ΠΟΛΥΤΕΧΝΕΙΟ ΚΡΗΤΗΣ

ΣΧΟΛΗ ΜΗΧΑΝΙΚΩΝ ΠΑΡΑΓΩΓΗΣ ΚΑΙ ΔΙΟΙΚΗΣΗΣ

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Εξελικτικοί αλγόριθμοι για το ανοικτό – κλειστό πρόβλημα δρομολόγησης οχημάτων.

ΣΚΕΥΟΦΥΛΑΞ ΠΑΝΑΓΙΩΤΗΣ

Επιβλέπων Καθηγητής : Ιωάννης Μαρινάκης

Χανιά 2019

Ευχαριστίες

Αρχικά θα ήθελα να ευχαριστήσω τον επιβλέποντα της διπλωματικής μου εργασίας κ. Ιωάννη Μαρινάκη για την πολύτιμη βοήθεια και την σωστή καθοδήγηση του, επίσης θα ήθελα να ευχαριστήσω τον κ. Λευτέρη Τσακιράκη για την άριστη συνεργασία που είχαμε καθ' όλη την διάρκεια της εκπόνησης διπλωματικής μου εργασίας. Επιπλέων θέλω να ευχαριστήσω τους φίλους μου για τα υπέροχα αυτά χρόνια και τις αξέχαστες αναμνήσεις που περάσαμε. Τέλος θέλω να ευχαριστήσω την οικογένεια μου για την ηθική αλλά και την οικονομική στηρίξει που μου παρείχε όλα αυτά το χρόνια.

ΠΕΡΙΕΧΟΜΕΝΑ

Περίληψη

Σε μια καταναλωτική κοινωνία όπου παρατηρείτε έντονη ανταγωνιστικότητα καθώς και επιλεκτική αγορά προϊόντων ο τομές της διαχείρισης της εφοδιαστικής αλυσίδας είναι σημαντικός για την βιωσιμότητα μια εταιρίας. Στην παρούσα διπλωματική εργασία ασχολούμαστε με το ανοιχτό κλειστό πρόβλημα δρομολογήσης οχημάτων (open close vehicle routing problem). Το συγκεκριμένο πρόβλημα εφαρμόζεται κυρίως από εταιρίες στις οποίες ο αριθμός των ιδιόκτητων οχημάτων, δεν επαρκεί για την συνολική κάλυψη της ζήτησης. Σκοπός του προβλήματος είναι η εύρεση μονοπατιών ελάχιστου κόστους. Όλα τα οχήματα ξεκινώντας από την αποθήκη προσπαθούν να καλύψουν την συνολική ζήτηση, χωρίς να παραβιάζονται οι περιορισμοί της χωρητικότητας καθώς και της μέγιστης απόσταση που μπορούν να διανύσουν. Η ιδιαιτερότητα του συγκεκριμένου προβλήματος είναι, τα ενοικιαζόμενα οχήματα να μην έχουν τη δυνατότητα να επιστρέψουν στην αποθήκη.

Αρχικά για την εύρεση μια αρχικής εφικτής λύσης χρησιμοποιήθηκε ο αλγόριθμος του GRASP και στην συνέχεια, για την βελτίωση της, εφαρμόστηκε ο αλγόριθμος Ant Colony Optimization, σε συνδυασμό με τρεις αλγόριθμους τοπικής αναζήτησης (2-opt , 1-1 exchange, 1-0 relocate). Ο αλγόριθμος εφαρμόστηκε σε γνωστά παράδειγμα από την βιβλιογραφία όπου και συγκριθήκαν τα αποτελέσματα τους. Για την επίλυση του συγκεκριμένου προβλήματος αναπτύχθηκε αλγόριθμος σε περιβάλλον matlab.

Κεφάλαιο 1 - Εισαγωγή

1.1 Εφοδιαστική

Η εφοδιαστική (Logistics) με την έννοια που πλησιάζει σήμερα, ανάγεται στην εποχή των Ρωμαϊκών χρόνων όταν η ταχεία ανάπτυξη των λεγεώνων στα διάφορα σημεία των αυτοκρατορικών συνόρων για την αντιμετώπιση βαρβαρικών επιδρομών, ήταν αναγκαία και έθετε ιδιαίτερες απαιτήσεις στην ύπαρξη οδών και στην επιτυχή και ταχεία μεταφορά διαταγών, υλικών και ανδρών. Ο όρος Logistics αναφέρεται στη φυσική διανομή των προϊόντων και συγκεκριμένα σε εκείνο το τμήμα της Διαχείρισης Εφοδιαστικής Αλυσίδας που σχεδιάζει, υλοποιεί και ελέγχει την αποδοτική και αποτελεσματική κανονική και αντίστροφη ροή και αποθήκευση των προϊόντων, υπηρεσιών και των σχετικών πληροφοριών από το σημείο προέλευσης τους έως το σημείο κατανάλωσης τους, ώστε να ικανοποιηθούν οι απαιτήσεις των πελατών. Ωστόσο η εφοδιαστική δεν σημαίνει μόνο "μεταφορά". Υπάρχουν πολλές διαφορετικές δραστηριότητες εφοδιαστικής ή λειτουργίες εφοδιαστικής που χρησιμοποιούνται από μια εταιρεία.

Ενδεικτικές περιοχές εφαρμογών των Logistics περιλαμβάνουν τα: Business Logistics, Systems Logistics, Maritime Logistics, Logistics Υγείας, Logistics Στρατού, Περιβαλλοντικά Logistics, City Logistics, Crisis Logistics, Logistics Υπηρεσιών, Agro-logistics, Supply Chain Management Logistics Information Systems και Reverse Logistics.

Τα Logistics βρίσκουν εφαρμογή σε δύο κυρίως πεδία :

- Το πρώτο πεδίο είναι η επιχείρηση, η οποία πρέπει να οργανώσει την εισροή, την εσωτερική διακίνηση και την εκροή υλικών και προϊόντων κατά τέτοιον τρόπο, έτσι ώστε να εξασφαλίζει τη μέγιστη ικανοποίηση των πελατών της.
- Το δεύτερο πεδίο είναι η εφοδιαστική αλυσίδα, η οποία αποτελείται από όλες εκείνες τις επιχειρήσεις και οργανισμούς που είναι απαραίτητοι έτσι ώστε ένα προϊόν, από πρώτες ύλες να καταλήξει στον τελικό πελάτη.

Η αποτελεσματική οργάνωση και διοίκηση της ροής προϊόντων και πληροφοριών σε αυτήν την αλυσίδα αποτελεί επιτακτική ανάγκη σε μία παγκοσμιοποιημένη και ψηφιακή οικονομία, όπου ο ανταγωνισμός από ατομικός (επιχείρηση εναντίον επιχείρησης) γίνεται συλλογικός (εφοδιαστική αλυσίδα εναντίον εφοδιαστικής αλυσίδας).

1.2 Εφοδιαστική Αλυσίδα

Οδηγούμενοι από την παγκοσμιοποίηση και τις διαρκώς εκτεινόμενες απαιτήσεις των πελατών, η εφοδιαστική αλυσίδα παίζει καθοριστικό ρόλο στην δημιουργία πλεονεκτήματος για όλες τις επιχειρήσεις. Με τον όρο εφοδιαστική αλυσίδα εννοούμε όχι μόνο τη ροή υλικών από τον προμηθευτή πρώτων υλών ή τον κατασκευαστή μέχρι τον τελικό καταναλωτή, αλλά παράλληλα και την ροή πληροφοριών μεταξύ των μελών της ίδιας αλυσίδας. Αποτελείτε απ' όλα τα στάδια που εμπλέκονται έμμεσα ή άμεσα, στην

ικανοποίηση των απαιτήσεων των πελατών. Συνεπώς, η εφοδιαστική αλυσίδα αποτελείται από τους κατασκευαστές και προμηθευτές, από χώρους αποθήκευσης, κέντρα διανομών μεταφορείς, πωλητές λιανικής, πελάτες αλλά και από τις πρώτες ύλες, αποθέματα κατά την διαδικασία παραγωγής και έτοιμα προϊόντα. Η διαχείριση της γίνεται σε 2 στάδια:

- Επίπεδο προγραμματισμού: αναλύονται τα δεδομένα των προμηθειών, αναλώσεων παραγωγής, αποθεματοποίηση και πωλήσεων. Γίνονται προβλέψεις και πλάνα στα οποία βασίζεται ο προγραμματισμός.
- Επίπεδο εκτέλεσης: στο στάδιο αυτό εκτελείται το πλάνο που έχει καθοριστεί στο επίπεδο του προγραμματισμού και παρακολουθείται η εξέλιξη του βάση των δεδομένων και πληροφοριών που συλλέγονται από όλο το εύρος της εφοδιαστικής αλυσίδας.

Η ικανοποίηση των πελατών, η ανάπτυξη νέων προϊόντων, η προβολή των προϊόντων στην αγορά, η χρηματοδότηση, η εξυπηρέτηση πελατών κ.α. αποτελούν επίσης συστατικά στοιχεία της εφοδιαστικής αλυσίδας

Είναι προφανές ότι η εφοδιαστική αλυσίδα λαμβάνει υπόψη όλα τα έξοδα που δημιουργούνται στα διάφορα στάδια καθώς και από την αλληλεπίδραση των διαφόρων σταδίων. Τα κύρια κριτήρια για την απόδοση της εφοδιαστικής αλυσίδας είναι :

- Το ολικό κόστος
- Το ολικό κέρδος
- Ο χρονικός κύκλος

Το ολικό κόστος που προέρχεται από τις μεταφορές την διανομή κ.λ.π. πρέπει να ελαχιστοποιηθεί. Στόχος είναι να ληφθεί υπόψη για ελαχιστοποίηση όλο το σύστημα και όχι το κάθε κόστος χωριστά.

Το ολικό κέρδος αναφέρεται στο ολικό κέρδος που πρέπει να διανεμηθεί κατά μήκος της αλυσίδας Ομοίως και εδώ δεν θα πρέπει η απόδοση να μετρηθεί με τα επιμέρους κέρδη των μεμονωμένων στοιχείων της.

Ο χρονικός κύκλος είναι ο ολικός χρόνος που απαιτείται για να ολοκληρωθεί η ολική διαδικασία από τις πρώτες ύλες στο έτοιμο προϊόν στον πελάτη. Υπολογίζεται ότι μόνο το 5% του κύκλου χρησιμοποιείτε για την εκτέλεση της πραγματικής διαδικασίας .

1.3 Διαχείριση Εφοδιαστικής Αλυσίδας

Η ανάγκη για συνεχή εφοδιασμό και για αποθέματα εμφανίστηκε από τα αρχαία χρόνια. Χαρακτηριστικό παράδειγμα είναι ότι η εκστρατεία του Μεγάλου Αλέξανδρου προς την Ασία, θα ήταν ανέφικτη χωρίς να διαθέτει το σωστό σύστημα εφοδιασμού. Όμως οι άνθρωποι άρχισαν να ασχολούνται με τη Διαχείριση Εφοδιαστικής Αλυσίδας (ΔΕΑ), με την έννοια αυτή που γνωρίζουμε, από το 1900 και μετά .

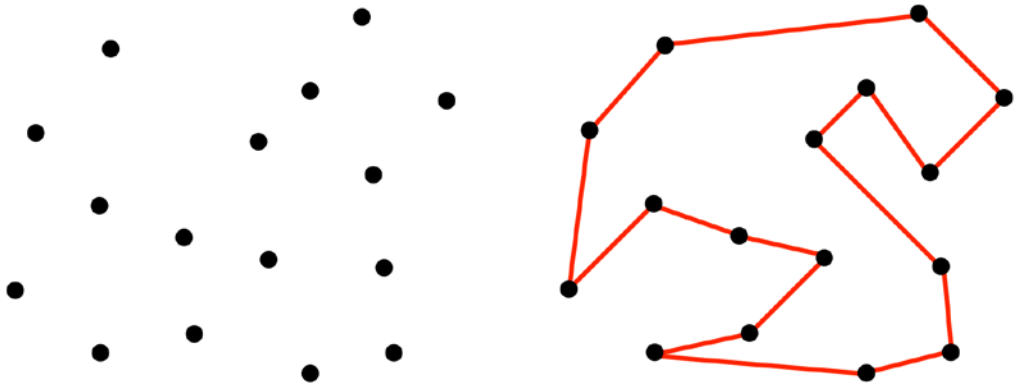
Ως Διαχείριση Εφοδιαστικής Αλυσίδας (SCM) ορίζεται ο σχεδιασμός, η οργάνωση, και ο συντονισμός όλων των δραστηριοτήτων της εφοδιαστικής αλυσίδας. Όπου με τον όρο εφοδιαστική αλυσίδα (ΕΑ) εννοούμε την ροή υλικών, πληροφοριών και υπηρεσιών από τους προμηθευτές πρώτων υλών μέσα από τα εργοστάσια και τις αποθήκες, στους τελικούς πελάτες.

Η Διαχείριση της Εφοδιαστικής Αλυσίδας (Supply Chain Management) αποτελεί ένα σχετικά νέο και πολλά υποσχόμενο τομέα της επιστήμης, με μεγάλη επίδραση στην αποτελεσματικότητα των σημερινών επιχειρήσεων και στην ευρύτερη διασφάλιση ποιοτικών διαδικασιών, στο ιδιαίτερα ανταγωνιστικό περιβάλλον της σύγχρονης επιχειρηματικότητας. Η διάδοσή της οφείλεται κατά κύριο λόγο στα ιδιαίτερα σημαντικά αποτελέσματα που επιφέρει, τόσο προς την κατεύθυνση της μείωσης του κόστους των επιχειρήσεων όσο και προς την κατεύθυνση του βέλτιστου συντονισμού των διεργασιών της επιχείρησης που συνδέονται με τους προμηθευτές και τους διανομείς. Με την ολοκληρωμένη εφαρμογή της διαχειρίσεις αυτής ο πελάτης βρίσκει το προϊόν την κατάλληλη στιγμή, στην κατάλληλη ποιότητα και ποσότητα και στην καταλληλότερη τιμή, περιορίζοντας ουσιαστικά όλους εκείνους τους παράγοντες που αυξάνουν το κόστος του προϊόντος.

Κεφάλαιο 2 - Δρομολόγηση Οχημάτων

2.1 Το Πρόβλημα του Περιπλανώμενου Πωλητή (Travelling Salesman Problem)

Το πρόβλημα του πλανόδιου πωλητή (Travelling salesman problem-TSP) αφορά την εύρεση της συντομότερης (σε χρόνο, απόσταση ή άλλο κόστος) διαδρομής για ένα όχημα (ή πωλητή) με αφετηρία κάποιο σημείο, π.χ. ένα κέντρο διανομής, και επιστροφή στο ίδιο σημείο αφού πρώτα επισκεφθεί έναν σταθερό αριθμό πελατών ακριβώς μια φορά τον καθένα. Η αντίληψη που έχουμε για την συγκεκριμένη διαδρομή είναι του σχήματος 1, δηλαδή ένας κύκλος που ξεκινάει από έναν κόμβο (αφετηρία) και αφού επισκεφτεί όλους τους κόμβους μια φορά επιστρέφει σε αυτόν.



Σχήμα 1: Επίλυση του πλανόδιου πωλητή (πηγή: algorist.com)

2.2 Το πρόβλημα Δρομολόγησης Οχημάτων (VRP)

Το πρόβλημα δρομολόγησης οχημάτων (Vehicle routing problem) είναι ένα από τα ποιο διαδεδομένα και ευρέως μελετημένα προβλήματα στη συνδυαστική βελτιστοποίηση. Παρουσιάστηκε για πρώτη φορά από τους Dantzig και Ramser το 1959 και από τότε κατέχει κεντρική θέση στους τομείς της φυσικής διανομής και των logistics. Το πρόβλημα δρομολόγησης οχημάτων είναι στην ουσία επέκταση του προβλήματος του πλανόδιου πωλητή στις περιπτώσεις όπου ένα μόνο όχημα δεν δύναται να επισκεφθεί όλους τους πελάτες. Ο προσδιορισμός της βέλτιστης λύσης, είναι δυσκολίας–NP, με αποτέλεσμα τα προβλήματα που μπορούν να επιλυθούν με βέλτιστο τρόπο να είναι περιορισμένα. Επομένως, για την λύση των προβλημάτων χρησιμοποιούνται κυρίως ευρετικοί αλγόριθμοι. Το VRP έχει πολλές εμφανείς εφαρμογές στην βιομηχανία. Στην πραγματικότητα, η χρήση προγραμμάτων βελτιστοποίησης μπορεί να προσφέρει εξοικονόμηση 5% σε μια επιχείρηση

καθώς η μεταφορά είναι συνήθως ένα σημαντικό στοιχείο του κόστους ενός προϊόντος. Ο τομέας των μεταφορών αποτελεί 10 % του ΑΕγχΠ της Ευρωπαϊκής Ένωσης. Κατά συνέπεια κάθε εξοικονόμηση που δημιουργείται ακόμα και λιγότερο από 5%, είναι σημαντική. Στο συγκεκριμένο πρόβλημα υπάρχει μια αποθήκη (αφετηρία όλων των οχημάτων), ένα σύνολο πελατών με γνωστή ζήτηση και ένας στόλος οχημάτων με συγκεκριμένη χωρητικότητα. Επίσης είναι γνωστές οι αποστάσεις από την αποθήκη προς όλους τους πελάτες, καθώς και οι αποστάσεις μεταξύ των πελατών. Ο κάθε πελάτης πρέπει να εξυπηρετηθεί από ένα μόνο όχημα. Ένα όχημα δεν μπορεί να εξυπηρετήσει κάποιον πελάτη ο οποίος έχει μεγαλύτερη ζήτηση από την χωρητικότητα του. Σκοπός του συγκεκριμένου προβλήματος είναι να βρεθούν οι διαδρομές που εκτελούν τα οχήματα οι οποίες ελαχιστοποιούν το συνολικό κόστος δεδομένου ότι, τα οχήματα πρέπει να επιστρέψουν στην αποθήκη.

Παρακάτω παρουσιάζεται η μοντελοποίηση του προβλήματος που παρουσιάστηκε από τους Fisher & Jaikumar.

Έστω

$$x_{ijk} = \begin{cases} 1, & εαν\ το\ οχημα\ k\ επισκέπτεται\ τον\ πελάτη\ j\ αμέσως\ μετά\ τον\ i \\ 0, & αλλιώς \end{cases}$$

$$y_{jk} = \begin{cases} 1, & εαν\ ο\ πελάτης\ i\ επισκέπτεται\ απο\ το\ όχημα\ k \\ 0, & αλλιώς \end{cases}$$

Αντικειμενική συνάρτηση :

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

υπό:

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

$$\sum_i q_i y_{ij} \le Q_k \qquad k = 1,\dots,m \qquad (2)$$

$$\sum_j x_{ijk} = \sum_j x_{jik} = y_{ik} \quad i = 1,\dots,n \quad (3)$$

$$\sum_{i,j\in S} x_{ijk} \le |S| - 1 \quad για\ όλα\ τα\ S \subseteq \{2,\dots,n\}, \quad k = 1,\dots,m \quad (4)$$

$$y_{ik} \in \{0,1\}, \quad \begin{matrix} k = 1,\dots,m \\ i = 1,\dots,n \end{matrix} \qquad (5)$$

$$x_{ijk} \in \{0,1\}, \quad \begin{matrix} i,j = 1,\dots,n \\ k = 1,\dots,m \end{matrix} \qquad (6)$$

Ο περιορισμός (1) δείχνει ότι κάθε πελάτης εκχωρείται σε ένα μόνο όχημα, εκτός από την αποθήκη που την επισκέπτονται όλα τα οχήματα , ο περιορισμός (2) είναι ο περιορισμός της χωρητικότητας των οχημάτων , ο περιορισμός (3) μας εξασφαλίζει ότι ένα όχημα που επισκέπτεται ένα πελάτη φεύγει από αυτόν.

2.3 Πρόβλημα Δρομολόγησης Οχημάτων Περιορισμένης Χωρητικότητας Και Απόστασης

Το πρόβλημα δρομολόγησης οχημάτων με περιορισμένη χωρητικότητα και απόσταση είναι μια παραλλαγή του κλασικού προβλήματος δρομολόγησης οχημάτων με μόνη διαφορά την προσθήκη του περιορισμού της απόστασης ή αλλιώς τον διαθέσιμο χρόνο του οχήματος. Στο συγκεκριμένο πρόβλημα τοποθετούνται περιορισμοί στο χρονικό διάστημα όπου το όχημα μπορεί να ξοδέψει μακριά από την αποθήκη ή την μέγιστη απόσταση σε χιλιόμετρα που μπορεί να διάνυσει ένα όχημα χωρίς να επιστρέψει σε αυτήν. Για την μοντελοποίηση του προβλήματος πρέπει να υπάρχει μια αποθήκη, ένα σύνολο πελατών με γνωστή ζήτηση και ένα σύνολο οχημάτων με συγκεκριμένη χωρητικότητα τα οποία μπορούν να διανύσουν μέχρι μια δεδομένη απόσταση. Οι αποστάσεις μεταξύ των πελατών και πελατών-αποθήκης είναι γνώστες. Σκοπός του προβλήματος είναι η εύρεση των διαδρομών που ελαχιστοποιούν το κόστος.

2.3 Το ανοιχτό Πρόβλημα Δρομολόγησης Οχημάτων

Στις αρχές του αιώνα ένα νέο πρόβλημα το Ανοιχτό Πρόβλημα Δρομολόγησης Οχημάτων (open-VRP) μελετήθηκε από τους ερευνητές (Sariklis και Powell, 2000). Το ανοιχτό πρόβλημα δρομολόγησης οχημάτων είναι στενά συνδεδεμένο με το κλασικό πρόβλημα δρομολόγησης οχημάτων, αλλά σε αντίθεση με αυτό η διαδρομή του οχήματος τελειώνει μόλις εξυπηρετηθεί και ο τελευταίος πελάτης καθώς τα οχήματα δεν χρειάζεται να επιστρέψουν στην αποθήκη. Το συγκεκριμένο πρόβλημα χρησιμοποιείται κυρίως από εταιρίες οι οποίες δεν έχουν ιδιόκτητα οχήματα ή τα οχήματα τους είναι ακατάλληλα για την μεταφορά των συγκεκριμένων προϊόντων. Αυτό έχει ως αποτέλεσμα την ενοικίαση οχημάτων για την διανομή των προϊόντων τους. Τα ενοικιαζόμενα όχημα αν και ξεκινούν από την αποθήκη δεν χρειάζεται να επιστρέψουν σε αυτήν. Για την εύρεση της βέλτιστης λύσης, το συγκεκριμένο πρόβλημα πρέπει να λυθεί ξεχωριστά και όχι σαν κλειστό πρόβλημα δρομολόγησης οχημάτων, στο όποιο διαγράφουμε την επιστροφή στην αποθήκη.

2.4 Το Ανοιχτό Κλειστό Πρόβλημα Δρομολόγησης Οχημάτων

Το ανοιχτό κλειστό πρόβλημα δρομολόγησης οχημάτων (COMVRP) είναι ένας συνδυασμός του κλειστού προβλήματος δρομολόγησης οχημάτων (Close-VRP) και του ανοιχτού προβλήματος δρομολόγησης οχημάτων (Open-VRP). Τόσο το Open-VRP όσο και το Close-

VRP λαμβάνονται υπόψη για το σχεδιασμό των βέλτιστων διαδρομών που χρησιμοποιούνται από έναν στόλο οχημάτων με σκοπό την εξυπηρετήση ενός συνόλου πελατών. Στο συγκεκριμένο πρόβλημα όλα τα οχήματα ξεκινούν υποχρεωτικά από την αποθήκη, αλλά μόνο τα ιδιόκτητα είναι υποχρεωμένα να γυρίσουν σε αυτήν. Είναι λοιπόν προφανές ότι τα ενοικιαζόμενα οχήματα παράγουν αποτελεσματικότερες διαδρομές αφού δεν είναι υποχρεωμένα να επιστρέψουν στην αποθήκη. Στα συγκεκριμένα οχήματα πρέπει να ληφθεί υπόψη και το κόστος ενοικίασης. Σημαντικό είναι να γραφεί αλγόριθμος που να επιλύει το συγκεκριμένο πρόβλημα και να μην γίνει συνδυασμός των λύσεων του Close-VRP και του Open-VRP διότι δεν θα βρεθεί βέλτιστη λύση. Ωστόσο η δημιουργία αλγορίθμου που να επιλύει το COMVRP κάθε άλλο παρά εύκολη μπορεί να θεωρηθεί. Τόσο το Close-VRP όσο και το Open-VRP έχουν αποδειχτεί ότι είναι δυσκολίας NP-hard (Brandao, 2004; Laporte,2009). Ομοίως και το COMVRP θεωρείτε δυσκολίας NP-hard καθώς περιορίζεται στο Close-VRP όταν ο αριθμός των οχημάτων που εκτελούν ανοιχτές διαδρομές είναι μηδέν, ενώ αν ο αριθμός των οχημάτων που εκτελούν κλειστές διαδρομές είναι μηδέν τότε θεωρείτε Open-VRP. Το COMVRP εφαρμόζεται κυρίως από εταιρίες στις οποίες δεν επαρκούν τα ιδιόκτητα οχήματα για την εξυπηρέτηση των πελατών της (π.χ. λόγω αυξημένης ζήτηση).

Παρακάτω παρουσιάζεται η μοντελοποίηση του COMVRP.

Οι μεταβλητές που χρησιμοποιούντα είναι :

x_{ij}^{k}= 1 αν το όχημα k αμέσως μετά τον i πελάτη επισκεφθεί το j ,0 αλλιώς.

u_i :συνεχείς μεταβλητή , χρησιμοποιείται σαν άνω φράγμα για τη χωρητικότητα του οχήματος.

h_i : συνεχείς μεταβλητή , χρησιμοποιείται σαν άνω φράγμα για τη απόσταση που μπορεί να διανύσει το όχημα.

q_i: ζήτηση πελάτη i

Q_k: χωρητικότητα οχήματος k

H_k: μέγιστη απόσταση οχήματος

F_k: κόστος χρησιμοποίησης οχήματος k

C_{ij}: απόσταση μεταξύ κόμβων i,j

Nu: μέγιστος αριθμός ιδιωτικών οχημάτων

N: αριθμός πελατών

K: αριθμός οχημάτων

Αντικειμενική συνάρτηση :

min $\sum_{k\in K} F_k \sum_{i\in N} x_{0i}^k + \sum_{k\in K} \sum_{i\in V} \sum_{j\in V} c_{ij}\, x_{ij}^k - \sum_{i\in N} c_{i0}\, x_{i0}^1$

υπο:

$\sum_{k\in K} \sum_{i\in V, i\neq j} x_{ij}^k = 1 \quad \forall \text{j} \in \text{N}$ (1)

$\sum_{i\in V, i\neq j} x_{ij}^k = \sum_{i\in V, i\neq j} x_{ji}^k \quad \forall \text{j} \in V, k \in K$ (2)

$u_i + q_j - \left(1 - x_{ij}^k\right)M \leq u_j \quad \forall i,\ \text{j} \in N,\ k \in K,\ i \neq j$ (3)

$q_i \leq u_i \leq \sum_{k\in K} \sum_{j\in V} x_{ij}^k\, Q_k \quad \forall \text{ i} \in \text{N}$ (4)

h_0=0 (5)

$h_i + c_{ij} - \left(1 - x_{ij}^k\right)M \leq h_j \quad \forall \text{i} \in V,\ \text{j} \in N,\ k \in K,\ i \neq j$ (6)

$h_i + c_{i0} - \left(1 - x_{ij}^0\right)M \leq H_0 \quad \forall \text{ i} \in \text{N}$ (7)

$0 \leq h_i \leq \sum_{k\in K} \sum_{j\in V} x_{ij}^k\, H_k \quad \forall \text{ i} \in \text{N}$ (8)

$\sum_{i\in N} x_{0i}^0 \leq N_u$ (9)

$x_{ij}^k \in \{0,1\} \quad \forall \text{ i} \in \text{V},\ \text{j} \in V,\ i \neq j$ (10)

Στην αντικειμενική συνάρτηση ο πρώτος όρος ($\sum_{k\in K} F_k \sum_{i\in N} x_{0i}^k$) είναι το κόστος των οχημάτων, ο δεύτερος όρος ($\sum_{k\in K} \sum_{i\in V} \sum_{j\in V} c_{ij}\, x_{ij}^k$) αντιπροσωπεύει το συνολικό κόστος. Θεωρούμε ότι όλα τα οχήματα επιστρέφουν στην αποθήκη και στον τρίτο ορό ($\sum_{i\in N} c_{i0}\, x_{i0}^1$) αφαιρούμε το κόστος επιστροφής στην αποθήκη από το ενοικιαζόμενα οχήματα, καθώς η διαδρομή τους τελειώνει μόλις επισκεφτούν τον τελευταίο πελάτη. Ο σκοπός είναι η ελαχιστοποίηση της αντικειμενικής συνάρτησης. Όσο αναφορά τους περιορισμούς, ο περιορισμός (1) μας εξασφαλίζει ότι τον κάθε πελάτης θα τον επισκεφτεί ένα μόνο όχημα, ο περιορισμός (2) μας εξασφαλίζει ότι, όταν ένα όχημα επισκεφτεί έναν πελάτη θα φύγει από αυτόν. Οι επόμενοι δύο περιορισμοί (3) και (4) αποτρέπουν την υπερφόρτωση των οχημάτων. Οι περιορισμοί (5),(6), (7) και (8) έχουν να κάνουν με την μέγιστη απόσταση που μπορούν να διανύσουν τα οχήματα και ο περιορισμός (9) έχει να κάνει με τον μέγιστο αριθμό ιδιωτικών οχημάτων.

Στην παρούσα διπλωματική εργασία το ανοιχτό κλειστό πρόβλημα δρομολόγησης οχημάτων επιλύθηκε με δύο διαφορετικές παραλλαγές. Στην πρώτη περίπτωση για δεδομένο αριθμό ιδιόκτητων οχημάτων, τα οποία δεν επαρκούν για την πλήρη κάλυψη της ζήτησης, προσπαθούμε να ελαχιστοποιήσουμε το κόστος. Για την κάλυψη της συνολικής ζήτησης χρησιμοποιούνται ενοικιαζόμενα οχήματα. Όλα τα οχήματα έχουν συγκεκριμένη

χωρητικότητα και είναι διαθέσιμα για περιορισμένο χρόνο. Τα ιδιωτικά οχήματα έχουν ένα κόστος λειτουργίας το όποιο προφανώς είναι μικρότερο από το κόστος ενοικίασης των ενοικιαζόμενων οχημάτων. Στην συγκεκριμένη παραλλαγή τα οχήματα έχουν την δυνατότητα επιστροφής στην αποθήκη για επαναφόρτιση, δηλαδή έχουν την δυνατότητα εκτέλεσης περισσότερων από μια διαδρομές. Η δεύτερη περίπτωση είναι παρόμοια με την πρώτη, με μόνη διαφορά ότι τα οχήματα δεν έχουν την δυνατότητα επιστροφής στην αποθήκη για επαναφόρτιση, δηλαδή κάθε όχημα εκτελεί ακριβώς μια διαδρομή. Με τη συγκεκριμένη παραλλαγή προσπαθούμε να αποφύγουμε τη δημιουργία διαδρομών με μικρό αριθμό πελατών.

Κεφάλαιο 3 - Αλγόριθμοι Βελτιστοποίησης Προβλημάτων Εφοδιαστικής Αλυσίδας

3.1 Απλοί Ευρετικοί Αλγόριθμοι

Οι ευρετικοί αλγόριθμοι είναι τεχνικές σχεδιασμένες να επιλύουν προβλήματα γρηγορότερα όταν οι κλασικές τεχνικές είναι πολύ αργές ή να βρίσκουν προσεγγιστικές λύσεις όταν οι κλασικές αδυνατούν να βρουν ακριβής λύση. Αυτό επιτυγχάνεται καθώς δίνουν βάση στην ταχύτητα και όχι στην εύρεση της βέλτιστης λύσης. Μια λύση ενός ευρετίκου αλγορίθμου γίνεται αποδεκτή αν ικανοποιεί κάποια κριτήρια όπως η ποιότητα της λύσης, δηλαδή η απόκλιση της από την βέλτιστη. Στην συγκεκριμένη κατηγορία ανήκουν και οι παρακάτω αλγόριθμοι:

- Αλγόριθμοι απληστίας (greedy algorithms)
- Προσεγγιστικοί αλγόριθμοι (approximation algorithms)
- Αλγόριθμοι τοπικής αναζήτησης (local search algorithms)

Στην παρούσα διπλωματική θα ασχοληθούμε με τους: αλγόριθμους απληστίας και αλγόριθμους τοπικής αναζήτησης .

3.1.1 Αλγόριθμοι Απληστίας (Greedy Algorithms)

Οι αλγόριθμοι απληστίας προσπαθούν να οδηγήσουν σε μια εφικτή λύση του προβλήματος. Πολλές φορές χρειάζονται πάρα πολύ μεγάλο χρόνο γιατί είναι μυωπικοί αλγόριθμοι, δηλαδή βλέπουν μόνο μπροστά. Η λογική τους είναι απλή, κάποιος ξεκινάει από μια μερική μη-εφικτή λύση και σε κάθε βήμα καθορίζει μια ή και περισσότερες μεταβλητές μέχρι να βρεθεί μια εφικτή λύση, η οποία είναι και η λύση του προβλήματος.

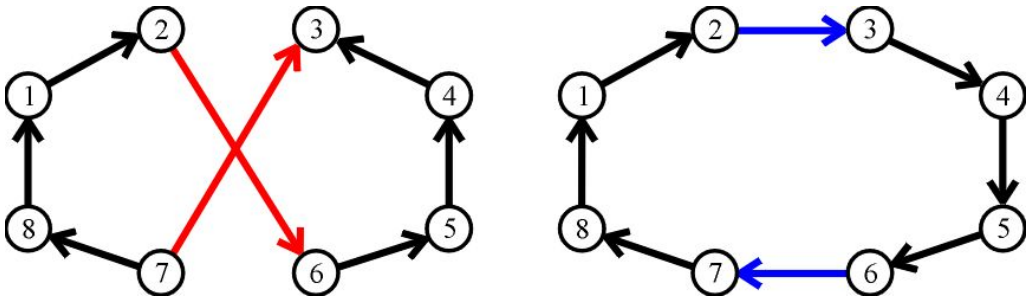
3.2 Αλγόριθμοι Τοπικής Αναζήτησης

Η τοπική αναζήτηση βασίζεται στην αρχαιότερη μέθοδο βελτιστοποίησης, στην μέθοδο δοκιμής και σφάλματος. Η ιδέα είναι τόσο απλή που είναι εκπληκτικό πόσο επιτυχημένη έχει αποδειχτεί. Οι αλγόριθμοι τοπικής αναζήτησης προσπαθούν από μια αρχική εφικτή λύση να βελτιώσουν τη λύση, με κάποια μέθοδο αναζήτησης στην γειτονιά της λύσης. Ένα σημαντικό χαρακτηριστικό της τοπικής αναζήτησης είναι το γεγονός ότι μπορεί να εκτελείται από πολλά διαφορετικά αρχικά σημεία και να επιλέγεται το καλύτερο σαν βέλτιστη λύση. Τέλος, ίσως το σημαντικότερο στοιχείο για την επιτυχία της διαδικασίας είναι η επιλογή της μεθόδου που θα χρησιμοποιηθεί για την αναζήτηση. Στην παρούσα

διπλωματική εργασία χρησιμοποιήθηκαν τρεις τοπικές αναζήτησης η 2 opt, η 1-1 exchange και η 1-0 relocate οι οποίες θα αναλυθούν στην συνέχεια.

3.2.1 Αλγόριθμος 2- opt

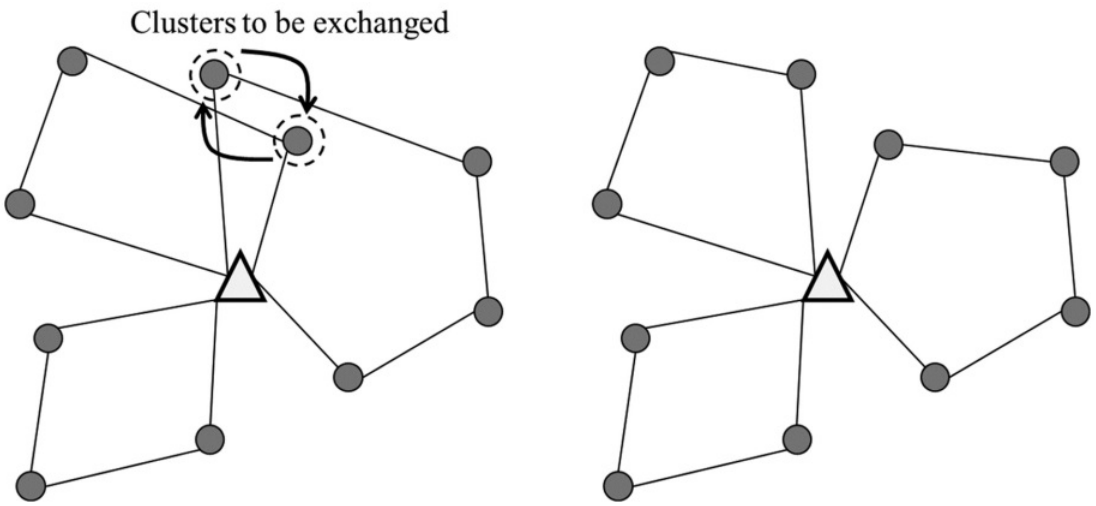
Ο 2-opt είναι ένας αλγόριθμος τοπικής αναζήτησης ο οποίος προτάθηκε από τον Croes το 1958. Η συγκεκριμένη μέθοδος αποτελείτε γενικά από την διαγραφή δύο ακμών και την επανασύνδεσή δύο μονοπατιών με διαφορετικό τρόπο για να καθορίσουμε μια καινούργια διαδρομή. Η κύρια ιδέα του αλγορίθμου, είναι η κατάλληλη διαμόρφωση των διαδρομών ώστε να μην υπάρχουν διασταυρώσεις. Τέλος , ο συγκεκριμένος αλγόριθμος μπορεί στην χειρότερη περίπτωση να μας εγγυηθεί ότι μια κίνηση βελτίωσης μειώνει το κόστος της διαδρομής τουλάχιστον κατά μια μονάδα.



Σχημα 2: παράδειγμα 2-opt (πηγη: http://www.devx.com/supportitems/showSupportItem.php?co=33574&supportitem=figure3)

3.2.2 Αλγόριθμος 1-1 Exchange

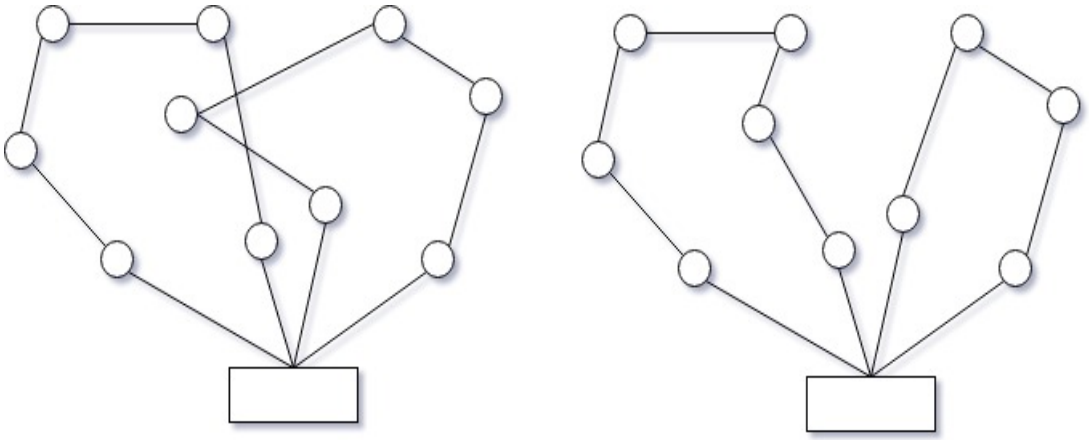
Ο 1-1 exchange είναι ένας αλγόριθμος τοπικής αναζήτησης ο όποιος προτάθηκε από τον Waters. Η κύρια διαφορά του με τον αλγόριθμο 2-opt είναι ότι ο 1-1 exchange χρειάζεται δύο διαδρομές για να μπορεί να εφαρμοστεί ενώ ο 2- opt μια. Η λογική του είναι σχετικά απλή, αρχικά γίνεται μια ταυτόχρονη ανταλλαγή 2 πελατών που βρίσκονται σε διαφορετικές διαδρομές και αν προκύψει μικρότερο κόστος μετά την ανταλλαγή, αυτή αποθηκεύεται.



Σχήμα 3: παράδειγμα 1-1exchange (πηγη: http://www.scielo.br/scielo.php?script=sci_arttext&pid=S0104-530X2016005004104&lng=en&nrm=iso&tlng=en)

3.2.3 Αλγόριθμος 1-0 Relocate

Ο 1-0 relocate είναι ένας αλγόριθμος τοπικής αναζήτησης ο όποιος προτάθηκε από τον Waters. Κύριος στόχος του συγκεκριμένου αλγορίθμου είναι η δημιουργία ομοιόμορφων διαδρομών καθώς και η διαγραφή των περιττών διαδρομών. Η λογική του είναι απλή, διαγράφεται ένας πελάτης από μια διαδρομή και επανατοποθετείτε σε μια άλλη με μικρότερο κόστος.



Σχήμα 4: παράδειγμα 1-0 relocate

3.3 Μεθευρετικοί Αλγόριθμοι

Οι μεθευρετικοί αλγόριθμοι είναι μέθοδοι επίλυσης που συνδυάζουν διαδικασίες τοπικής αναζήτησης και υψηλότερου επιπέδου στρατηγικές για να δημιουργήσουν μια διαδικασία που είναι ικανή να ξεφύγει από κάποιο τοπικό ελάχιστο. Στη συγκεκριμένη κατηγορία ανήκει και ο αλγόριθμος: διαδικασία άπληστης τυποποιημένης προσαρμοστικής αναζήτησης (greedy randomized adaptive search procedure –GRASP) τον οποίο θα δούμε στην συνέχεια. Αξίζει να σημειωθεί ότι τα τελευταία χρόνια οι περισσότεροι αλγόριθμοι που έχουν αναπτυχθεί για την επίλυση προβλημάτων συνδυαστικής βελτιστοποιήσεις ανήκουν στην συγκεκριμένη κατηγορία.

Οι μεθευρετικοί αλγόριθμοι συνήθως χρησιμοποιούν πιο παραδοσιακούς ευρετικούς αλγορίθμους, σαν υποδιαδικασίες τους. Πολλές φορές για να αποφευχθεί κάποιο τοπικό ελάχιστο και για να βρουν καλύτερη ολική λύση, κάποιοι μεθευρετικοι αλγόριθμοι επιτρέπουν βήματα που οδηγούν σε μη εφικτή ενδιάμεση λύση σε κάποιο βήμα του αλγορίθμου τους. Ένα επιπλέον χαρακτηριστικό αυτών των αλγορίθμων είναι ότι προσωμοιάζουν μια διαδικασία που συνήθως έχει εφαρμοστεί στην φύση. Τα στοιχεία που χρησιμοποιούν αυτοί οι αλγόριθμοι και μεταφορικά τα παρατηρούμε και στην φύση και είναι τα εξής:

- Χρησιμοποιούν ένα αριθμό από επαναληπτικές δοκιμές
- Περιλαμβάνουν ένα ή περισσότερους πράκτορες (νευρώνες μόρια μυρμήγκια)
- Λειτουργούν (στην περίπτωση των πολύ – πρακτόρων) με βάση ενός μηχανισμού συνεργασίας και ανταγωνισμού
- Περιλαμβάνουν διαδικασίες αυτό – τροποποιήσεων των ευρετικών παραμέτρων ή ακόμα και της αναπαράστασης του προβλήματος

Τα χαρακτηριστικά των μεθευρετικών αλγορίθμων είναι :

1. Μοντελοποιούν ένα φαινόμενο που υπάρχει στη φύση
2. Μπορούν να μεταφερθούν εύκολα σε παράλληλη μορφή
3. Είναι προσαρμοστικοί αλγόριθμοι

3.3.1 Η Διαδικασία Άπληστης Τυποποιημένης Προσαρμοστικής Αναζήτησης (Greedy Randomized Adaptive Search Procedure –GRASP)

Η διαδικασία άπληστης τυποποιημένης προσαρμοστικής αναζήτησης είναι μια επαναληπτική διαδικασία για την εύρεση προσεγγιστικών λύσεων σε προβλήματα συνδυαστικής βελτιστοποίησης. Η συγκεκριμένη τεχνική παρέχει σε κάθε επανάληψη μια εφικτή λύση και οι επαναλήψεις σταματούν όταν κάποιο κριτήριο σταματήσει να ικανοποιείται. Το τελικό αποτέλεσμα του αλγορίθμου είναι η καλύτερη λύση που βρέθηκε από όλες τις επαναλήψεις.

Κάθε επανάληψη αποτελείται από δυο φάσεις, μια φάση κατασκευής όπου μια τυχοποιημένη συνάρτηση απληστίας χρησιμοποιείται για να κατασκευάσει μια αρχική λύση και μια διαδικασία τοπικής αναζήτησης στην οποία βελτιώνεται η αρχική λύση. Στην φάση της κατασκευής προστίθεται επαναληπτικά ένα στοιχείο στη μη ολοκληρωμένη λύση, το κάθε στοιχείο επιλέγεται τυχαία από μια λίστα υποψηφίων, που ονομάζεται λίστα περιορισμού των υποψηφίων, στην οποία κάθε στοιχείο κατατάσσεται βάση μιας συνάρτησης απληστίας. Η συγκεκριμένη λύση δεν εγγυάται ότι είναι τοπικό ελάχιστο και για τον λόγο αυτό είναι απαραίτητη η φάση της τοπικής αναζήτησης, η οποία μας παράγει μια λύση που περιέχει τοπικό ελάχιστο. Στη φάση της τοπικής αναζήτησης καθορίζεται μια συνάρτηση που κάνει αναζήτηση στη γειτονία της αρχικής λύσης. Όπως είναι λογικό διαφορετικά προβλήματα χρειάζονται διαφορετικές συναρτήσεις απληστίας και διαφορετικές διαδικασίες τοπικής αναζήτησης για την εύρεση τοπικού ελαχίστου. Όταν αυτά καθοριστούν σωστά το μόνο που μας ενδιαφέρει για να εξετάσουμε τον αλγόριθμο είναι το μέγεθος της λίστας και ο αριθμός των επαναλήψεων που χρειάζεται.

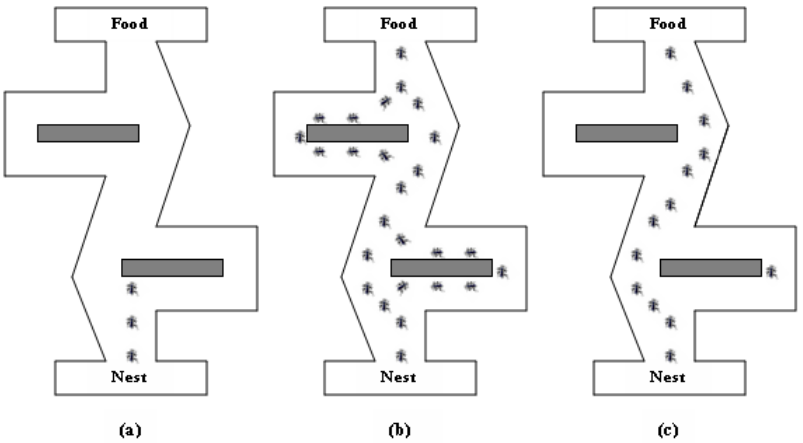
3.4 Εξελικτικές Στρατηγικές και Γενετικοί Αλγόριθμοι

Η ιδέα της μίμησης στην διαδικασίας της εξέλιξης, στην αναζήτηση για καλύτερες λύσεις οδήγησαν στην ανάπτυξη του πεδίου των γενετικών αλγορίθμων. Ο πρώτος γενετικός αλγόριθμος αναπτύχθηκε από τον John H. Holland την δεκαετία του 1960. Οι γενετικοί αλγόριθμοι μιμούνται τη διαδικασία της εξέλιξης στην φύση. Έτσι, σε αντίθεση με τους περισσότερους ευρετικούς αλγορίθμους, οι γενετικοί αλγόριθμοι κατά τη διάρκεια της αναζήτησης μια καλύτερης λύσης, χρησιμοποιούν πληροφορίες από ένα πληθυσμό λύσεων, όπου ονομάζονται άτομα. Ένας γενετικός αλγόριθμος, είναι μια στοχαστική επαναληπτική διαδικασία, που έχει την δυνατότητα, να διατηρεί το μέγεθος του πληθυσμού του σταθερό σε κάθε επανάληψη. Η βασική τους λειτουργία είναι το ταίριασμα των δυο λύσεων, με στόχο να δημιουργηθεί μια νέα λύση. Στην κατηγορία των εξελικτικών αλγόριθμων και ιδιαίτερα στην υποκατηγορία των αλγορίθμων νοημοσύνης σμήνους ανήκει και ο Αλγόριθμος Βελτιστοποίησης Αποικίας Μυρμηγκιών (Ant colony optimization - ACO) που θα αναλυθεί στη συνέχεια.

3.4.1 Αλγόριθμος Βελτιστοποίησης Αποικίας Μυρμηγκιών (Ant Colony Optimization- ACO)

Η βελτιστοποίηση αποικίας μυρμηγκιών (Ant colony optimization- ACO), είναι μια σχετικά νέα στρατηγική, η όποια προσπαθεί να μοντελοποίησει τη συμπεριφορά των πραγματικών μυρμηγκιών στον τρόπο εύρεσης τις τροφής τους, και με παρόμοιο τρόπο να λύσει προβλήματα συνδυαστικής βελτιστοποίησης. Τα μυρμήγκια αναπτύσσουν μια τεχνική για την εύρεση της συντομότερης διαδρομής, από την φωλιά τους προς την τροφή τους. Τα μυρμήγκια ξεκινούν για την εύρεση της τροφής τους με τυχαίο τρόπο και καθώς κινούνται αφήνουν μια ποσότητα, μιας ουσίας που ονομάζεται φερομόνη, και με αυτόν τον τρόπο

μαρκάρουν το μονοπάτι που έχουν διανύσει. Η ποσότητα της φερομόνης που αφήνει το κάθε μυρμήγκι εξαρτάται από την απόσταση, την ποιότητα και την ποσότητα της τροφής. Το επόμενο μυρμήγκι που θα φύγει από την φωλιά του, είναι πιθανό να ακολουθήσει τη φερομόνη που υπάρχει σε ένα μονοπάτι, αφήνοντας και αυτό με την σειρά του μια ποσότητα φερομόνης στο ίδιο μονοπάτι. Καθώς η ποσότητα τις φερομόνης αυξάνεται στη συγκεκριμένη διαδρομή, όλο και περισσότερα μυρμηγκιά θα ακολουθήσουν το συγκεκριμένο μονοπάτι. Η φερομόνη καθώς περνά η ώρα, ιδιαίτερα στα μονοπάτια που δεν πηγαίνουν πολλά μυρμήγκια, ελαττώνεται. Τελικά από όλα τα υπόλοιπα μονοπάτια, η φερομόνη εξατμίζεται και όλα τα μυρμήγκια ακολουθούν το ίδιο μονοπάτι, το οποίο μονοπάτι είναι και η βέλτιστη ή σχεδόν – βέλτιστη λύση.



Σχημα 5: Εύρεση συντομότερης διαδρομής μυρμηγκιών (πηγη: http://article.sapub.org/10.5923.j.eee.20120204.09.html)

Στο πρόβλημα δρομολόγησης οχημάτων το κάθε μυρμήγκι – λύση έχει στη διάθεση του 2 πληροφορίες. Η πρώτη πληροφορία είναι η ποσότητα της φερομόνης η οποία υπάρχει στα τόξα που είναι πιθανά να επιλεγούν, και συμβολίζεται με τ_{ij} , ενώ η δεύτερη πληροφορία είναι η ευρετικη πληροφορία $n_{ij}=1/c_{ij}$ όπου, c_{ij} είναι η απόσταση από τον κόμβο i στον κόμβο j. Σε περίπτωση που το c_{ij} είναι μηδέν, δίνεται στη ευρετικη συνάρτηση μια πολύ μικρή τιμή.

Η αρχική φερομόνη που θα τοποθετηθεί πάνω στο γράφημα μπορεί να υπολογιστεί από την σχέση τ_{ij} = m/TC, όπου m το πλήθος των μυρμηγκιών και TC είναι το κόστος μιας αρχικής λύσης που έχει κατασκευαστεί με απλούστερο τρόπο.

Σε κάθε βήμα κατασκευής της διαδρομής, το κάθε μυρμήγκι εφαρμόζει έναν πιθανοτικό κανόνα για να επιλέξει σε ποιο πελάτη θα πάει στην συνέχεια. Το κάθε μυρμήγκι που βρίσκεται στον πελάτη i, επιλέγει αν θα πάει ή όχι στον πελάτη j, με βάση την πιθανότητα

$$\mathbf{p_{ij}} = \frac{[\mathbf{t_{ij}}]^{\mathbf{a}}[\mathbf{n_{ij}}]^{\mathbf{b}}}{\sum_{\mathbf{l=1}}^{\mathbf{M}}[\mathbf{t_{ij}}]^{\mathbf{a}}[\mathbf{n_{ij}}]^{\mathbf{b}}}$$

Όπου Μ είναι το σύνολο των πελατών και α, β δύο παράμετροι που καθορίζουν αν επηρεάζει περισσότερο την επιλογή η φερομόνη που βρίσκεται πάνω στο τόξο ή η ευρετική πληροφορία που έχει υπολογιστεί για κάθε τόξο. Αν το α=0 τότε οι κοντινότεροι πελάτες είναι πιθανόν να επιλεγούν, ενώ εάν το β=0 τότε χρησιμοποιείται μόνο η φερομόνη.

Η ποσότητα της φερομόνης που υπάρχει σε κάθε τόξο ενημερώνεται και μια καινούργια ποσότητα προστίθεται στα τόξα που χρησιμοποιήθηκαν από τα μυρμήγκια. Η φερομόνη μειώνεται σε όλα τα τόξα κατά μια ποσότητα $\tau_{ij}^{new}=(1-\rho)\tau_{ij}^{old}$ όπου $0<\rho<1$ είναι ο συντελεστής εξάτμισης και χρησιμοποιείται για να εξασφαλίσει ότι ο αλγόριθμος θα ξεχάσει κακές αποφάσεις από προηγούμενες επαναλήψεις. Στα τόξα που δεν επιλέγονται η φερομόνη μειώνεται εκθετικά μέχρι να πλησιάσει κοντά στο μηδέν. Στα τόξα που έχουν χρησιμοποιηθεί, η φερομόνη αυξάνεται κατά $\tau_{ij}= \tau_{ij}+\sum_{k=1}^{m} \Delta\tau_{ij}^{k}$,όπου m είναι το πλήθος των μυρμηγκιών και $\Delta\tau_{ij}$ είναι η ποσότητα της φερομόνης που αφήνει το κ μυρμήγκι από τα τόξα που πέρασε και ισχύει $\Delta\tau_{ij}=\frac{1}{c}$, όπου c το συνολικό κόστος.

Κεφάλαιο 4 - Περιγραφή Προβλήματος και Αλγορίθμου

Στην παρούσα διπλωματική εργασία ασχολούμαστε με το ανοιχτό κλειστό πρόβλημα δρομολόγησης οχημάτων. Σε αυτό το κεφάλαιο θα γίνει μια αναλυτική περιγραφή του κώδικα που υλοποιήθηκε και στη συνέχεια χρησιμοποιήθηκε για την εύρεση των βέλτιστων διαδρομών. Ο κώδικας χωρίζεται σε δύο στάδια, το πρώτο στάδιο έχει σαν στόχο την δημιουργία μιας αρχικής εφικτής λύσης, με τον αλγόριθμο του grasp και την περαίτερω βελτίωση της με την χρήση αλγορίθμου τοπικής αναζήτησης. Ενώ στο δεύτερο και κυριότερο στάδιο του κώδικα, με την χρήση του αλγορίθμου Ant Colony Optimization και τριών τοπικών αναζητήσεων, βρίσκεται η βέλτιστη λύση του προβλήματος.

4.1 Δεδομένα και Επεξεργασία

Για την λύση του συγκεκριμένου προβλήματος απαραίτητα είναι τα παρακάτω δεδομένα : το πλήθος πελατών, οι γεωγραφικές συντεταγμένες των πελατών, η ζήτησή τους, ο αριθμός των ιδιόκτητων οχημάτων, η μέγιστη χωρητικότητα τους και ο μέγιστος διαθέσιμος χρόνος τους. Λόγο του ότι δεν υπάρχουν ειδικά δεδομένα για το συγκεκριμένο πρόβλημα, τροποποιήθηκαν κατάλληλα τα δεδομένα, από τα CVRP και DCVRP. Τα δεδομένα που χρησιμοποιήθηκαν είναι τα : (Augerat et al.,1995) data set A,B,P, Christofides and Eilon data set E for CVRP και Daniele Vigo data set D for DCVRP. Αρχικά τα δεδομένα αποθηκεύτηκαν σε ένα αρχείο excel, και στην συνέχεια εισήχθηκαν στην matlab. Επιπλέον θεωρούμε τις αποστάσεις μεταξύ των κόμβων ως ευθείες, και για τον λόγο αυτό χρησιμοποιήθηκε ο τύπος της ευκλείδειας απόστασης για τον υπολογισμού της απόστασης μεταξύ δύο πελατών.

$$\sqrt{(X_1 - X_2)^2 - (Y_1 - Y_2)^2}$$ **,όπου (X_i, Y_i) η συντεταγμένη του σημείου i.**

Στην συνέχεια οι αποστάσεις αποθηκεύτηκαν σε έναν πίνακα με διαστάσεις NxN, όπου N το πλήθος των πελατών. Ο συγκεκριμένος πίνακας είναι συμμετρικός καθώς η μετάβαση από τον πελάτη i στον πελάτη j, ισούται με την μετάβαση από τον πελάτη j στον πελάτη i. Το σημείο (i,j) μας δείχνει τον χρόνο μετάβασης από τον πελάτη i στον πελάτη j, επιπλέον θεωρούμε ότι ο πρώτος πελάτης είναι η αποθήκη, και επομένως η ζήτησή της είναι μηδέν. Λόγω του ότι, το συγκεκριμένο πρόβλημα δεν είναι ευρέως διαδεδομένο, τα data set που χρησιμοποιήθηκαν ήταν φτιαγμένα για το Πρόβλημα Δρομολόγησης Οχημάτων Περιορισμένης Χωρητικότητας και επομένως δεν υπήρχε ο περιορισμός της μέγιστης απόστασης. Για να ξεπεραστεί το συγκεκριμένο εμπόδιο χρησιμοποιήθηκε η προσέγγιση των Ran Liu, Zhibin Jiang στη δημοσίευση (The close-open mixed vehicle routing problem) όπου η μέγιστη απόσταση που μπορεί να διανύσει το κάθε όχημα υπολογίστηκε από τον τύπο :

$$\textbf{maxRT= 0,2*2*}max_{i \in v}(c_{0i}) \textbf{ +0,8 *}r_{cw}\textbf{.}$$

Αξίζει να σημειωθεί ότι στα συγκεκριμένα παραδείγματα δεν λαμβάνεται υπόψη ο χρόνος εξυπηρέτησης των πελατών, και επίσης θεωρούμε ότι όλες οι μονάδες των μεταβλητών είναι ίδιες. Σε κάθε παράδειγμα ο αριθμός των ιδιόκτητων οχημάτων είναι δεδομένος, ενώ θεωρούμε ότι μπορούν να χρησιμοποιηθούν άπειρα ενοικιαζόμενα οχήματα.

Τέλος μετά των σχολιασμό των δεδομένων του προβλήματος αναλύονται και οι περιορισμοί που το διέπουν.

Οι περιορισμοί είναι οι εξής:

1. rt≤ maxRT : η μεταβλητή rt είναι ένας μετρητής ο οποίος αθροίζει τις συνολικές αποστάσεις (ή τον συνολικό χρόνο) που το όχημα έχει διανύσει. Να σημειωθεί ότι ο συγκεκριμένος περιορισμός τροποποιείται όταν το όχημα είναι ιδιόκτητο, καθώς πρέπει να ληφθεί υπόψη και η απόσταση (ή χρόνος) επιστροφής του στην αποθήκη. Σε περίπτωση που δεν παραβιαστεί ο συγκεκριμένος περιορισμός αλλά το όχημα επιστρέψει στην αποθήκη, η μεταβλητή rt δεν μηδενίζεται. Με αυτόν τον τρόπο δίνεται η δυνατότητα στο ίδιο όχημα να ξεκινήσει νέα διαδρομή. Ελαχιστοποιώντας έτσι τον συνολικό αριθμό οχημάτων.
2. q ≤Qmax : η μεταβλητή q είναι ένας μετρητής ο οποίος αθροίζει τη συνολική ζήτηση που έχει εξυπηρετήσει το όχημα. Ο συγκεκριμένος περιορισμός χρησιμοποιείται για να εξασφαλίσει ότι ένα όχημα δεν θα εξυπηρετήσει περισσότερη ζήτηση από την συνολική του χωρητικότητα. Το όχημα αφού εξυπηρετήσει την συνολική ζήτηση που του ανατέθηκε έχει την δυνατότητα ή να επιστρέψει στην αποθήκη ή να τελείωσει εκεί την διαδρομή του.
3. Arithmosoximatos≤ N_u: Ο συγκεκριμένος περιορισμός μας εξασφαλίζει ότι δεν θα χρησιμοποιηθούν περισσότερα ιδιόκτητα οχήματα (N_u) από τον αριθμό που μας επιτρέπονται. Είναι αρκετά σημαντικός περιορισμός διότι τα ιδιόκτητα οχήματα έχουν μικρότερο κόστος από τα ενοικιαζόμενα, με αποτέλεσμα ο αλγόριθμος να τείνει να τα χρησιμοποιεί.

4.2 Τοπική Αναζήτηση

Όπως αναφέρθηκε και στο κεφάλαιο τρία, οι τοπικές αναζητήσεις που χρησιμοποιήθηκαν είναι η 2-opt, η 1-1 exchange και η 1-0 relocate. Λόγω του ότι, χρησιμοποιούνται σε διαφορετικά σημεία στον κώδικα, οι τοπικές αναζητήσεις γράφτηκαν σαν συναρτήσεις. Τα βασικά ορίσματα που έχουν είναι: οι διαδρομές, τα κόστη κάθε διαδρομής οι χρόνοι εκτέλεσης των διαδρομών καθώς και η ζήτηση που καλύπτουν σε κάθε διαδρομή. Ο 2-opt εφαρμόζονταν σε κάθε λύση, ενώ η 1-1 exchange και η 1-0 relocate εφαρμόζονταν μόνο στις καλύτερες λύσης και ο κυρίως λόγος που γίνεται αυτό είναι ότι οι 1-1 exchange, 1-0 relocate είναι πιο στοχευόμενοι.

- Στον 2-opt για δεδομένο αριθμό επαναλήψεων σε κάθε διαδρομή, επιλέγονται τυχαία δύο πελάτες και ανταλλάζουν τη θέση τους, ελέγχονται οι περιορισμοί και

εφόσον δεν παραβιάζονται, υπολογίζετε το κόστος της νέας διαδρομής. Εάν είναι μικρότερο το κόστος της καινούργιας διαδρομής από το προηγούμενο κόστος, τότε αποθηκεύεται η συγκεκριμένη αλλαγή και ενημερώνονται οι κατάλληλες μεταβλητές.

- Στον 1-1 exchange για δεδομένο αριθμό επαναλήψεων επιλέγονται τυχαία δύο πελάτες από διαφορετικές διαδρομές και ανταλλάζουν θέση. Ελέγχονται οι περιορισμοί του προβλήματος και εφόσον δεν παραβιάζονται οι περιορισμοί, εξετάζεται το κόστος. Εάν προκύψει συνολικά μικρότερο κόστος, δηλαδή αν το άθροισμα των κοστών, των νέων διαδρομών είναι μικρότερο από των παλιών, τότε αποθηκεύονται οι καινούργιες διαδρομές. Υπάρχει όμως η πιθανότητα ο αλγόριθμος να παγιδευτεί σε τοπικό ελάχιστο, και για τον λόγο αυτό, αν μετά από έναν αριθμό επαναλήψεων δεν υπάρχει σημαντική βελτίωση της λύσης, ο αλγόριθμος τερματίζει.
- 1-0 relocate εφαρμόστηκε κυρίως στα προβλήματα όπου ο αλγόριθμος δεν μπόρεσε να επιτύχει τον ελάχιστο αριθμό διαδρομών. Ο βασικός στόχος του συγκεκριμένου αλγορίθμου ήταν η μείωση των συνολικών διαδρομών. Ο τρόπος επιλογής των διαδρομών από τις οποίες θα φύγουν οι πελάτες είναι απλός. Αρχικά προσπαθεί να διαγράψει την τελευταία διαδρομή, στην συνέχεια διαγράφει την διαδρομή με τους λιγότερους πελάτες και τέλος επανατοποθετεί τυχαία πελάτες με στόχο την περαιτέρω μείωση του κόστους. Η διαδρομή στην όποια θα εισαχθεί ο πελάτης επιλέγεται τυχαία. Επίσης ο πελάτης που θα επιλεγεί καθώς και η θέση του στην νέα διαδρομή, που θα εισήχθει επιλέγονται τυχαία. Τέλος ελέγχονται οι περιορισμοί του προβλήματος και εφόσον δεν παραβιάζονται, υπολογίζεται το άθροισμα των κοστών των νέων διάδρομων, και στην περίπτωση που προκύψει μικρότερο κόστος αποθηκεύονται οι αλλαγές και ενημερώνονται οι μεταβλητές.

4.3 Δημιουργία Αρχικών Λύσεων με GRASP

Για την γρηγορότερη σύγκληση του αλγορίθμου Ant Colony Optimization χρειάζεται να τοποθετηθεί στο γράφημα η αρχική ποσότητα φερομόνης Για την σωστή τοποθέτηση της φερομόνης χρησιμοποιήθηκε ο αλγόριθμος GRASP. Επιπλέων για την δημιουργία της αρχικής λύσης ο αλγόριθμος επιλέγει τυχαία από ένα σύνολο κόμβων, το οποίο ονομάζεται λίστα περιορισμού των υποψηφίων, τον επόμενο κόμβο που θα επισκεφτεί. Σημαντικό είναι να επιλεγεί το κατάλληλο μέγεθος για τη λίστα περιορισμού των υποψηφίων, καθώς αν επιλεγεί ή πολύ μεγάλο μέγεθος ή πολύ μικρό, αυτό συνήθως δεν οδηγεί σε τοπικό ελάχιστο. Στην συνέχεια, αφού ο αλγόριθμος βεβαιωθεί, ότι δεν παραβιάζεται κανένας από τους περιορισμούς, τοποθετεί τον κόμβο στην διαδρομή. Η διαδικασία αυτή επαναλαμβάνεται έως ότου εξυπηρετηθεί και ο τελευταίως πελάτης. Τέλος για περαιτέρω βελτίωση της τελικής λύσης, εφαρμόζεται η τοπική αναζήτηση 2-opt και στη συνέχεια τοποθετείται κατάλληλα η φερομόνη στο γράφημα. Το γράφημα αντιπροσωπεύεται από έναν πινάκα NxN, όπου το N είναι το πλήθος των κόμβων και ονομάζεται fer. Στο σημείο i,j

αποθηκεύεται η φερομόνη του τόξου i,j και ισχύει ότι fer(i,j)= fer(j,i). Η παραπάνω διαδικασία πραγματοποιείται αρκετές φορές (ο αριθμός των επαναλήψεων της, εξαρτάται από το εκάστοτε παράδειγμα) και έχει σαν στόχο την τοποθέτηση σχετικά μεγάλης ποσότητας φερομόνης στα «καλά» τόξα.

4.4 Υλοποίηση Ant Colony Optimization

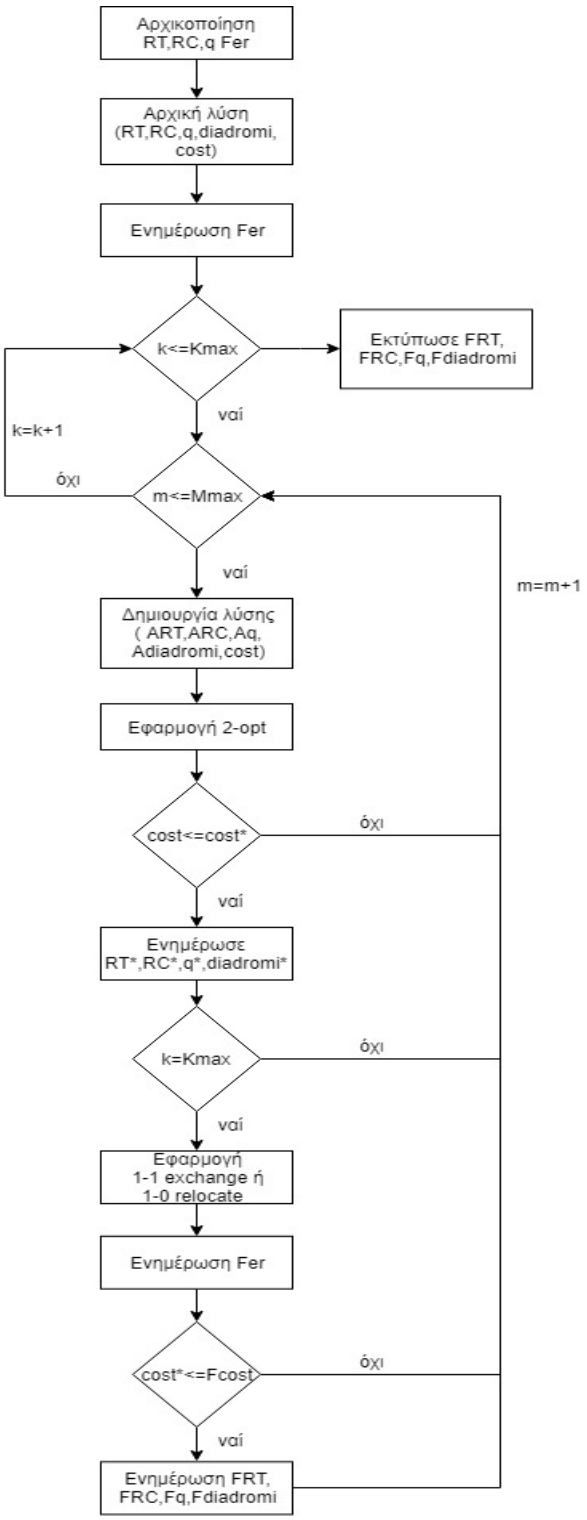
Για την εύρεση της βέλτιστης λύσης χρησιμοποιήθηκε ο αλγόριθμος Ant Colony Optimization (ACO), ο συγκεκριμένος αλγόριθμος για να λειτουργήσει χρειάζεται εκτός από τις αποστάσεις μεταξύ των κόμβων, την ζήτηση των πελατών, τη μέγιστη χωρητικότητα και τη μέγιστη απόσταση των οχημάτων, ενώ επίσης χρειάζεται και τον πίνακα με την αρχική ποσότητα τις φερομόνης. Οι παράμετροι του ACO είναι ο αριθμός των μυρμηγκιών, ο ρυθμός εξάτμισης της φερομόνης και οι δύο μεταβλητές α, β όπου α είναι το ποσοστό χρησιμοποίησης της φερομόνης και το β το ποσοστό χρησιμοποίησης του κόστους. Οι συγκεκριμένες μεταβλητές βοηθούν στην εύρεση του επόμενου πελάτη που θα τοποθετηθεί στην διαδρομή. Οι παραπάνω παράμετροι θα αναλυθούν εκτενέστερα στην συνέχεια. Αρχικά αρχικοποιούνται οι απαραίτητες μεταβλητές και οι απαραίτητοι πίνακες, για την εκτέλεση του αλγορίθμου. Στη συνέχεια όλοι οι κόμβοι που δεν τους έχει επισκεφτεί ο αλγόριθμος, μαζί με τις αντίστοιχες πιθανότητες επιλογής τους (οι οποίες εξαρτώνται από τα α, β) τοποθετούνται σε ένα διάνυσμα όπου με τυχαίο τρόπο επιλέγεται ο επόμενος πιθανός κόμβος. Αν ο συγκεκριμένος κόμβος δεν παραβιάζει κανένα από τους περιορισμούς, τοποθετείται στην διαδρομή. Αν παραβιάζεται μόνο η χωρητικότητα του οχήματος, τότε το όχημα επιστρέφει στην αποθήκη, για να ξεκινήσει νέα διαδρομή. Αντιθέτως αν παραβιάζονται και οι δυο περιορισμοί (διαθέσιμος χρόνος και χωρητικότητα), τότε το όχημα σταματά στον τελευταίο πελάτη και ξεκινά νέα διαδρομή με καινούργιο όχημα. Αυτή η διαδικασία επαναλαμβάνεται μέχρι να εξυπηρετηθεί και ο τελευταίος πελάτης. Στη λύση που προκύπτει γίνεται η τοπική αναζήτηση 2-opt και θεωρούμε ότι είναι η λύση του ενός μυρμηγκιού. Για την ενημέρωση της φερομόνης του γραφήματος, κρατάμε την καλύτερη λύση (δηλαδή την ελάχιστη) ανά είκοσι μυρμήγκια, στην οποία λύση, εφαρμόζεται περαιτέρω τοπική αναζήτηση 2-opt και 1-1 exchange ή 1-0 relocate. Η διαδικασία αυτή επαναλαμβάνεται μέχρι να εξαντληθεί ο συνολικός αριθμός των μυρμηγκιών, που αρχικά έχουμε ορίσει. Κάθε φορά που προστίθεται φερομόνη στο γράφημα, ένα ποσοστό της εξατμίζεται με ρυθμό 1-ρ. Με αυτόν τον τρόπο ο αλγόριθμος καταφέρνει να ξεπερνά τοπικά ελάχιστα.

4.5 Διάγραμμα Ροής

Παρακάτω παρουσιάζεται το διάγραμμα ροής του αλγορίθμου Ant Colony Optimization .

- Όπου RT ,RC ,q, diadromi ,fer είναι το κόστος, ο χρόνος ,η ζήτηση , οι διαδρομές και η αρχική φερομόνη αντίστοιχα

- Κ είναι ο αριθμός των συνολικών μυρμηγκιών
- Μ ο αριθμός των επαναλήψεων κάθε μυρμηγκιού
- Όπου ART, ARC,Aq, Adiadromi, Acost η λύση κάθε μυρμηγκιού
- Όπου RT*,RC*,q*,diadromi* η καλύτερη λύση κάθε μυρμηγκιού
- Όπου FRT,FRC,Fq,Fdiadromi η καλύτερη λύση του αλγορίθμου.



Κεφάλαιο 5 Αποτελέσματα και Συμπεράσματα

Στο συγκεκριμένο κεφάλαιο θα αναλυθεί η διαδικασία που ακολουθήθηκε για την εύρεση των βέλτιστων αποτελεσμάτων του αλγορίθμου. Όπως αναφέρθηκε και στο κεφάλαιο δύο για την επίλυση του συγκεκριμένου προβλήματος εφαρμόστηκαν δύο παραλλαγές. Η μόνη διαφορά των παραλλαγών είναι ότι στην πρώτη τα οχήματα έχουν την δυνατότητα επιστροφής στην αποθήκη για επαναφόρτιση, με άλλα λόγια τα οχήματα μπορούν να κάνουν παραπάνω από μία διαδρομές. Ενώ στην δεύτερη περίπτωση το κάθε όχημα πρέπει να κάνει ακριβώς μία διαδρομή. Η δυσκολία του συγκεκριμένου προβλήματος είναι η εύρεση του συνδυασμού των μεταβλητών (αριθμός των μυρμηγκιών, ο ρυθμός εξάτμισης της φερομόνης και τα α, β) που οδηγούν στην βέλτιστη λύση. Για την εύρεση των τιμών των συγκεκριμένων μεταβλητών πραγματοποιήθηκε σε κάθε παράδειγμα ένα μεγάλο πλήθος επαναλήψεων, όπου σε κάθε μια άλλαζαν και οι τιμές των παραμέτρων.

Ο αλγόριθμος εφαρμόστηκε σε εικοσιένα (21) παραδείγματα που βρέθηκαν στην βιβλιογραφία και συγκρίθηκαν τα αποτελέσματά τους. Ο τρόπος γραφής των παραδειγμάτων είναι π.χ. A-N32-K5 όπου A είναι το data set που χρησιμοποιείται, N32 ο αριθμός των κόμβων του προβλήματος και K5 είναι ο ελάχιστος αριθμός των διαδρομών.

5.1 Διαδικασία Εύρεση Μεταβλητών και Παρουσίαση Τελικών Αποτελέσματών

Καθοριστικό ρόλο για την επιτυχία του αλγορίθμου έχει η αρχική λύση. Αυτό που παρατηρήθηκε μετά από ένα μεγάλο πλήθος επαναλήψεων είναι ότι πολύ καλές αρχικές λύσεις δεν οδηγούν πάντα τον αλγόριθμο στο βέλτιστο αποτέλεσμα. Αντιθέτως μια μέτρια λύση εξήγαγε πολύ καλύτερα τελικά αποτελέσματά. Αυτό οφείλεται κυρίως στο ότι όταν βρεθεί μια καλή λύση ο αλγόριθμος δυσκολεύεται στον εντοπισμό καλύτερης λύσης, για την εύρεση της αρχικής λύσης χρησιμοποιήθηκε ο αλγόριθμος του GRASP. Η μόνη μεταβλητή που επηρεάζει την λύση του είναι το μέγεθος της λίστας των υποψηφίων. Μετά από αρκετές δοκιμές συγκλίναμε στο ότι το καταλληλότερο μέγεθος της λίστας των υποψηφίων είναι το 10% των συνολικών κόμβων του εκάστοτε προβλήματος.

Για την βελτίωση της αρχικής λύσης του προβλήματος χρησιμοποιήθηκε ο αλγόριθμος Ant Colony Optimization. Αρχικά εφαρμόστηκε όπως παρουσιάζεται στο κεφάλαιο τρία αλλά μετά από ένα μικρό πλήθος επαναλήψεων έγινε αντιληπτό ότι τα αποτελέσματα που εξήγαγε δεν ήταν ικανοποιητικά και χρειάζονταν βελτίωση. Παρακάτω παρουσιάζονται όλες οι διαφοροποιήσεις που έγιναν με σκοπό την βελτίωση των τελικών αποτελεσμάτων.

Πρώτα από όλα για την ευκολότερη κατανόηση της ανάλυσης που θα ακολουθήσει, καλό θα είναι να αναφερθούμε στις μεταβλητές του συγκεκριμένου αλγορίθμου. Στο συγκεκριμένο πρόβλημα υπάρχουν τέσσερις μεταβλητές που επηρεάζουν την λύση. Η πρώτη και κύρια μεταβλητή είναι ο αριθμός των μυρμηγκιών που χρησιμοποιείται, με άλλα

λόγια είναι ο αριθμός των επαναλήψεων που θα εκτελέσει ο αλγόριθμος (για τα μικρά παραδείγματα μέχρι 100 κόμβους χρησιμοποιήθηκαν 250 μυρμήγκια). Επίσης δύο σημαντικές μεταβλητές είναι οι παράμετροι α, β οι οποίοι παίρνουν τιμές στο διάστημα [0,1] και ισχύει ότι α+β=1. Οι συγκεκριμένοι παράμετροι επηρεάζουν τον τρόπο επιλογής του επόμενου πελάτη, για μεγάλη τιμή της παραμέτρου α το μυρμήγκι επηρεάζεται περισσότερο από τη φερομόνη αντιθέτως, για μεγάλη τιμή της παραμέτρου β το μυρμήγκι επηρεάζεται περισσότερο από το κόστος της διαδρομής. Τέλος η τελευταία μεταβλητή του προβλήματος είναι ο ρυθμός εξάτμισης της φερομόνης (ρ), όσο μεγαλύτερη η τιμή της παραμέτρου ρ τόσο μεγαλύτερο είναι το ποσοστό τις φερομόνης που «εξατμίζεται».

Όπως αναφέρθηκε και παραπάνω η πρώτη περίπτωση που εφαρμόστηκε είναι αυτή που αναφέρθηκε στο κεφάλαιο τρία. Πολύ γρήγορα έγινε αντιληπτό ότι τα αποτελέσματα που μας εξήγαγε ο αλγόριθμος δεν ήταν τα αναμενόμενα, και ο κύριος λόγος ήταν ότι ο αλγόριθμος δεν κατόρθωνε να ξεφύγει από κακές λύσεις μυρμηγκιών. Ο τρόπος που ξεπεράστηκε η συγκεκριμένη δυσκολία είναι απλός, ο αλγόριθμος τροποποιήθηκε κατάλληλα έτσι ώστε αντί το κάθε μυρμήγκι να εκτελεί ακριβώς μια διαδρομή να εκτελεί παραπάνω (συνήθως 20-25). Στη συνέχεια επιλέγεται η καλύτερη λύση η οποία θεωρείται και η λύση του συγκεκριμένου μυρμηγκιού, οι άλλες λύσεις διαγράφονται.

Με την συγκεκριμένη τροποποίηση τα αποτελέσματα βελτιώθηκαν αισθητά, φυσικά οι αποκλείσεις από τις βέλτιστες λύσεις ήταν μεγάλες και ο αλγόριθμος χρειαζόταν περαιτέρω βελτίωση. Αυτό που παρατηρήθηκε, ήταν ότι το κόστος τον οχημάτων επηρεάζει σημαντικά τις τελικές λύσεις και πολλές φορές φτάνει το 50% του συνολικού κόστους. Επομένως για να πλησιάσουμε τις βέλτιστες λύσεις χρειάζεται ο αριθμός των οχημάτων να είναι ο ελάχιστος δυνατός. Στη προκειμένη περίπτωση ο αλγόριθμος αδυνατούσε να φτάσει τον ελάχιστο αριθμό οχημάτων και πολλές φορές χρησιμοποιούσε αρκετά μεγαλύτερο αριθμό οχημάτων. Η λύση στην συγκεκριμένη δυσκολία κρύβεται στις τοπικές αναζητήσεις και συγκεκριμένα στην 1-0 relocate, φυσικά πρέπει να τροποποιηθεί κατάλληλα για το συγκεκριμένο πρόβλημα.

Επιλέξαμε να χρησιμοποιήσουμε τον 1-0 relocate όχι όπως αναφέρεται στην βιβλιογραφία αλλά πιο στοχευμένα. Αρχικά προσπαθεί να διαγράψει την τελευταία διαδρομή τοποθετώντας, τους πελάτες της, σε διαφορετικές διαδρομές. Στην συνέχεια αφαιρεί την διαδρομή με τους λιγότερους πελάτες και τέλος επιλέγει τυχαία πελάτες για περαιτέρω μείωση του κόστους. Η διαδικασία αναφέρεται αναλυτικά στο υποκεφάλαιο 4.3. Μετά την εφαρμογή της τοπικής αναζήτησης 1-0 relocate παρατηρήθηκε σημαντική μείωση του κόστους στα παραδείγματα όπου προηγουμένως ο αλγόριθμος αδυνατούσε να βρει τον ελάχιστο αριθμό οχημάτων. Στα παραδείγματα με μικρό αριθμό πελατών, ο 1-0 relocate δεν ήταν πολύ αποδοτικός και ο κύριος λόγος είναι ότι ο αλγόριθμος είχε επιτύχει τον ελάχιστο αριθμό οχημάτων. Στα συγκεκριμένα παραδείγματα χρησιμοποιείται η τοπική αναζήτηση 1-1 exchange. Σε γενικές γραμμές στα παραδείγματα με πολλούς πελάτες εφαρμόζεται ο 1-0 relocate ενώ σε αυτά με μικρό αριθμό πελατών ο 1-1 exchange, φυσικά

υπάρχουν και οι περιπτώσεις που χρησιμοποιούνται και οι δύο τοπικές αναζητήσεις ταυτόχρονα.

Εφαρμόζωντας τις συγκεκριμένες αλλαγές στον αλγόριθμο παρατηρήθηκε μεγάλη βελτίωση και οι λύσεις άρχισαν να έχουν μικρές αποκλίσεις από τις βέλτιστες. Στην συνέχεια έγινε διερεύνηση για την εύρεση του συνδυασμού των μεταβλητών που ελαχιστοποιεί το κόστος. Ο αριθμός των μυρμηγκιών στα παραδείγματα με μικρότερο αριθμό πελατών από 281 ήταν σταθερός και ίσος με 250. Για την εύρεση του βέλτιστου συνδυασμού των υπολοίπων μεταβλητών χρειάστηκε να εφαρμοστούν όλοι οι πιθανοί συνδυασμοί στον αλγόριθμο. Λόγω της ύπαρξης τυχαιότητας στον αλγόριθμο και για να έχουμε όσο το δυνατόν πιο αντικειμενικά αποτελέσματα ο κάθε πιθανός συνδυασμός επαναλήφθηκε 5 φόρες. Όπως γίνεται αντιληπτό ο αριθμός των επαναλήψεων είναι αρκετά μεγάλος, συγκεκριμένα πεντακόσιες (500) για κάθε παράδειγμα. Τα αποτελέσματα παρουσιάζονται στους παρακάτω πίνακες.

Τα αποτελέσματα ήταν ικανοποιητικά, αλλά λαμβάνοντας υπόψιν τις δυνατότητες του αλγορίθμου Ant Colony Optimization, είναι σαφές ότι υπάρχει περιθώριο βελτίωσης. Παρατηρήθηκε ότι η καλύτερη λύση του αλγορίθμου εμφανιζόταν πολύ πριν τον τερματισμό του, συνήθως στην μέση των επαναλήψεων. Αυτό οφείλεται στον κορεσμό της φερομόνης του γραφήματος, καθώς αυξάνεται υπερβολικά σε κάποια τόξα με αποτέλεσμα όλα τα μυρμήγκια να ακολουθούν την ίδια διαδρομή.

Μια νέα στρατηγική που ακολουθήθηκε για να ξεπεραστεί η παραπάνω δυσκολία ήταν η εισαγωγή μιας νέας μεταβλητής που ονομάζεται εξάτμιση, η δουλεία της συγκεκριμένης μεταβλητής είναι κάθε φορά που ο αλγόριθμος βρίσκει μια καλύτερη λύση να «εξατμίζει» μια ποσότητα φερομόνης από το γράφημα. Με αυτόν τον τρόπο γίνεται προσπάθεια να αποφευχθεί ο γρήγορος κορεσμός της φερομόνης με τελικό σκοπό την βελτίωση των λύσεων. Η τελευταία στρατηγική που εφαρμόστηκε ήταν και η πιο αποδοτική, καθώς αντί να αρχικοποιούνται οι μεταβλητής α,β (επιλογή φερομόνης και επιλογή κόστους αντίστοιχα) στην αρχή του αλγορίθμου και να παραμένουν σταθερές καθ’ όλη την διάρκειά του, επιλέχτηκε να μεταβάλλονται δυναμικά σε κάθε επανάληψη. Ο τρόπος με τον οποίο επιτεύχθηκε η παραπάνω στρατηγική ήταν με την χρήση της εντολής rand οποία επιλέγει μια τιμή στο διάστημα [0,1] για την μεταβλητή της φερομόνης. Για την εύρεση της μεταβλητής του κόστους γίνεται η αφαίρεση β=1-α. Με την συγκεκριμένη στρατηγική επιτεύχθηκε μεγάλη βελτίωση των αποτελεσμάτων, καθώς καταφέραμε να έχουμε μηδενικές αποκλίσεις και σε αρκετές περιπτώσεις να βρούμε καλύτερα αποτελέσματα.

Σε γενικές γραμμές μπορούμε να πούμε ότι ο αλγόριθμος λειτουργεί καλύτερα για μεγάλες τιμές της παραμέτρου α ($0.7<\alpha<1$) και προφανώς για μικρές τιμές της παραμέτρου β καθώς ισχύει ότι β=1-α. Όσον αφορά τον ρυθμό εξάτμισης της φερομόνης δεν μπορούμε να συγκλίνουμε σε συγκεκριμένες τιμές. Τέλος παρατηρήθηκε ότι για μεγάλα παραδείγματα η μέθοδος χρειάζονταν περισσότερες επαναλήψεις για να προσεγγίσει το βέλτιστο αποτέλεσμα.

Στους παρακάτω 2 πίνακες παρουσιάζονται οι αρχικές λύσεις (μετά και την εφαρμογή της 1-0 relocate) καθώς και οι τελικές λύσεις των δύο παραλλαγών που εφαρμόστηκαν. Οι ποσότητες της φερομόνης, κόστους και ρυθμού εξάτμισης αναφέρονται στην αρχική λύση, καθώς στην τελική λύση μεταβάλλονται δυναμικά και δεν υπάρχει νόημα στην παρουσίαση τους.

Πρώτη παραλλαγή τα οχήματα εκτελούν πολλαπλές διαδρομές.

Παράδειγμα	Φερομόνη	Κόστος	Ρυθμός εξάτμισης	Αρχική λύση	Τελική λύση
A-n32-k5	0.9	0.1	0.9	1098.78	1024.612
A-n33-k5	0.8	0.2	1.0	1045.85	927.71
A-n33-k6	0.7	0.3	0.6	1180.28	1091.54
A-n34-k5	0.9	0.1	0.9	1162.74	1103.42
A-n36-k5	0.9	0.1	0.8	1118.24	1028.05
B-n50-k8	0.9	0.1	0.6	1839.08	1645.25
B-n51-k7	0.8	0.2	0.6	1591.72	1442.71
B-n52-k7	0.8	0.2	0.6	1199.30	1128.05
B-n56-k7	0.2	0.8	0.1	1187.08	1067.76
B-n57-k7	0.9	0.1	0.8	1674.63	1615.12
E-n22-k4	0.9	0.1	0.8	700.94	700.94
E-n30-k3	0.8	0.2	0.7	707.58	701.22
E-n33-k4	0.7	0.3	0.6	1087.84	1063.01
E-n51-k5	0.9	0.1	0.8	971.17	960.58
E-n76-k10	0.7	0.3	0.6	1720.97	1566.24
P-n21-k2	0.5	0.5	0.3	383.46	383.46
P-n22-k8	0.9	0.1	0.8	1089.77	975.13
D101-09c	0.8	0.2	0.6	1591.72	1266.17
D151-14b	0.9	0.1	1.0	2292.43	1745.94
D281-08k	0.8	0.2	0.8	12414.78	10710.40
D361-09k	0.9	0.1	0.8	15144.91	12837.36
Total gap				**5.66%**	

Πίνακας 1: αρχικά αποτελέσματα πρώτης παραλλαγής παραλλαγής

Δεύτερη παραλλαγή τα οχήματα εκτελούν μία διαδρομή.

Παράδειγμα	Φερομόνη	Κόστος	Ρυθμός εξάτμισης	Αρχική λύση	Τελική λύση
A-n32-k5	0.5	0.5	0.7	1134.08	1119.18
A-n33-k5	0.9	0.1	0.7	1074.11	1009.25
A-n33-k6	0.9	0.1	0.7	1247.94	1171.57
A-n34-k5	0.9	0.1	0.4	1131.18	1102.17
A-n36-k5	0.8	0.2	0.3	1194.74	1103.18
B-n50-k8	0.9	0.1	0.3	1776.66	1737.84
B-n51-k7	0.9	0.1	0.7	1645.53	1504.19
B-n52-k7	0.9	0.1	0.2	1323.31	1212.06
B-n56-k7	0.9	0.1	0.8	1270.48	1220.775
B-n57-k7	0.9	0.1	0.7	1698.82	1627.99
E-n22-k4	0.9	0.1	0.9	735.61	700.93
E-n30-k3	0.9	0.1	0.9	791.98	730.83
E-n33-k4	0.2	0.8	0.4	1075.13	1061.68
E-n51-k5	0.8	0.2	0.4	1192.77	935.05
E-n76-k10	0.9	0.1	0.1	1987.04	1765.60
P-n21-k2	0.9	0.1	0.1	388.74	383.46
P-n22-k8	0.4	0.6	0.1	1201.93	1183.87
D101-09c	0.9	0.1	0.1	1745.61	1529.04
D151-14b	0.8	0.2	0.7	2343.13	2239.37
D281-08k	0.8	0.2	0.7	12092.93	11054.54
D361-09k	0.8	0.2	0.7	15646.73	13585.86
Total gap				**9%**	

Πίνακας 2: αρχικά αποτελέσματα δεύτερης παραλλαγής

5.2 Παρουσίαση Αποτελεσμάτων

Στους παρακάτω πίνακες παρουσιάζονται αναλυτικά τα τελικά αποτελέσματα που εξήγαγε ο αλγόριθμος. Στην πρώτη στήλη βρίσκονται τα παραδείγματα και στην δεύτερη στήλη είναι τα αποτελέσματα που εξήγαγε ο solver CPLEX. Για να λειτουργήσει ο συγκεκριμένος solver χρειάζεται να του δοθεί ένα κάτω φράγμα (παράρτημα Α), ο solver σταμάτα μόλις φτάσει το κάτω φράγμα ή μέχρι να περάσει ένα συγκεκριμένο χρονικό διάστημα, στην

προκειμένη περίπτωση το χρονικό όριο είναι 48 ώρες. Στην 3 στήλη βρίσκεται η βέλτιστη λύση που υπάρχει στην βιβλιογραφία, ενώ στην τέταρτη τα οχήματα που χρησιμοποιήθηκαν στην BKS (ο πρώτος αριθμός είναι ο αριθμός των ιδιόκτητων οχημάτων, ενώ ο δεύτερος είναι ο συνολικός αριθμός των οχημάτων που χρησιμοποιήθηκαν). Στην πέμπτη στήλη τα βέλτιστα αποτελέσματα που μας εξήγαγε ο αλγόριθμος και στην έκτη τα οχήματα που χρησιμοποιήθηκαν στην λύση μας. Στην συνέχεια στην έβδομη στήλη η απόκλιση της λύσης μας από την βέλτιστη και τέλος στην όγδοη στήλη βρίσκεται ο μέσος χρόνος 5 επαναλήψεων που χρειάστηκε ο αλγόριθμος για να τρέξει.

Πρώτη παραλλαγή οχήματα που εκτελούν πολλαπλές διαδρομές.

Παράδειγμα	CPLEX	BKS	Οχήματα BKS	ACO	Οχήματα ACO	Gap(%)	Time (sec)
A-n32-k5	1063.53	1063.53	(3,5)	1024.612	(3,4)	-0.037	25.91
A-n33-k5	1011.15	994.91	(3,5)	927.71	(3,4)	-0.068	27.58
A-n33-k6	1171.35	1169.29	(3,6)	1091.54	(3,5)	-0.066	33.05
A-n34-k5	1109.61	1102.17	(3,5)	1103.42	(3,5)	0.001	29.99
A-n36-k5	1106.55	1096.35	(3,5)	1028.05	(3,4)	-0.062	37.05
B-n50-k8	1756.27	1739.95	(4,8)	1645.25	(4,6)	-0.054	48.62
B-n51-k7	1586.59	1494.54	(4,7)	1442.71	(4,6)	-0.035	49.67
B-n52-k7	1263.50	1202.7	(4,7)	1128.05	(4,5)	-0.062	50.00
B-n56-k7	1172.21	1168.22	(4,7)	1067.76	(4,5)	-0.086	59.04
B-n57-k7	1801.78	1549.4	(4,7)	1615.12	(4,7)	0.042	63.42
E-n22-k4	700.94a	700.93	(2,4)	700.93	(2,4)	0.000	19.95
E-n30-k3	733.24	730.83	(2,3)	701.22	(2,3)	-0.041	22.99
E-n33-k4	1076.90	1057.05	(2,4)	1063.01	(2,4)	0.006	25.76
E-n51-k5	933.37	921.28	(3,5)	960.58	(3,5)	0.043	45.00
E-n76-k10	1847.55	1750.26	(5,10)	1566.24	(5,8)	-0.105	103.70
P-n21-k2	383.46a	383.46a	(1,2)	383.46	(1,2)	0.000	15.55
P-n22-k8	1183.87	1183.87a	(5,8)	975.13	(5,8)	-0.176	30.50
D101-09c	1476.68	1476.33	(5,8)	1266.17	(4,5)	-0.139	158.54
D151-14b	2699.22	2057.53	(7,12)	1745.94	(7,9)	-0.150	288.18
D281-08k	10198.94	9041.52	(4,7)	10710.40	(4,8)	0.185	640.28
D361-09k	12208.27	11104.70	(5,8)	12837.36	(5,11)	0.160	1202.94
Average						**-3.1%**	

Πίνακας 3: αποτελέσματα πρώτης παραλλαγής

Δεύτερη παραλλαγή τα οχήματα εκτελούν μία διαδρομή.

Παράδειγμα	CPLEX	BKS	Οχήματα BKS	ACO	Οχήματα ACO	Gap(%)	Time (sec)
A-n32-k5	1063.53	1063.53	(3,5)	1119.18	(3,5)	0.052	20.47
A-n33-k5	1011.15	994.91	(3,5)	1009.25	(3,5)	0.014	21.70
A-n33-k6	1171.35	1169.29	(3,6)	1171.57	(3,6)	0.002	28.09
A-n34-k5	1109.61	1102.17	(3,5)	1102.17	(3,5)	0.000	27.05
A-n36-k5	1106.55	1096.35	(3,5)	1103.18	(3,5)	0.006	27.58
B-n50-k8	1756.27	1739.95	(4,8)	1737.84	(4,8)	-0.001	44.30
B-n51-k7	1586.59	1494.54	(4,7)	1504.19	(4,7)	0.006	45.26
B-n52-k7	1263.50	1202.7	(4,7)	1212.06	(4,7)	0.008	38.75
B-n56-k7	1172.21	1168.22	(4,7)	1220.775	(4,7)	0.045	44.38
B-n57-k7	1801.78	1549.4	(4,7)	1627.99	(4,7)	0.051	47.53
E-n22-k4	700.94a	700.93	(2,4)	700.93	(2,4)	0.000	19.81
E-n30-k3	733.24	730.83	(2,3)	730.83	(2,3)	0.000	21.98
E-n33-k4	1076.90	1057.05	(2,4)	1061.68	(2,4)	0.004	19.19
E-n51-k5	933.37	921.28	(3,5)	935.05	(3,5)	0.015	33.46
E-n76-k10	1847.55	1750.26	(5,10)	1765.60	(5,10)	0.009	74.03
P-n21-k2	383.46a	383.46a	(1,2)	383.46	(1,2)	0.000	15.80
P-n22-k8	1183.87	1183.87a	(5,8)	1183.87	(5,8)	0.000	27.70
D101-09c	1476.68	1476.33	(5,8)	1529.04	(5,8)	0.042	153.24
D151-14b	2699.22	2057.53	(7,12)	2239.37	(7,12)	0.088	258.69
D281-08k	10198.94	9041.52	(4,7)	11054.54	(4,7)	0.223	712.09
D361-09k	12208.27	11104.70	(5,8)	13585.86	(5,8)	0.223	1326.49
Average						**3.8%**	

Πίνακας 4: αποτελέσματα δεύτερης παραλλαγής

Εν κατακλείδι, τα αποτελέσματα που εξήχθησαν από τον αλγόριθμο είναι πολύ ικανοποιητικά και πληρούν σε μεγάλο βαθμό τον στόχο και τις απαιτήσεις της παρούσας διπλωματικής εργασίας. Και στις δύο παραλλαγές τα αποτελέσματα πλησίασαν αρκετά τα βέλτιστα και σε αρκετές περιπτώσεις τα ξεπέρασαν. Η συνολική απόκλιση στην πρώτη παραλλαγή ήταν -3,1% ενώ στην δεύτερη 3,8%. Καθοριστικό ρόλο στα αποτελέσματα είχε αναμφισβήτητα η εκτεταμένη αναζήτηση του βέλτιστου συνδυασμού των μεταβλητών, καθώς και η στοχευμένη τοπική αναζήτηση που εφαρμόστηκε. Όπως ήταν αναμενόμενο η πρώτη παραλλαγή εξήγαγε καλύτερα αποτελέσματα. Αυτό οφείλεται κυρίως στην δυνατότητα επιστροφής στην αποθήκη για επαναφόρτιση, με αποτέλεσμα να χρησιμοποιούνται λιγότερα οχήματα και να μειώνεται δραματικά το συνολικό κόστος των οχημάτων. Συνεπώς, γίνεται αντιληπτή η αξιοπιστία του αλγορίθμου ACO στην εύρεση βέλτιστων αποτελεσμάτων και επιπλέον είναι φανερό ότι η πρώτη παραλλαγή υπερτερεί έναντι της δεύτερης.

5.2 Γραφική Παρουσίαση Αποτελεσμάτων

Παρακάτω παρουσιάζονται γράφηκα κάποιες από τις καλύτερες λύσεις του αλγορίθμου του αλγορίθμου, σε κάθε παράδειγμα αναφέρονται οι τιμές των μεταβλητών, καθώς και η παραλλαγή που χρησιμοποιήθηκε για την εξαγωγή τα αποτελέσματα.

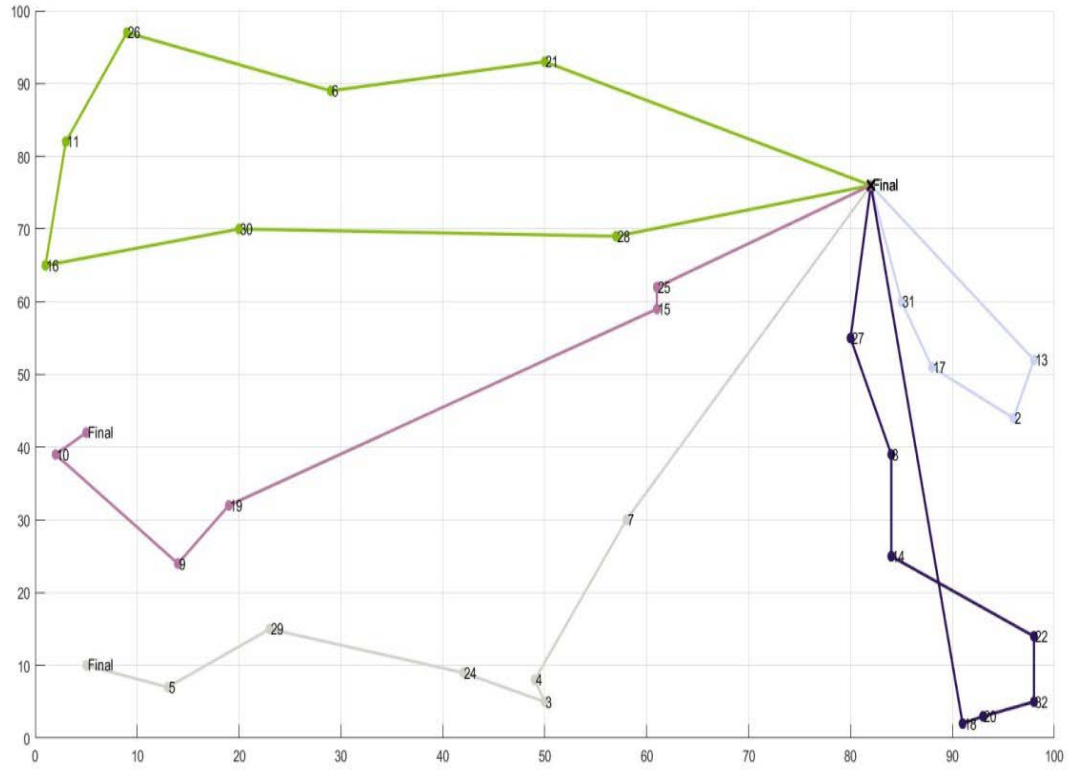
Παράδειγμα A-n32-k5

Δεύτερη παραλλαγή

Αριθμός μυρμηγκιών=250

eksatmisi=0.5

r=0.8



Διαδρομές:

1	21	6	26	11	16	30	28	1
1	31	17	2	13	1	0	0	0
1	27	8	14	22	32	20	18	1
1	7	4	3	24	29	5	12	0
1	25	15	19	9	10	23	0	0

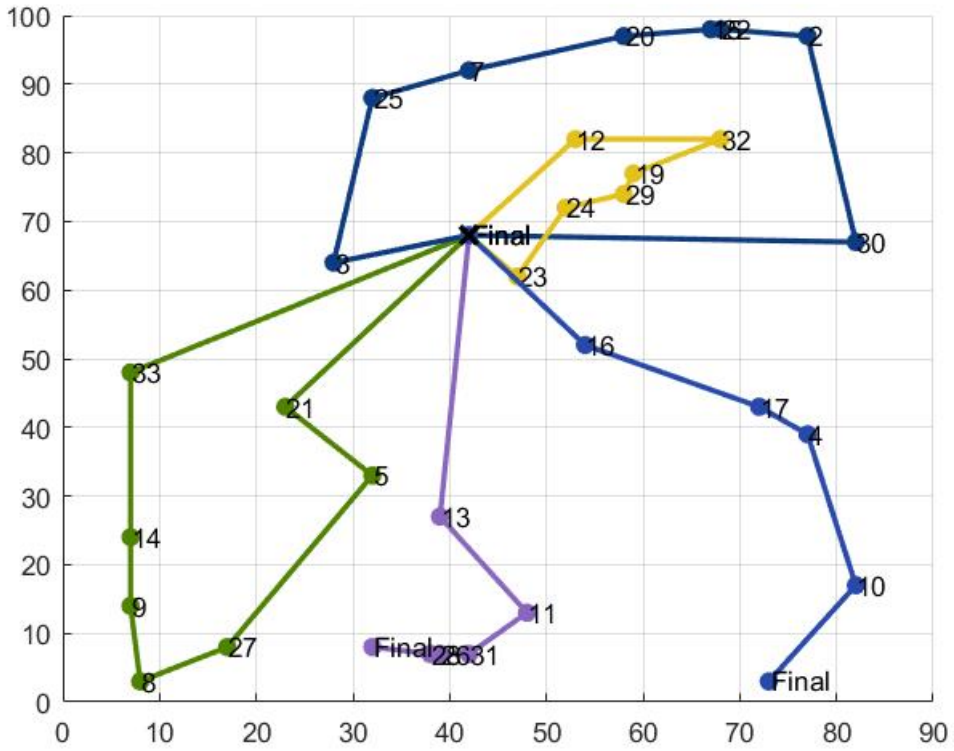
Παράδειγμα A-n33-k5

Δεύτερη Παραλλαγή

Αριθμός μυρμηγκιών=250

eksatmisi=0.1

r=0.8



<u>Διαδρομές</u>:

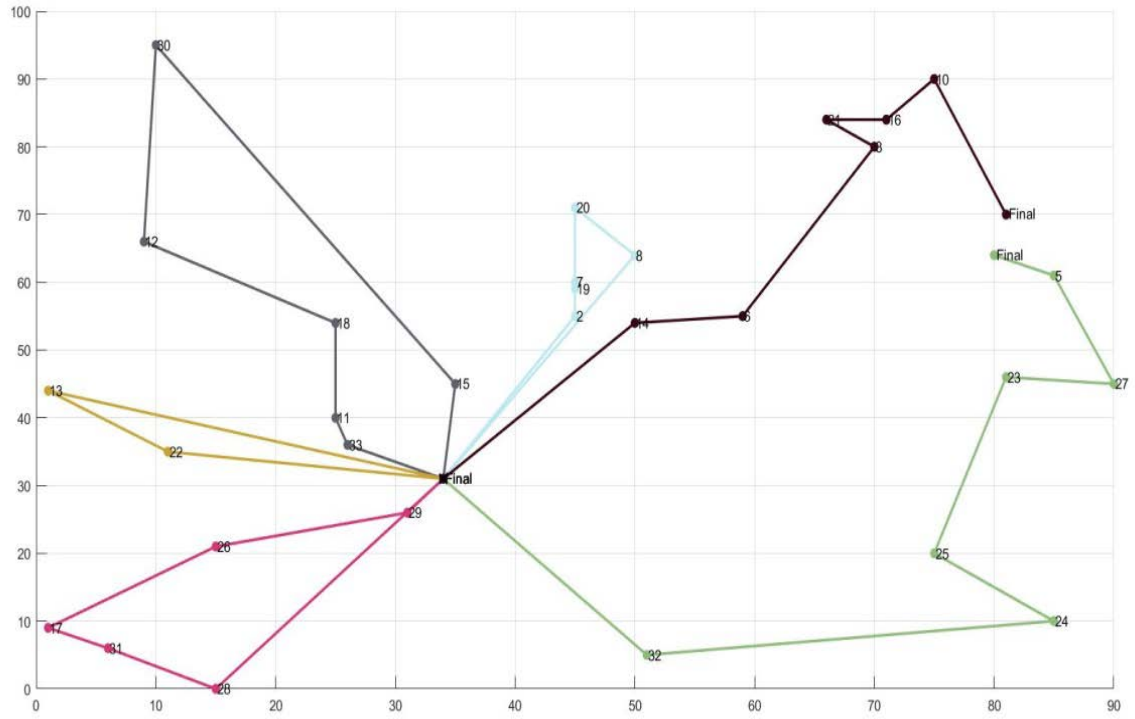
1	21	5	27	8	9	14	33	1	0
1	3	25	7	20	15	22	2	30	1
1	12	32	19	29	24	23	1	0	0
1	13	11	31	26	28	6	0	0	0
1	16	17	4	10	18	0	0	0	0

Παράδειγμα A-n33-k6

Πρώτη παραλλαγή

Αριθμός μυρμηγκιών=250

r=0.8



<u>Διαδρομές</u> :

1	33	11	18	12	30	15	1
1	2	19	7	20	8	1	0
1	22	13	1	0	0	0	0
1	28	31	17	26	29	1	0
1	32	24	25	23	27	5	9
1	14	6	3	21	16	10	4

Παράδειγμα A-n34-k5

Πρώτη παραλλαγή

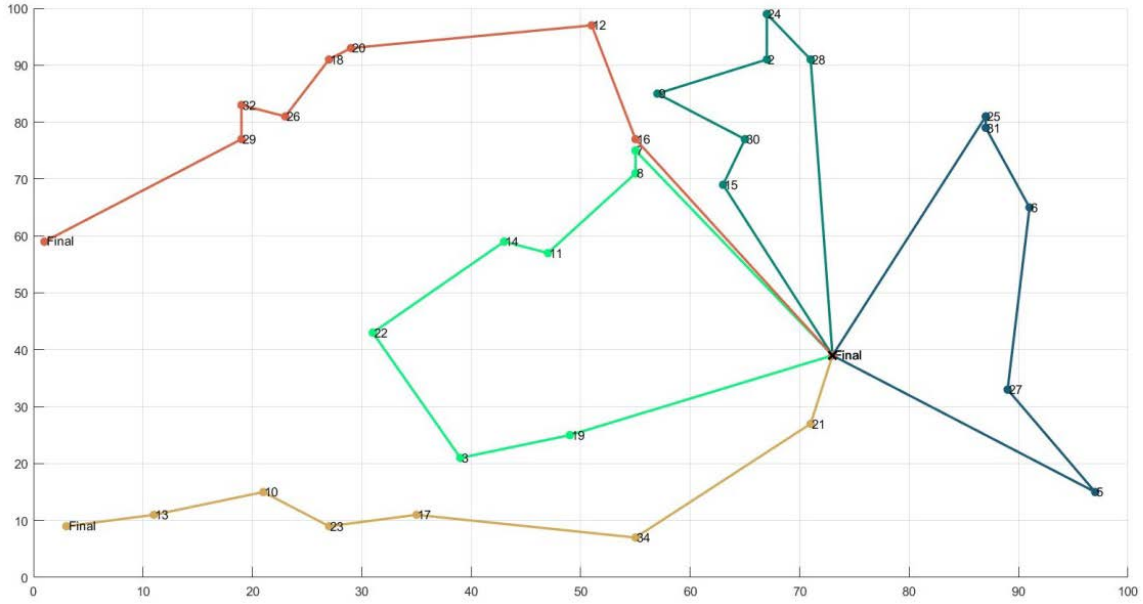
Αριθμός μυρμηγκιών=250

eksatmisi=0.1

ρ=0.8

α=0.9

β=0.1



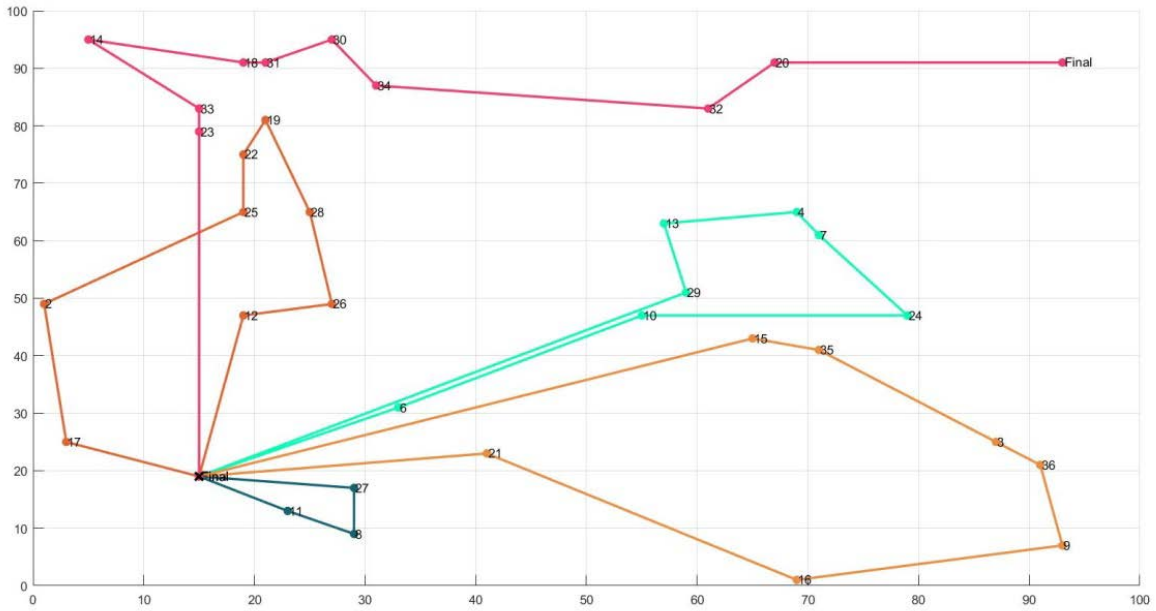
<u>Διαδρομές</u>:

1	7	8	11	14	22	3	19	1
1	15	30	9	2	24	28	1	0
1	25	31	6	27	5	1	0	0
1	21	34	17	23	10	13	4	0
1	16	12	20	18	26	32	29	33

Παράδειγμα A-n36-k5

Πρώτη παραλλαγή

Αριθμός μυρμηγκιών=250



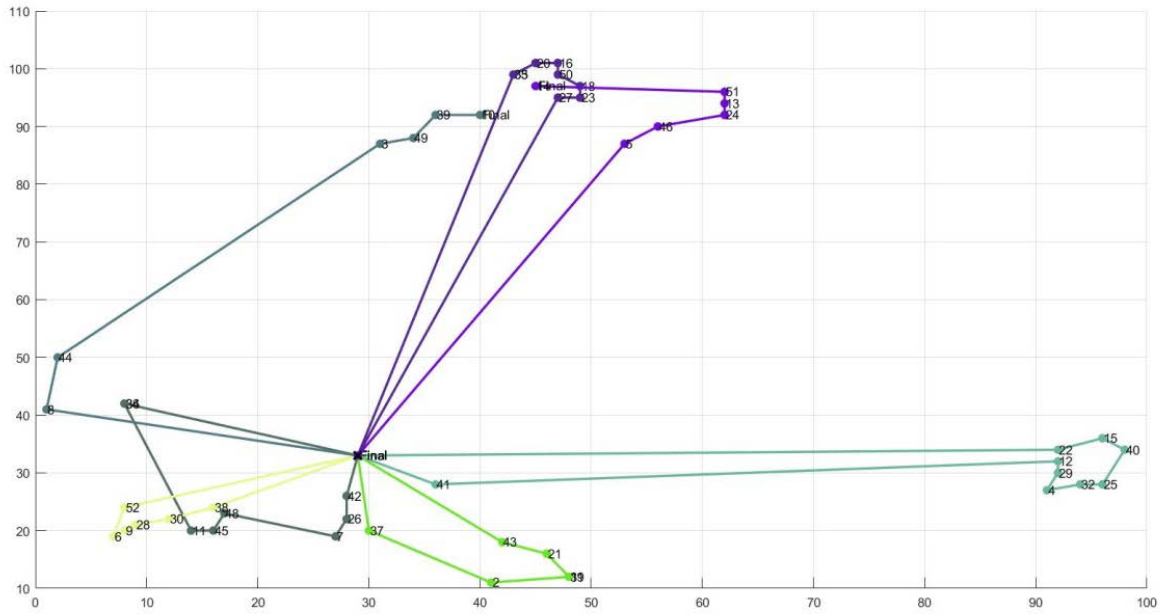
<u>Διαδρομές</u>:

1	6	10	24	7	4	13	29	1	0	0
1	11	8	27	1	0	0	0	0	0	0
1	17	2	25	22	19	28	26	12	1	0
1	15	35	3	36	9	16	21	1	0	0
1	23	33	14	18	31	30	34	32	20	5

Παράδειγμα B-n52-k7

Πρώτη παραλλαγή

Αριθμός μυρμηγκιών=250



<u>Διαδρομές</u>:

1	42	26	7	48	45	11	34	36	1	0
1	37	2	19	31	21	43	1	0	0	0
1	27	23	18	50	16	20	35	33	1	0
1	22	15	40	25	32	4	29	12	41	1
1	38	30	28	9	6	52	1	0	0	0
1	8	44	3	49	39	10	17	0	0	0
1	5	46	24	13	51	14	47	0	0	0

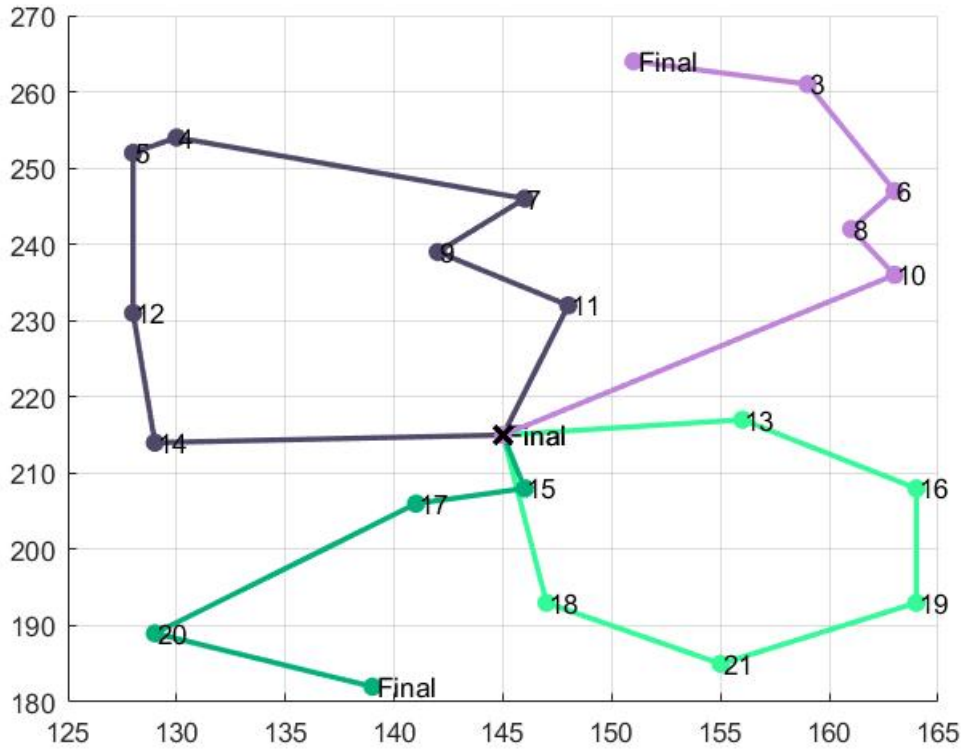
Παράδειγμα E-n22-k4

Και με τις δύο παραλλαγές

Αριθμός μυρμηγκιών=250

eksatmisi=0.1

ρ=0.8



<u>Διαδρομές</u>:

1	11	9	7	4	5	12	14	1
1	13	16	19	21	18	1	0	0
1	15	17	20	22	0	0	0	0
1	10	8	6	3	2	0	0	0

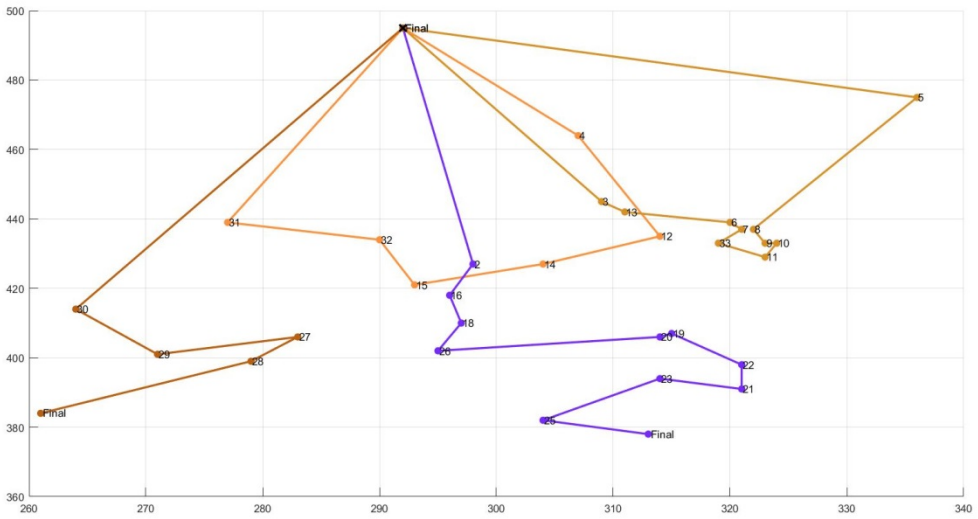
Παράδειγμα E-n33-k4

Δεύτερη παραλλαγή

Αριθμός μυρμηγκιών=250

eksatmisi=0.5

ρ=0.8



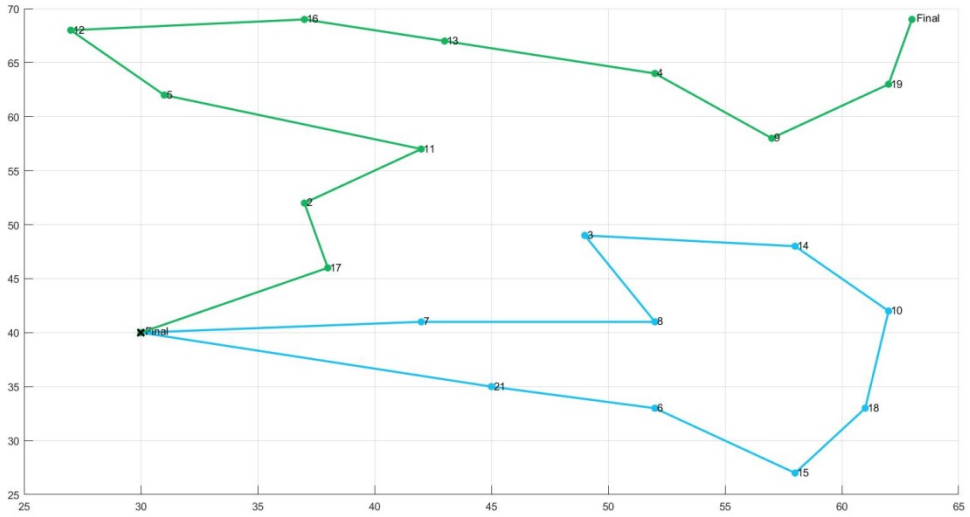
<u>Διαδρομές</u>:

1	31	32	15	14	12	4	1	0	0	0	0
1	3	13	6	7	33	11	10	9	8	5	1
1	2	16	18	26	20	19	22	21	23	25	24
1	30	29	27	28	17	0	0	0	0	0	0

Παράδειγμα P-n21-k2

Και με τις δύο παραλλαγές

Αριθμός μυρμηγκιών=250



Διαδρομές:

1	21	6	15	18	10	14	3	8	7	0	0
1	17	2	11	5	12	16	13	4	9	20	0

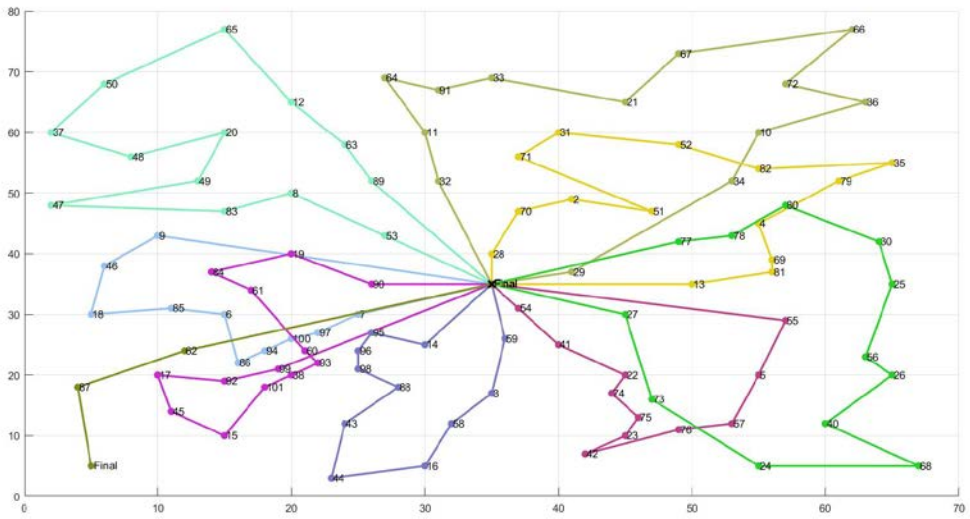
Παράδειγμα D-n101-09c

Πρώτη παραλλαγή

Αριθμός μυρμηγκιών=250

eksatmisi=0.7

ρ=0.8



<u>Διαδρομές</u>:

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

Παράρτημα Α

Στον παρακάτω πίνακα παρουσιάζονται οι τιμές των περιορισμών καθώς και το lower bound που χρησιμοποιήθηκε στον solver CPLEX. Η τρίτη στήλη αναφέρεται στον περιορισμό της χωρητικότητας, ενώ η τέταρτη στήλη στον περιορισμό του διαθέσιμου χρόνου. Τέλος οι δύο τελευταίες στήλες αναφέρονται στον αριθμό τον ιδιόκτητων οχημάτων Nu και στον συνολικό αριθμό οχημάτων Nv που χρησιμοποιήθηκαν στην εύρεση της καλύτερης λύσης.

Παράδειγμα	**LB**	**Qmax**	**maxRT**	**Nu**	**Nv**
A-n32-k5	1026.91	100	246.89	3	5
A-n33-k5	906.42	100	176.68	3	5
A-n33-k6	1071.42	100	175.83	3	6
A-n34-k5	972.29	100	201.73	3	5
A-n36-k5	963.03	100	236.53	3	5
B-n50-k8	1386.47	100	232.11	4	8
B-n51-k7	1211.32	100	186.69	4	7
B-n52-k7	940.85	100	158.17	4	7
B-n56-k7	921.14	100	186.22	4	7
B-n57-k7	1009.31	100	197.65	4	7
E-n22-k4	700.94	6000	110.47	2	4
E-n30-k3	622.46	4500	215.12	2	3
E-n33-k4	944.27	8000	262.48	2	4
E-n51-k5	886.47	160	135.77	3	5
E-n76-k10	1566.21	140	129.29	5	10
P-n21-k2	383.46	160	132.10	1	2
P-n22-k8	1183.87	3000	109.71	5	8
D101-09c	1365.38	200	230.00	5	8
D151-14b	1856.21	200	200.00	7	12
D281-08k	8418.28	900	1500.00	4	7
D361-09k	9682.37	900	1300.00	5	8

Πίνακας 5: περιορισμοί προβλήματος

Βιβλιογραφία

1. Σχεδιασμός και βελτιστοποίηση της εφοδιαστικής αλυσίδας, Ιωάννης Μαρινάκης, Αθανάσιος Μυγδαλάς, Εκδόσεις σοφία 2008.
2. Fisher, M. L., Jaikumar, R., and Wassenhove, L. N. V (1986). A Multiplier Adjustment Method for the Generalized Assigment Problem, Management Science, 32 (9), 1095-1103
3. The close-open mixed vehicle routing problem, Ran Liu, Zhidin Jiang, 2012.
4. New Best Solutions for the Close-Open Mixed Vehicle Routing Problem by using an Improved Harmony Search Algorithm, Tantikorn Pichpibul, Ruengsak Kawtummachai
5. Distance-constrained capacitated vehicle routing problems with flexible assignment of start and end depots, Alvina G.H. Kek, Ruey Long Cheu, Qiang Meng
6. https://en.wikipedia.org/wiki/Heuristic_(computer_science)
7. https://en.wikipedia.org/wiki/2-opt
8. https://el.wikiversity.org/wiki/Εφοδιαστική_Αλυσίδα_Σχεδιασμού_Συστημάτων#Logistics_-_ΔΕΑ_και_Εταιρική_Στρατηγική
9. https://www.supplychain.gr/βιβλιοθήκη/71-ορισμός-των-logistics.html
10. https://www.marketing91.com/logistics-activities/
11. https://en.wikipedia.org/wiki/Vehicle_routing_problem