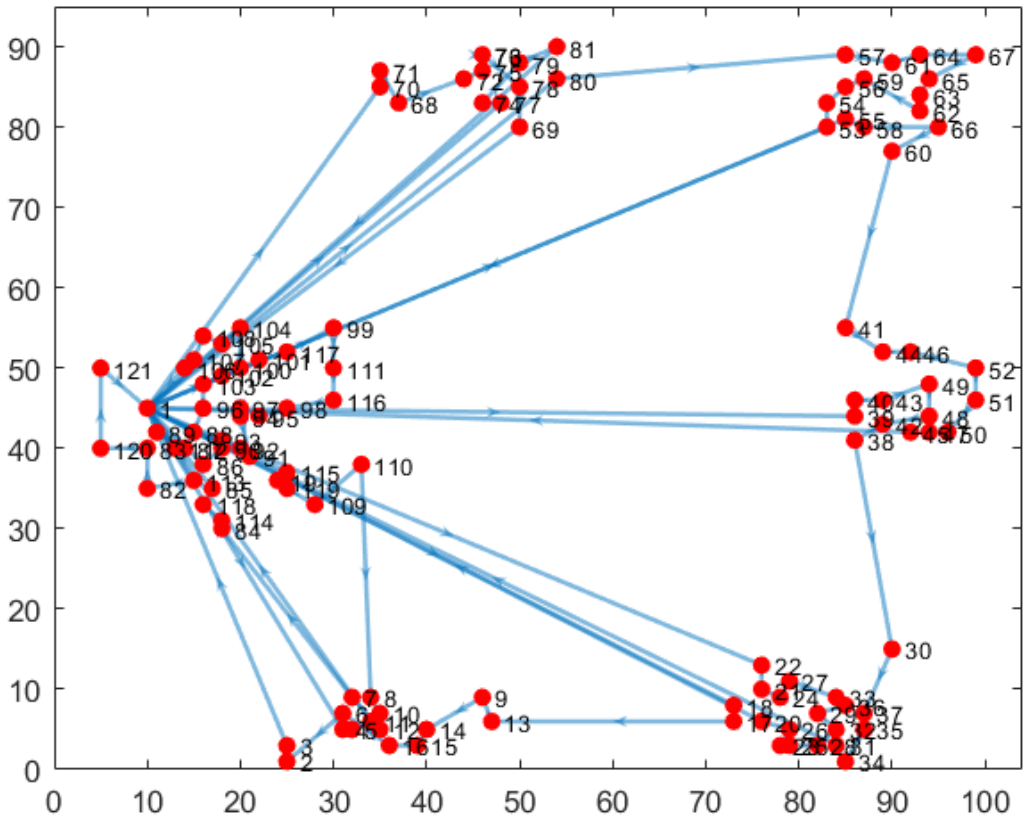
| CMT10 | |
|---|---|
| **Route 1** | 1 61 82 169 51 13 131 41 77 152 18 127 96 |
| **Route 2** | 1 27 101 150 52 8 133 36 103 9 177 66 47 176 |
| **Route 3** | 1 35 112 |
| **Route 4** | 1 128 5 88 59 4 99 172 173 121 140 31 126 51 |
| **Route 5** | 1 49 60 6 104 89 139 167 21 38 125 155 16 |
| **Route 6** | 1 30 80 154 84 14 124 129 71 20 168 180 100 28 |
| **Route 7** | 1 179 79 178 134 15 85 135 86 109 70 181 170 |
| **Route 8** | 1 130 120 39 166 165 171 12 53 151 72 |
| **Route 9** | 1 78 11 81 132 32 163 76 190 58 110 40 10 |
| **Route 10** | 1 162 98 |
| **Route 11** | 1 188 33 107 45 56 4 118 17 189 97 106 34 194 |
| **Route 12** | 1 196 199 |
| **Route 13** | 1 54 193 185 161 24 184 105 68 160 183 50 64 182 |
| **Route 14** | 1 148 147 136 93 149 164 111 26 57 119 73 |
| **Route 15** | 1 19 74 146 25 208 75 117 63 186 198 |
| **Route 16** | 1 44 191 116 91 42 144 69 90 138 114 43 200 137 |
| **Route 17** | 1 67 195 192 2 197 187 23 142 92 143 115 94 |
| **Route 18** | 1 22 174 48 156 37 123 175 83 122 141 95 157 65 |
| **Route 19** | 1 153 3 158 102 29 87 113 62 7 1 |
| **Distance:** 1583.6 | |



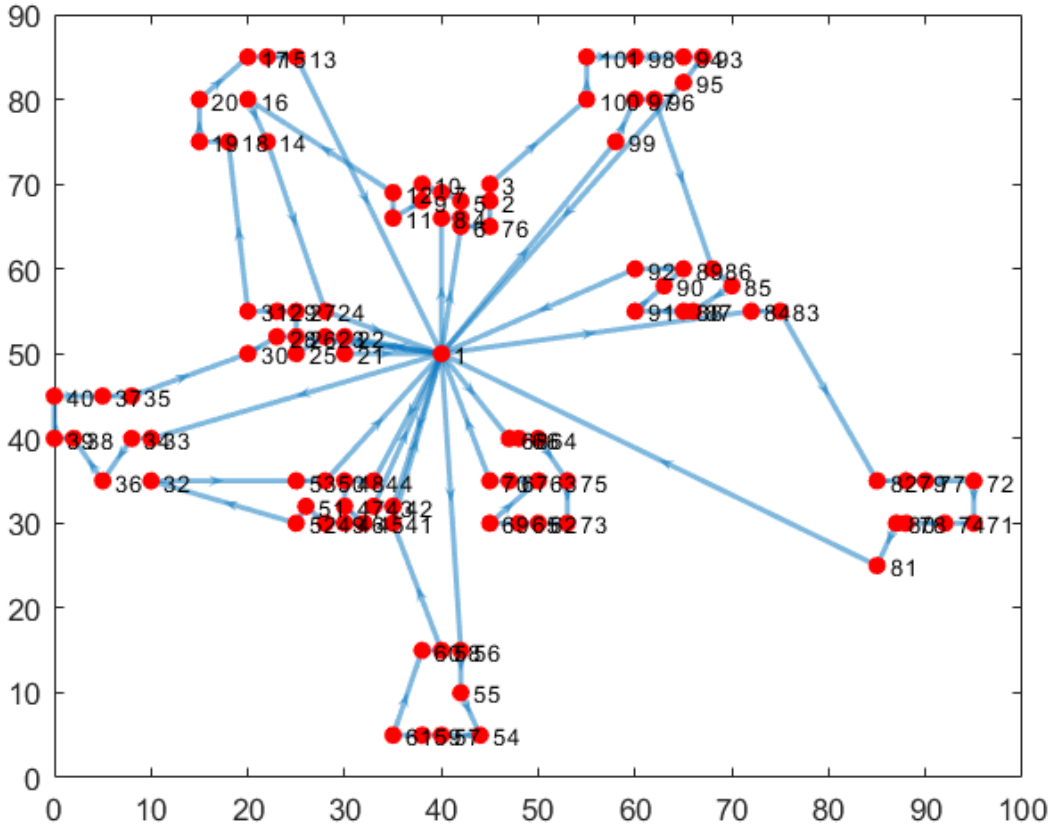
| CMT11 | |
|---|---|
| **Route 1** | 1 121 120 82 19 119 109 110 115 91 92 90 93 88 87 112 83 89 |
| **Route 2** | 1 86 85 7 8 10 15 16 12 11 6 5 4 2 3 84 114 118 113 |
| **Route 3** | 1 14 9 13 18 17 20 26 23 25 28 34 31 32 35 37 36 33 29 24 21 22 |
| **Route 4** | 1 27 30 38 39 40 43 42 45 47 48 50 51 52 49 46 44 41 |
| **Route 5** | 1 53 55 58 60 66 62 63 65 67 64 61 57 59 56 54 80 81 79 |
| **Route 6** | 1 99 69 74 77 78 76 73 75 72 71 70 68 104 105 108 107 |
| **Route 7** | 1 106 104 102 100 101 117 116 98 95 94 97 96 1 |
| **Distance:** 1057.6 | |



| CMT12 | |
|---|---|
| **Route 1** | 1 68 66 64 75 63 67 69 70 |
| **Route 2** | 1 58 60 61 59 57 54 55 56 65 62 |
| **Route 3** | 1 73 81 80 78 74 71 72 77 79 82 83 84 |
| **Route 4** | 1 91 88 87 85 86 89 90 92 96 97 |
| **Route 5** | 1 99 95 93 94 98 101 100 3 2 |
| **Route 6** | 1 76 6 4 5 7 10 13 15 17 12 9 8 |
| **Route 7** | 1 11 14 16 20 19 18 31 29 27 24 |
| **Route 8** | 1 26 25 28 30 35 37 40 39 38 36 34 |
| **Route 9** | 1 33 32 53 50 48 47 51 52 49 46 45 41 42 43 44 |
| **Route 10** | 1 21 23 22 1 |
| **Distance:** 909.42 | |



| CMT13 | |
|---|---|
| **Route 1** | 1 89 112 87 88 96 103 102 100 104 105 108 107 106 |
| **Route 2** | 1 70 71 68 72 75 73 76 78 69 |
| **Route 3** | 1 77 74 79 81 |
| **Route 4** | 1 80 57 61 64 67 65 63 62 59 56 |
| **Route 5** | 1 55 58 66 60 41 44 46 52 51 50 |
| **Route 6** | 1 39 40 43 49 48 42 38 30 37 35 |
| **Route 7** | 1 22 21 24 27 33 36 29 32 28 25 |
| **Route 8** | 1 18 17 13 9 14 15 16 6 2 3 |
| **Route 9** | 1 4 5 11 |
| **Route 10** | 1 115 19 119 109 110 8 10 12 7 84 114 118 85 |
| **Route 11** | 1 101 117 99 111 116 98 |
| **Route 12** | 1 113 82 83 120 121 1 |
| **Distance:** 1671.2 | |



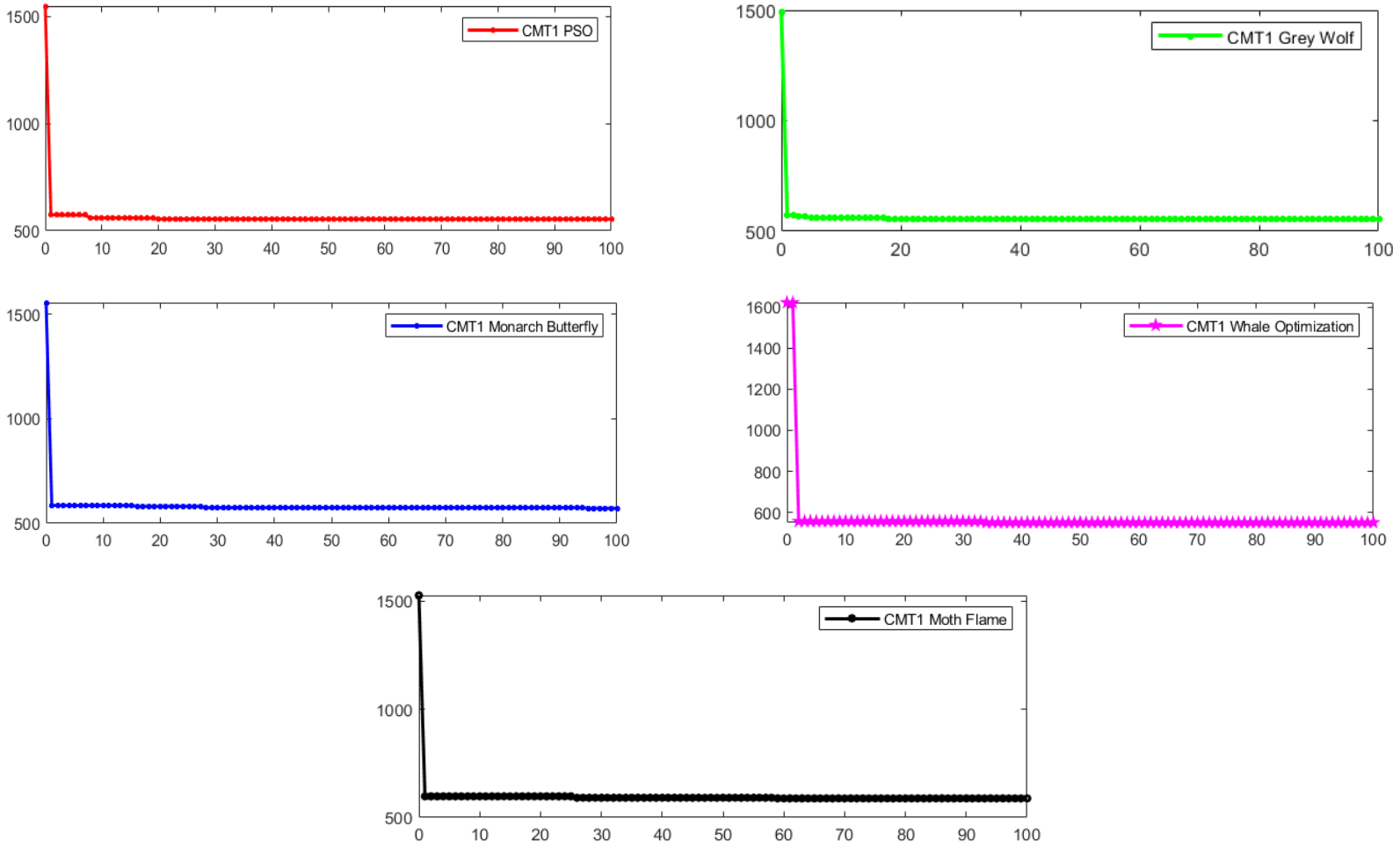

| CMT13 | |
|---|---|
| **Route 1** | 1 6 76 2 3 100 101 98 94 93 95 |
| **Route 2** | 1 99 97 96 86 85 87 88 91 90 89 92 |
| **Route 3** | 1 84 83 82 79 77 72 71 74 78 80 81 |
| **Route 4** | 1 68 66 64 75 73 62 65 69 63 67 70 |
| **Route 5** | 1 56 55 54 57 59 61 60 58 41 |
| **Route 6** | 1 42 43 |
| **Route 7** | 1 44 48 47 45 46 49 51 52 32 53 50 |
| **Route 8** | 1 33 34 36 38 39 40 37 35 30 28 26 |
| **Route 9** | 1 23 25 27 29 31 18 19 20 17 15 13 |
| **Route 10** | 1 8 4 5 7 10 9 11 12 16 14 24 |
| **Route 11** | 1 22 21 1 |
| **Distance:** 974.67 | |

## 6.4 Συγκριτική Ανάλυση Αλγορίθμων

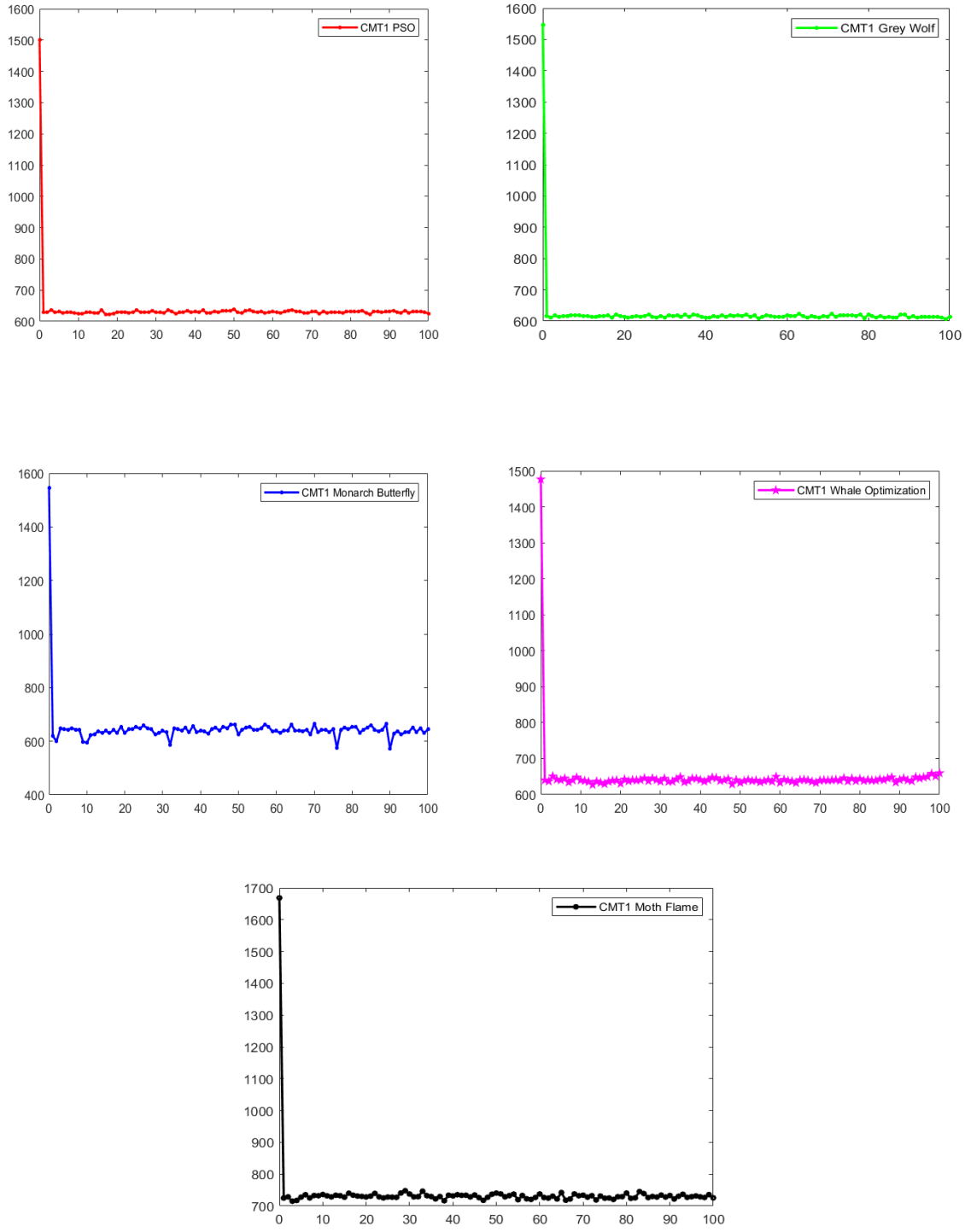
| Instance | Lower Bound | Best | Average | Algorithm |
|---|---|---|---|---|
| **CMT1** | 524.61 | 524.91 | 526.14 | Particle Swarm |
| **CMT2** | 835.26 | 850.18 | 861.85 | Grey Wolf |
| **CMT3** | 826.14 | 851.44 | 870.78 | Grey Wolf |
| **CMT4** | 1028.42 | 1094.2 | 1128.4 | Grey Wolf |
| **CMT5** | 1291.29 | 1336.1 | 1337.3 | Monarch Butterfly |
| **CMT6** | 555.43 | 568.22 | 582.27 | Grey Wolf |
| **CMT7** | 909.68 | 912.40 | 933.16 | Particle Swarm |
| **CMT8** | 865.94 | 916.36 | 965.33 | Particle Swarm |
| **CMT9** | 1162.55 | 1238.2 | 1294.7 | Particle Swarm |
| **CMT10** | 1395.85 | 1583.6 | 1599.7 | Grey Wolf |
| **CMT11** | 1042.11 | 1057.6 | 1067.6 | Grey Wolf |
| **CMT12** | 819.56 | 909.42 | 966.63 | Grey Wolf |
| **CMT13** | 1541.14 | 1671.2 | 1709.4 | Monarch Butterfly |
| **CMT14** | 866.37 | 974.67 | 1039.3 | Particle Swarm |

| Instance | Best Gap(%) | Average Gap(%) | Algorithm |
|---|---|---|---|
| **CMT1** | 0.06% | 0.29% | Particle Swarm |
| **CMT2** | 1.79% | 3.18% | Grey Wolf |
| **CMT3** | 3.06% | 5.40% | Grey Wolf |
| **CMT4** | 6.40% | 9.72% | Grey Wolf |
| **CMT5** | 3.47% | 3.56% | Monarch Butterfly |
| **CMT6** | 2.30% | 4.83% | Grey Wolf |
| **CMT7** | 0.30% | 2.58% | Particle Swarm |
| **CMT8** | 5.82% | 11.48% | Particle Swarm |
| **CMT9** | 6.51% | 11.37% | Particle Swarm |
| **CMT10** | 15.45% | 24.60% | Grey Wolf |
| **CMT11** | 1.49% | 2.45% | Grey Wolf |
| **CMT12** | 10.96% | 17.94% | Grey Wolf |
| **CMT13** | 8.44% | 11.92% | Monarch Butterfly |
| **CMT14** | 13.50% | 17.96% | Particle Swarm |

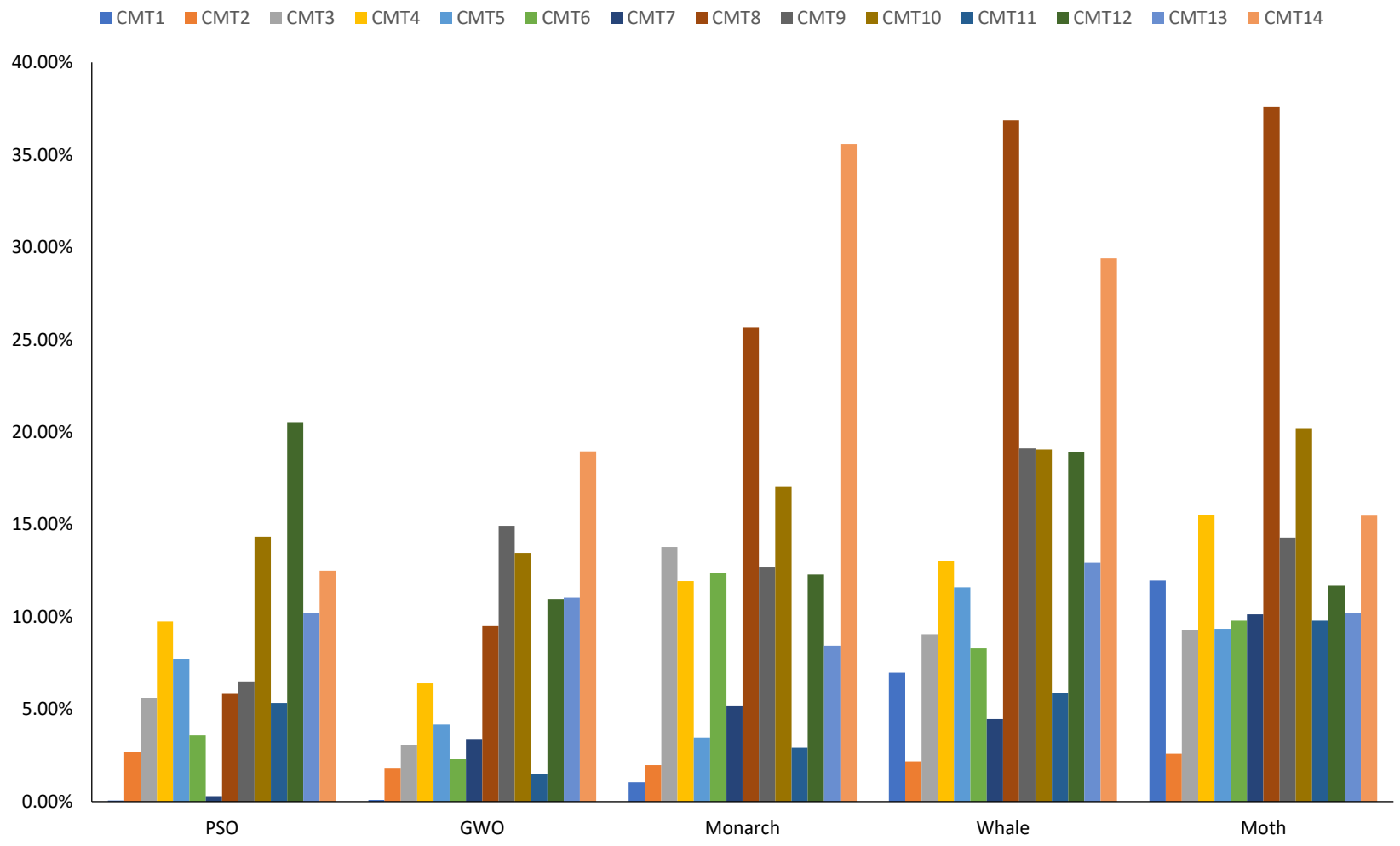
**Πίνακας:** Συγκριτική Ανάλυση Αποτελεσμάτων




**Γράφημα:** Συμπεριφορά Αλγορίθμων για τυχαία εφαρμογή – *CMT1 Instance*




**Γράφημα:** Μέση τιμή αντικειμενικής συνάρτησης πληθυσμού για τυχαία εφαρμογή – *CMT1*




**Γράφημα:** Ποσοστιαία Διαφορά Καλύτερων Αποτελεσμάτων Αλγορίθμων από τα Βέλτιστα για κάθε Πρόβλημα

Στον παραπάνω πίνακα παρουσιάζονται οι διαφορές των υπολογιστικών αποτελεσμάτων σε σχέση με τα βέλτιστα της βιβλιογραφίας. Παρατηρείται ότι στα προβλήματα με περισσότερους πελάτες η διαφορά από τα βέλτιστα παρουσιάζει αύξηση. Στα αρχικά προβλήματα η αποτελεσματικότητα των αλγορίθμων κρίνεται επαρκής. Η διαφορά των τιμών σε ποσοστιαία απόκλιση κυμαίνονται από 0.06% στη μικρότερη διαφορά έως και 13.45%. Οι λύσεις που παρουσιάστηκαν προσεγγίζουν σε ικανό βαθμό τα βέλτιστα αποτελέσματα της βιβλιογραφίας. Σχετικά, με την ανάδειξη του αποτελεσματικότερου αλγορίθμου από τους συνολικά πέντε που παρουσιάστηκαν τα καλύτερα αποτελέσματα παρήχθησαν από τον αλγόριθμο Grey Wolf. Συγκεκριμένα, ο αλγόριθμος παρήγαγε τις καλύτερες τιμές για 7 από τα συνολικά 14 προβλήματα για τα οποία δοκιμάστηκε. Ακόμα, ο αλγόριθμος Particle Swarm Optimization παρήγαγε τα καλύτερα αποτελέσματα για 5 από τα συνολικά 14 προβλήματα. Τέλος, ο αλγόριθμος Monarch Butterfly παρήγαγε τα καλύτερα αποτελέσματα για 2 από τα 14 προβλήματα. Ο αλγόριθμος Monarch Butterfly παρουσίασε ικανοποιητικά αποτελέσματα κυρίως στα προβλήματα με τους περισσότερους πελάτες. Κατά μέσο όρο οι αποκλίσεις για τις καλύτερες λύσεις είναι 6% ενώ για τις μέσες τιμές είναι 10% για τους αλγορίθμους που επέστρεψαν τις καλύτερες λύσεις. Παρατηρώντας το συνολικό γράφημα αποκλίσεων παρατηρούνται αποκλίσεις που κυμαίνονται από 0.06% έως και 37%.

## 6.5 Συμπεράσματα

Το πρόβλημα δρομολόγησης οχημάτων με περιορισμό χωρητικότητας και χρόνους εξυπηρέτησης πελατών είναι ένα από τα βασικά προβλήματα δρομολόγησης οχημάτων. Για την επίλυση του προβλήματος δοκιμάστηκαν οι παραπάνω αλγόριθμοι εμπνευσμένοι από την φύση. Οι συγκεκριμένοι αλγόριθμοι λειτουργούν χρησιμοποιώντας νοημοσύνης σμήνους και είναι βασισμένοι στις φυσικές διεργασίες οργανισμών. Η δοκιμή των αλγορίθμων έγινε στα 14 βασικά προβλήματα που προτάθηκαν από τους Christoforides, Mingozzi και Toth (1979). Από τις λύσεις που προέκυψαν οι αποκλίσεις κυμαίνονται από 0.06% έως και 37%. Η μέση τιμής διαφοράς για τις καλύτερες λύσεις είναι 6% και για τις μέσες τιμές καλύτερων λύσεων είναι 10%. Πέρα από τις στατιστικές παρατηρήσεις που αφορούν τις διαφορές των τιμών από τις βέλτιστες αξίζει να σημειωθούν ζητήματα που αφορούν την εφαρμογή των αλγορίθμων. Αρχικά, προτείνεται ο προγραμματισμός σε γλώσσες προγραμματισμού με μεγαλύτερες ταχύτητες. Η MATLAB παρουσίαζε σημαντική επιβράδυνση σε μεγαλύτερης κλίμακας προβλήματα. Επίσης, η επιλογή

μεθόδων τοπικής αναζήτησης που δεν εκτελούν τυχαίες κινήσεις αλλά στοχευμένες αλλαγές θα μπορούσαν να βελτιώσουν ακόμα περισσότερο τα υπολογιστικά αποτελέσματα όπως και τη ταχύτητα εξαγωγής αποτελεσμάτων καθώς θα μπορούσαν να μειώσουν τον αριθμό μη βελτιωτικών επαναλήψεων. Επιπρόσθετα, παρατηρήθηκε ότι οι πολλαπλές επαναλήψεις κινήσεων τοπικής αναζήτησης είχε ως αποτέλεσμα την επιδείνωση των αποτελεσμάτων. Συνίσταται, λοιπόν, η διενέργεια μικρού αριθμού κινήσεων τοπικής αναζήτησης με εφαρμογή πολλαπλών τύπων κινήσεων (π.χ. ανταλλαγή κόμβων, επανατοποθέτηση τόξων κ.α.). Αξίζει να τονισθεί ότι είναι σημαντικό να διερευνηθεί σε μεγαλύτερο βάθος η αξία των δεικτών σύγκλισης και απόκλισης των αλγορίθμων καθώς και η επίδρασή στον ορθό εντοπισμό προσωρινών λύσεων που αξίζουν διερεύνησης και όχι λύσεις που εγκλωβίζουν σε τοπικό ελάχιστο.

# 7. Βιβλιογραφία


**[1]** *Ιωάννης Μαρινάκης, Μαγδαληνή Μαρινάκη, Αθανάσιος Μυγδαλάς, Πρόβλημα Δρομολόγησης Οχημάτων στη Διαχείριση της Εφοδιαστικής Αλυσίδας, (Οκτώβρης 2019)*

**[2]** *Shi, Y., Eberhart, R. Monitoring of particle swarm optimization, Frontiers of Computer Science in China, Volume 3, 31–37 (2009)*

**[3]** *Seyedali Mirjalili, Andrew Lewis, The Whale Optimization Algorithm, Advances in Engineering Software, Volume 95, 2016, Pages 51-67*

**[4]** *Seyedali Mirjalili, Seyed Mohammad Mirjalili, Andrew Lewis, Grey Wolf Optimizer, Advances in Engineering Software, Volume 69, 2014, Pages 46-61*

**[5]** *Seyedali Mirjalili, Moth-flame optimization algorithm: A novel nature-inspired heuristic paradigm, Knowledge-Based Systems, Volume 89, 2015, Pages 228-249,*

**[6]** *Yannis Marinakis, Georgia-Roumbini Iordanidou, Magdalene Marinaki, Particle Swarm Optimization for the Vehicle Routing Problem with Stochastic Demands, Applied Soft Computing, Volume 13, Issue 4, 2013, Pages 1693-1704,*

**[7]** *M. Ghetas, C. H. Yong and P. Sumari, "Harmony-based monarch butterfly optimization algorithm," 2015 IEEE International Conference on Control System, Computing and Engineering (ICCSCE), 2015, pp. 156-161,*

**[8]** *Hansen, P., Mladenović, N., Todosijević, R. et al. Variable neighborhood search: basics and variants. EURO J Comput Optim 5, 423–454 (2017).*

**[9]** *Can B. Kalayci, Can Kaya, An ant colony system empowered variable neighborhood search algorithm for the vehicle routing problem with simultaneous pickup and delivery, Expert Systems with Applications, Volume 66, 2016, Pages 163-175*

**[10]** *Florian Arnold, Michel Gendreau, Kenneth Sörensen, Efficiently solving very large-scale routing problems, Computers & Operations Research, Volume 107, 2019, Pages 32-42*

**[11]** *Amine Dhahri, Anis Mjirda, Kamel Zidi, Khaled Ghedira, A VNS-based Heuristic for Solving the Vehicle Routing Problem with Time Windows and Vehicle Preventive Maintenance Constraints, Procedia Computer Science, Volume 80, 2016, Pages 1212-1222*

**[12]** *Hideki Hashimoto, Mutsunori Yagiura, Toshihide Ibaraki, An iterated local search algorithm for the time-dependent vehicle routing problem with time windows, Discrete Optimization, Volume 5, Issue 2, 2008,Pages 434-456,*

**[13]** *Bio-inspired Algorithms for the Vehicle Routing Problem, Editors: Pereira, Francisco Baptista, Tavares, Jorge (Eds.), Studies in Computational Intelligence, Series Volume 161 (2009)*

**[14]** *Toth P, Vigo D. 1. An overview of vehicle routing problems. Vehicle Routing Problem 2002, Volume 1, Pages 1-26*

**[15]** *Eduardo Uchoa, Diego Pecin, Artur Pessoa, Marcus Poggi, Thibaut Vidal, Anand Subramanian, New benchmark instances for the Capacitated Vehicle Routing Problem, European Journal of Operational Research, Volume 257 Issue 3, 2017, Pages 845 - 858*

**[16]** *Florian Arnold, Kenneth Sorensen Knowledge-guided local search for the Vehicle Routing Problem - Operations Research Group, Computers and Operations Research, Volume 105, Pages 32 - 46*

**[17]** *Y. Shen, M. Liu, J. Yang, Y. Shi and M. Middendorf, "A Hybrid Swarm Intelligence Algorithm for Vehicle Routing Problem with Time Windows," in IEEE Access, vol. 8, Pages 93882-93893, 2020*

**[18]** *Asma M. Altabeeb, Abdulqader M. Mohsen, Laith Abualigah, Abdullatif Ghallab, "Solving capacitated vehicle routing problem using cooperative firefly algorithm", Applied Soft Computing Asma M. Volume 108, 2021, Pages 107403 - 107418,*