ΠΟΛΥΤΕΧΝΕΙΟ ΚΡΗΤΗΣ
ΣΧΟΛΗ ΜΗΧΑΝΙΚΩΝ ΠΑΡΑΓΩΓΗΣ & ΔΙΟΙΚΗΣΗΣ

Αλγόριθμος Προσομοιωμένης Ανόπτησης για την επίλυση του Ανοιχτού-Κλειστού Προβλήματος Δρομολόγησης Οχημάτων με Ιδιόκτητα και Ενοικιαζόμενα Οχήματα και Πολλαπλές Επιστροφές στην Αποθήκη.

Διπλωματική Εργασία

Θεμιστοκλής Μαλαξιανάκης

Επιβλέπων Καθηγητής : Δρ. Ιωάννης Μαρινάκης

ΧΑΝΙΑ, 2022

Μαλαξιανάκης Θ.(2022). Αλγόριθμος Προσομοιωμένης Ανόπτησης για την επίλυση του Ανοιχτού-Κλειστού Προβλήματος Δρομολόγησης Οχημάτων με Ιδιόκτητα και Νοικιασμένα Οχήματα και Πολλαπλές Επιστροφές στην Αποθήκη.

Διπλωματική Εργασία

Σχολή Μηχανικών Παραγωγής και Διοίκησης,

Πολυτεχνείο Κρήτης, Χανιά.

Ευχαριστίες

Θα ήθελα να ευχαριστήσω τον καθηγητή μου, κ. Ιωάννη Μαρινάκη για την μόνιμη και συνεχή βοήθεια αλλά και τον βοηθό του, υποψήφιο Διδάκτορα Πολυτεχνείου Κρήτης, κ. Νικόλαο Κυριακάκη για την υπομονή και την καθοδήγηση του καθ' όλη την διάρκεια της διπλωματικής εργασίας.

Την εργασία αυτή την αφιερώνω σε όλη μου την οικογένεια και τους καλούς μου φίλους που στάθηκαν δίπλα μου σε όλα τα χρόνια των σπουδών μου στο Πολυτεχνείο Κρήτης.

Περίληψη

Στη συγκεκριμένη διπλωματική εργασία θα εξετάσουμε το ανοιχτό-κλειστό πρόβλημα δρομολόγησης οχημάτων (Close – Open Vehicle Routing Problem) με ιδιόκτητα και νοικιασμένα οχήματα και πολλαπλές επιστροφές στην αποθήκη . Πρόκειται για μία παραλλαγή του ανοιχτού προβλήματος δρομολόγησης οχημάτων καθώς , λαμβάνουμε υπόψη μας και την πιθανότητα το όχημα να επιστρέψει στην αφετηρία-αποθήκη , να ξαναφορτώσει και να συνεχίσει για μια νέα διαδρομή μέχρι να καλύψει τα χρονικά περιθώρια που του αναλογούν. Στόχος μας είναι να ελαχιστοποιήσουμε το κόστος των διαδρομών που θα πραγματοποιήσουν συνολικά όλα τα οχήματα για την βέλτιστη εξυπηρέτηση των πελατών. Οι βασικοί μας περιορισμοί για την κάθε διαδρομή είναι, αρχικά να μην υπερβεί το όχημα το μέγιστο όριο χωρητικότητας και κατά δεύτερον να μην ξεπεράσει το μέγιστο επιτρεπτό χρονικό περιθώριο. Η σειρά με την οποία θα εξυπηρετηθούν οι πελάτες θα βασιστεί στον αλγόριθμο του Πλησιέστερου Γείτονα , με τον οποίο θα εξάγουμε και την αρχική μας λύση στο πρόβλημα. Έπειτα αποσκοπώντας σε μία καλύτερη και πιο αποδοτική λύση σε θέμα συνολικού κόστους, θα βελτιώσουμε την αρχική μας λύση μέσω τριών αλγορίθμων τοπικής αναζήτησης (2 Opt , 1-1 exchange , 1-0 relocate). Τέλος, για την περαιτέρω βελτίωση της λύσης μας θα κάνουμε χρήση του ευρετικού αλγορίθμου της Προσομοιωμένης Ανόπτησης (Simulated Annealing), ο οποίος θα εφαρμοστεί με μία σειρά διαφορετικών προσεγγίσεων σύμφωνα με τη μορφή της συνάρτησης μείωσης της θερμοκρασίας που θα επιλέξουμε.

Abstract

This thesis will explore the Close-Open Vehicle Routing Problem with private and household vehicles, including multiple returns to the depot. It is a variation of the open vehicle routing problem that considers the possibility of vehicles returning to the starting depot, reloading, and continuing a new route until they cover their assigned time slots. Our objective is to minimize the cost of all vehicles' trips while providing the best customer service. Our primary restrictions for each route are to ensure that the vehicle does not exceed the maximum capacity limit and the maximum allowed time margin. To determine the order in which customers will be served, we will use the Nearest Neighbor algorithm to obtain our initial solution to the problem. To further improve the overall cost efficiency of our solution, we will employ three local search algorithms (2 Opt, 1-1 exchange, 1-0 relocate). Finally, we will use the Simulated Annealing metaheuristic algorithm to enhance our solution, which will be applied with different approaches depending on the temperature reduction function used.

Περιεχόμενα

Κεφάλαιο 1 - Εισαγωγή στην εφοδιαστική αλυσίδα

1.1 Ιστορική Αναδρομή Εφοδιαστικής Υποστήριξης (Logistics)

Τα Logistics , ή πιο σωστά η Διοίκηση Logistics θεωρείται σήμερα από πολλούς ως ένας από τους σημαντικότερους επιχειρηματικούς κλάδους αλλά και από τους βασικότερους ανταγωνιστικούς παράγοντες στην επιχειρηματικότητα. Ετυμολογικά, ο όρος Εφοδιαστική/Logistics έχει προέλευση από τον ελληνικό όρο «λόγος», που σημαίνει λογική, με την έννοια της εκλογίκευσης και σκοπό την επίτευξη ορισμένων συγκεκριμένων στόχων. Με την έννοια αυτή λέγεται ότι έχει γίνει αρχική χρήση του όρου «Λογιστική» πρώτη φορά από τον αυτοκράτορα του Βυζαντίου «Λέοντα τον Σοφό», σε σχέση με τη μέριμνα για τον εφοδιασμό, την τροφοδοσία και τη διατήρηση του στρατού της αυτοκρατορίας με τρόφιμα, ρουχισμό, πολεμοφόδια, κτλ. Κατ' άλλους ιστορικούς, ως πρώτος "Logistician" αναφέρεται ο Μέγας Αλέξανδρος, ο οποίος εφάρμοσε ίδιες στρατηγικές για τον εφοδιασμό των στρατευμάτων της αυτοκρατορίας του (Engles, 1978). Ακόμη, ο Μέγας Ναπολέων είχε σημειώσει ότι «οι στρατοί προχωρούν με το στομάχι τους». Αξιόλογο είναι σε γενικότερο περισκόπιο να αναφέρουμε ότι η ανάπτυξη των πολιτισμών των αρχαίων Ελλήνων, των Αιγυπτίων, των Φοινίκων και αργότερα της Ρωμαϊκής αυτοκρατορίας, είχαν στηριχτεί σε πρωτοπόρα για την εποχή τους μεταφορικά συστήματα (δίκτυα), που αποτελούν σημαντική προϋπόθεση και παράγοντα της Εφοδιαστικής/Logistics.

Σε πιο σύγχρονη ιστορική και επιστημονική αναφορά, μαζική χρήση «Εφοδιαστικής» έγινε κατά τη διάρκεια του Β΄ Παγκοσμίου Πολέμου από τις ΗΠΑ και τους Συμμάχους για τον εφοδιασμό των νηοπομπών των συμμαχικών δυνάμεων, μέσω χρησιμοποίησης κατά βάση "επιχειρησιακής έρευνας " και εκτεταμένης χρήσης «προσομοιώσεων».

1.2 Εφοδιαστική(Logistics)

Η Διοίκηση Εφοδιαστικής Αλυσίδας ορίζεται ως η διαδικασία του σχεδιασμού, υλοποίησης και ελέγχου της αποτελεσματικής και αποδοτικής ροής και αποθήκευσης προϊόντων, υπηρεσιών και σχετικών πληροφοριών από την αρχική παραγγελία / παραγωγή μέχρι την τελική παράδοση στον τελικό καταναλωτή, με σκοπό την εκπλήρωση των απαιτήσεων του πελάτη. Περιλαμβάνει το σχεδιασμό, την εφαρμογή και τον έλεγχο ενός μεγάλου αριθμού λειτουργιών (όπως προμήθειες, διακίνηση υλικών, πρόβλεψη ζήτησης, αποθέματα, επεξεργασία παραγγελιών, αποθήκευση, συσκευασία, μεταφορές, ανταλλακτικά και επισκευές, εξυπηρέτηση πελατών, αντιμετώπιση επιστρεφόμενων προϊόντων, ανακύκλωση και αποκομιδή απορριμμάτων, κλπ.), για να μετασχηματίσει τις πρώτες ύλες που λαμβάνονται από τους προμηθευτές, σε έτοιμα προϊόντα που προσφέρονται στους πελάτες.

Τα Logistics βρίσκουν εφαρμογή σε δύο κυρίως πεδία :

• Στην επιχείρηση, η οποία πρέπει να οργανώσει την εισροή-εκροή των υλικών και την εσωτερική διακίνηση των προϊόντων κατά τέτοιον τρόπο, έτσι ώστε να εξασφαλίζει τη μέγιστη ικανοποίηση των πελατών της.

• Στην εφοδιαστική αλυσίδα, η οποία αποτελείται από όλες εκείνες τις επιχειρήσεις και τους οργανισμούς που είναι αναγκαίοι έτσι ώστε ένα προϊόν να καταλήξει στον πελάτη ενώ ήταν πρώτες ύλες.

Οι τομείς δραστηριοτήτων που υπάγονται στη λειτουργία των logistics είναι οι παρακάτω:

- Αποθήκευση
- Διαχείριση αποθεμάτων
- Διανομή - Μεταφορές
- Διαχείριση υλικών
- Συσκευασία
- Πληροφορική - Τηλεματική

1.3 Η έννοια της Εφοδιαστικής Αλυσίδας (Supply Chain)

Οι οργανισμοί που συμμετέχουν στη διάθεση αγαθών ή υπηρεσιών στους πελάτες αναφέρονται συλλογικά ως η εφοδιαστική αλυσίδα (SC) και περιλαμβάνουν κατασκευαστές, προμηθευτές υλικών, χώρους αποθήκευσης, κέντρα διανομών, μεταφορείς, πωλητές, πελάτες, πρώτες ύλες, αλλά και έτοιμα προϊόντα που μπορεί να υπάρχουν ανάμεσα. Πέρα από τους οργανισμούς, η εφοδιαστική αλυσίδα περιλαμβάνει όλες αυτές τις ενέργειες που σχετίζονται με τη διακίνηση και τη μετατροπή των αγαθών από το στάδιο των πρώτων υλών στον τελικό χρήστη, καθώς και τη ροή πληροφοριών που σχετίζονται με αυτές τις δραστηριότητες, . Αν και η λέξη "αλυσίδα" υποδηλώνει γραμμικότητα, προκύπτουν πολύπλοκα πολυεπίπεδα δίκτυα οργανισμών μέσα στα οποία διακινούνται αγαθά, κεφάλαια αλλά και πληροφορίες με αποτέλεσμα να αποτελεί ένα δυναμικό πολυσύνθετο σύστημα.

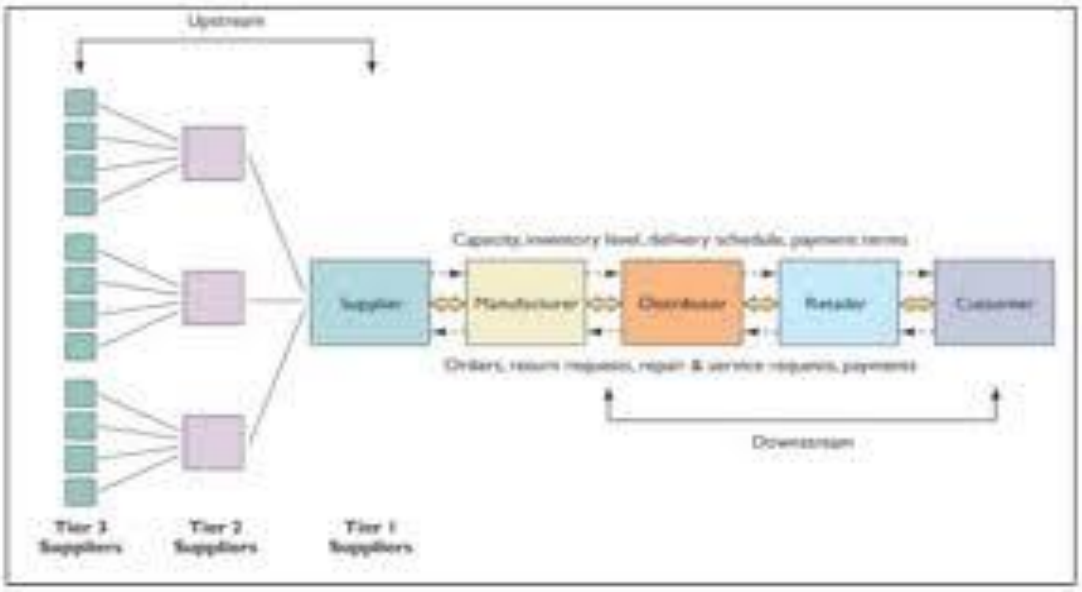
Εικόνα 1.1 Η γραμμική ροή των αγαθών στην Εφοδιαστική Αλυσίδα

Η εφοδιαστική αλυσίδα αποτελείται από εγκαταστάσεις που κατανέμονται διάσπαρτα στο χώρο και από συνδέσμους μεταφοράς που συνδέουν αυτές τις εγκαταστάσεις, ενώ

ταυτόχρονα υλικά, κεφάλαιο και πληροφορίες ρέουν κατά μήκος της. Οι ακόλουθες θεμελιώδεις ερμηνείες της εφοδιαστικής αλυσίδας έχουν αναπτυχθεί από την τεράστια γκάμα ορισμών που διατίθενται επί του παρόντος:

- Ένα δίκτυο οργανισμών που εμπλέκονται μέσω ανοδικών και καθοδικών συνδέσεων στις διαφορετικές διαδικασίες και δραστηριότητες που παράγουν αξία με τη μορφή προϊόντων ή υπηρεσιών στα χέρια του τελικού καταναλωτή (Christopher 1992),
- Μια αλληλουχία ενεργειών τροφοδοσίας που ορίζεται από έναν κόμβο, τους προμηθευτές και τους πελάτες του, μεταξύ των οποίων διακινούνται υλικά και πληροφορίες, και αναφέρεται, τόσο σε ενδοεπιχειρησιακές δραστηριότητες, όσο και σε δίκτυα επιχειρήσεων (Τσουδερός 2008).

Στη σημερινή εποχή, ο ανταγωνισμός δεν είναι πλέον μεταξύ επιχειρήσεων, αλλά μεταξύ αλυσίδων εφοδιασμού. Μια τυπική αλυσίδα εφοδιασμού περιλαμβάνει 5 επίπεδα: προμηθευτές που παρέχουν πρώτες ύλες, παραγωγούς, διάφορα κέντρα διανομής, έμπορους λιανικής και τελικούς καταναλωτές. Μεταξύ των επιπέδων της εφοδιαστικής αλυσίδας ανταλλάσσονται προϊόντα, χρηματικές αξίες, καθώς και πληροφορίες. Υπάρχουν δύο κατευθύνσεις ροών, η μία ονομάζεται κατωφερής κατεύθυνση (Downstream), από τους προμηθευτές προς τους τελικούς πελάτες και η άλλη ανωφερής κατεύθυνση (Upstream), από τους τελικούς καταναλωτές προς τους προμηθευτές (Βιδάλης, 2017)



Εικόνα 1.2 Επίπεδα Εφοδιαστικής Αλυσίδας και κατευθύνσεις ροών

Η ευτυχία των πελατών είναι ένας από τους κύριους στόχους της εφοδιαστικής αλυσίδας. Τα στάδια της εφοδιαστικής αλυσίδας, οι ροές πληροφοριών ή προϊόντων που την διέπουν καταλήγουν σε έξοδα, τα οποία στη συνέχεια προστίθενται στο τελικό προϊόν. Ο καταναλωτής είναι η μοναδική πηγή εσόδων, επομένως είναι επιτακτική ανάγκη να είναι ευχαριστημένος με το τελικό προϊόν όταν φτάσει στα χέρια του. Για να διασφαλιστεί ότι ο καταναλωτής λαμβάνει το τελικό προϊόν όσο το δυνατόν γρηγορότερα, στην καλύτερη τιμή, με τις καλύτερες υπηρεσίες και με ενσωματωμένες νέες τεχνολογίες, η διαχείριση της αλυσίδας εφοδιασμού πρέπει να γίνεται με τον βέλτιστο τρόπο. Ο κύριος στόχος είναι να μειωθούν όλα τα κόστη που μπορεί να προκύψουν σε όλη την αλυσίδα και σε κάθε στάδιο της, συμπεριλαμβανομένων των δαπανών μεταφοράς, διανομής και άλλων δαπανών, αυξάνοντας τελικά τα συνολικά κέρδη της εταιρείας, ελαχιστοποιώντας το ολικό κόστος και μειώνοντας δραστικά τον χρονικό κύκλο της συνολικής διαδικασίας.

1.4 Διαχείριση Εφοδιαστικής Αλυσίδας (Supply Chain Management)

Η διαχείριση της εφοδιαστικής αλυσίδας (SCM) είναι η διαδικασία διαχείρισης της ροής αγαθών, υπηρεσιών και πληροφοριών από το σημείο προέλευσης στο σημείο κατανάλωσης, προκειμένου να ικανοποιηθούν οι ανάγκες των πελατών. Αυτό περιλαμβάνει τα πάντα, από την προμήθεια πρώτων υλών, την κατασκευή και τη συναρμολόγηση αγαθών, την παράδοση και τη διανομή τους στους πελάτες. Αυτό συχνά περιλαμβάνει συντονισμό και συνεργασία με προμηθευτές, κατασκευαστές, διανομείς και άλλους εταίρους στην αλυσίδα εφοδιασμού. Η Διαχείρηση Εφοδιαστικής Αλυσίδας περιλαμβάνει επίσης τη διαχείριση του αποθέματος, καθώς και τη μεταφορά αγαθών και υλικών από τη μια τοποθεσία στην άλλη. Επιπλέον, τα συστήματα Διαχείρησης Εφοδιαστικής Αλυσίδας περιλαμβάνουν συνήθως εργαλεία διαχείρισης logistics και διαχείρισης μεταφοράς για να βοηθήσουν στη βελτιστοποίηση των χρόνων παράδοσης και στη μείωση του κόστους.

Στη σύγχρονη εποχή, η πολυπλοκότητα της εφοδιαστικής αλυσίδας έχει αυξηθεί πολύ με πολλαπλά στρώματα υποπρομηθευτών και μεσαζόντων, επομένως μια καλή διαχείριση της εφοδιαστικής αλυσίδας απαιτεί κατάλληλη τεχνολογική υποστήριξη με τη μορφή λογισμικού SCM για καλύτερο συντονισμό, ορατότητα και αυτοματοποίηση. Συνολικά, ο στόχος του SCM είναι να βελτιστοποιήσει ολόκληρη τη διαδικασία της εφοδιαστικής αλυσίδας, έτσι ώστε τα αγαθά και οι υπηρεσίες να παράγονται και να παραδίδονται με τον πιο αποτελεσματικό και οικονομικά αποδοτικό τρόπο, καλύπτοντας παράλληλα τις ανάγκες των πελατών.

Η διαχέιρηση της εφοδιαστικής αλυσίδες απαρτίζεται από τέσσερεις βασικούς συντελεστές οι οποίοι είναι οι εξής:

1. Παραγωγή: Μία επιχείρηση οφείλει να εστιάσει στη ποσότητα του εμπορεύματος που θα παραχθεί προβλέποντας τη ζήτηση του προϊόντος, σε ποια τοποθεσία και με ποιους προμηθευτές θα συνεργαστεί.
2. Αποθήκευση: Μία επιχείρηση πρέπει να αποφασίσει που θα αποθηκεύσει τα προϊόντα της με τον βέλτιστό τρόπο, πόσα τη συμφέρει να αποθηκεύσει και πως θα διαχειριστεί το απόθεμα τους.
3. Διανομή: Μία επιχείρηση πρέπει να πάρει τις σωστές αποφάσεις για την πιο οικονομική, γρήγορη, ασφαλή και αποδοτική μεταφορά των προϊόντων της.
4. Χρηματοδότηση: Μία επιχείρηση οφείλει να αναζητήσει τους καλύτερους όρους αποπληρωμής από τους πελάτες της και πίστωσης από τους προμηθευτές της.

Κεφάλαιο 2- Προβλήματα Δρομολόγησης Οχημάτων

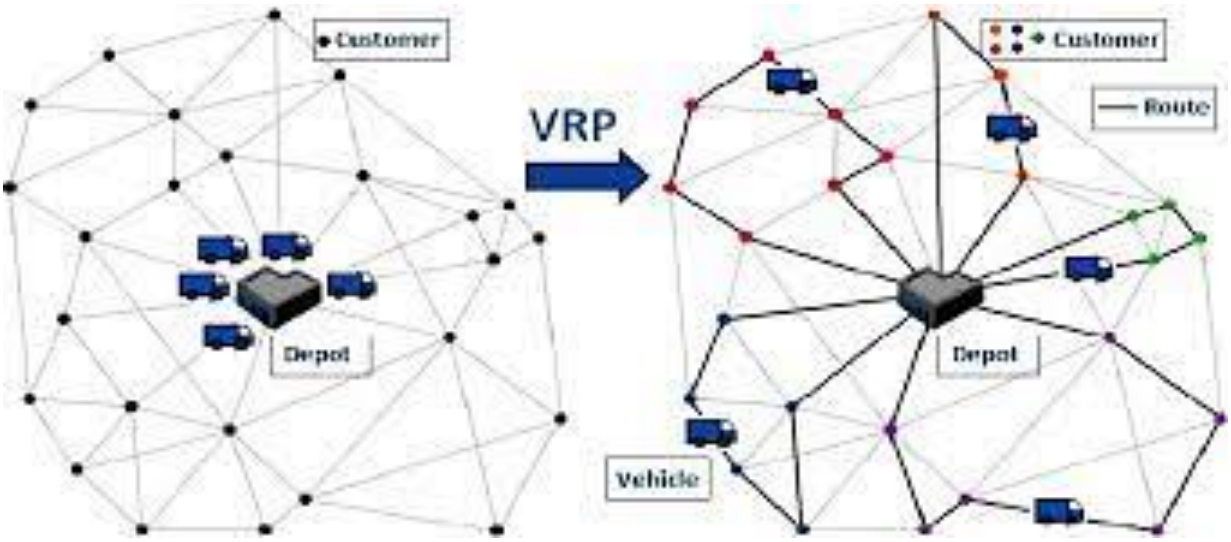
2.1 Το Πρόβλημα Δρομολόγησης Οχημάτων (VRP)

Το 1959, οι Dantzing & Ramser πρότειναν για πρώτη φορά το Πρόβλημα Δρομολόγησης Οχημάτων (VRP) ή αλλιώς το Περιορισμένης Χωρητικότητας Πρόβλημα Δρομολόγησης Οχημάτων (CVRP), το οποίο είναι τώρα ένα από τα πιο σημαντικά και πρακτικά ζητήματα διανομής της εφοδιαστικής αλυσίδας. Η πρώτη αλγοριθμική μέθοδος για την επίλυση αυτών των προβλημάτων αναπτύχθηκε από τους Dantzing και Ramser και χρησιμοποιήθηκε για την αυτοματοποίηση της παράδοσης βενζίνης σε πολλά πρατήρια καυσίμων. Στη συνέχεια, το 1964, οι Clarke & Wright επινόησαν μια ευρετική τεχνική εξοικονόμησης που ξεπερνάει κατά πολύ την Dantzing & Ramser's με το όνομα αλγόριθμος “Savings”.

Το Πρόβλημα Δρομολόγησης Οχημάτων είναι ουσιαστικά ένα πρόβλημα διανομής και παραλαβής προϊόντων και η επίλυσή του αποσκοπεί στην ελαχιστοποίηση του συνολικού κόστους των εταιρειών. Το πρόβλημα προϋποθέτει την ύπαρξη:

1) Μίας αποθήκης-αφετηρίας από την οποία ξεκινούν και καταλήγουν όλα τα οχήματα
2) Ενός στόλου οχημάτων με συγκεκριμένη χωρητικότητα
3) Ενός συνόλου πελατών με συγκεκριμένη ζήτηση
4) Την ύπαρξη συγκεκριμένου χρονικού περιθωρίου για το κάθε όχημα

Επιπλέον τόσο οι αποστάσεις μεταξύ των πελατών και της αποθήκης, όσο και οι αποστάσεις των πελατών μεταξύ τους οφείλουν να είναι γνωστές. Ακόμα, ο κάθε πελάτης, ο οποίος αποτελεί έναν κόμβο, μπορεί να εξυπηρετηθεί μόνο από ένα όχημα το οποίο εξυπηρετεί ένα υποσύνολο πελατών κάνοντας κυκλικές διαδρομές, ενώ όποιος πελάτης έχει μεγαλύτερη ζήτηση από την χωρητικότητα ενός οχήματος δεν μπορεί να εξυπηρετηθεί από αυτό και τέλος κάθε όχημα οφείλει μετά τη διαδρομή του να επιστρέψει στην αποθήκη. Συνεπώς σκοπός του προβλήματος είναι να βρεθεί η βέλτιστη διαδρομή για το κάθε όχημα έτσι ώστε να εξυπηρετηθούν όλοι οι πελάτες με το ελάχιστο δυνατό κόστος για την εταιρεία. Στην παρακάτω εικόνα παρατηρούμε ότι το πρόβλημα μας περιγράφεται ως ένα γράφημα όπου τα τόξα περιγράφουν τους δρόμους του δικτύου αλλά ταυτόχρονα και το κόστος το οποίο υπολογίζεται συναρτήσει της απόστασης που πρέπει να διασχίσει ένα όχημα. Οι κόμβοι αντιπροσωπεύουν τους πελάτες και ο κεντρικός κόμβος του γραφήματος αντιπροσωπεύει την αποθήκη από την οποία ξεκινούν όλα τα τόξα και όλα τα οχήματα τα οποία πάλι επιστρέφουν σε αυτή.



Εικόνα 2.1 Γράφημα του Περιορισμένης Χωρητικότητας Προβλήματος Δρομολόγησης Οχημάτων

Τη μαθηματική μοντελοποίηση του προβλήματος τη διατύπωσαν οι Fisher & Jaikumar και έχει ως εξής :

Έστω:

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

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

Αντικειμενική συνάρτηση :

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

υπο περιορισμούς:

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

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

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

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

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

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

Ο περιορισμός (1) δηλώνει ότι ο κάθε πελάτης εξυπηρετείται από μόνο ένα όχημα εκτός της αποθήκης που την επισκέπτονται όλα τα οχήματα, ο περιορισμός (2) εκφράζει τη συγκεκριμένη χωρητικότητα του κάθε οχήματος ενώ ο περιορισμός (3) δείχνει ότι φεύγει από τον πελάτη αν τον έχει ήδη επισκεφθεί. Ο περιορισμός (4) δείχνει ότι , εξαιρουμένης της αποθήκης, το όχημα έχει επισκεφθεί όλους τους πιθανούς πελάτες που μπορούσε και τέλος οι περιορισμοί (5),(6) εξασφαλίζουν ότι οι μεταβλητές χ,y είναι ακέραιες και μη αρνητικές.

Το Πρόβλημα Δρομολόγησης Οχημάτων αποτελεί ένα πρόβλημα ακέραιου γραμμικού προγραμματισμού το οποίο έγκειται στην κατηγορία μη-πολυωνυμικών δύσκολων προβλημάτων (NP-Hard) , γεγονός που δηλώνει ότι η υπολογιστική προσπάθεια που απαιτείται για την επίλυση του αυξάνεται εκθετικά σύμφωνα με τον αριθμό των πελατών που πρέπει να εξυπηρετηθούν.

2.2 Το Ανοιχτό Πρόβλημα Δρομολόγησης Οχημάτων (OVRP)

Σε κάθε ένα από τα προηγούμενα ζητήματα, το όχημα έπρεπε να κάνει έγκαιρη επιστροφή στην αποθήκη μετά την ολοκλήρωση των δρομολογίων. Στο ανοιχτό πρόβλημα δρομολόγησης οχημάτων, το όχημα δεν επιστρέφει στην αποθήκη, αλλά συνεχίζει τη διαδρομή του μέχρι να φτάσει στον τελευταίο πελάτη που μπορεί να εξυπηρετήσει. Αυτό συμβαίνει διότι ο στόλος των οχημάτων δεν ανήκει στην εταιρεία, αλλά η εταιρεία χρησιμοποιεί μεσάζοντες που

νοικιάζουν το στόλο των οχημάτων τους στην συγκεκριμένη εταιρεία για να ολοκληρώσει τις διαδρομές της και έτσι τελικά προκύπτει το ανοιχτό πρόβλημα δρομολόγησης οχημάτων. Ως εκ τούτου, χρησιμοποιεί το βέλτιστο αριθμό ενοικιαζόμενων οχημάτων για την εξυπηρετήσει των πελατών αντί να χάνει χρόνο οδηγώντας τα πίσω στην αποθήκη. Ο στόχος είναι η μείωση της συνολικής χιλιομετρικής απόστασης εξυπηρετώντας παράλληλα όλους τους πελάτες. Το OVRP χρησιμοποιείται συχνά σε εφαρμογές logistics και σχεδιασμού μεταφορών, όπως η βελτιστοποίηση διαδρομής παράδοσης ή η συλλογή σκουπιδιών.

2.3 Το Ανοιχτό – Κλειστό Πρόβλημα Δρομολόγησης Οχημάτων με Ιδιόκτητα Νοικιασμένα Οχήματα και Πολλαπλές Επιστροφές στην Αποθήκη

Στα παραπάνω προβλήματα δρομολόγησης που αναλύσαμε παρατηρήσαμε ότι σε περίπτωση που η συνολική χρονική διάρκεια της κάθε διαδρομής φτάσει το μέγιστο χρονικό όριο που το όχημα μπορεί να είναι εν κινήσει, τότε το όχημα οφείλει να γυρίσει στην αποθήκη αν είναι ιδιόκτητο (CVRP) ή να επιστρέψει στον εξωτερικό συνεργάτη της εταιρείας αν είναι νοικιασμένο (OVRP), παράγοντας πιο αποτελεσματικές διαδρομές αφού δεν αναγκάζονται να επιστρέψει στην αποθήκη. Ωστόσο στο COMVRP το οποίο πρόκειται για έναν συνδυασμό των CVRP και OVRP, σε περίπτωση όπου σε κάποιο όχημα περισσέψει χρόνος, έχει τη δυνατότητα να επιστρέψει στην αποθήκη , να ξαναφορτώσει και να συνεχίσει σε μία νέα διαδρομή, εξυπηρετώντας νέους πελάτες μέχρι να εξαντληθούν τα χρονικά του περιθώρια, γεγονός που ωθεί μία εταιρεία να μην νοικιάσει οχήματα αφού μπορεί να καλύψει την ζήτηση των πελατών της με το δικό της στόλο. Παρ’όλ’ αυτά σπάνια μία εταιρεία δύναται να καλύψει όλη τη ζήτηση με το δικό της στόλο οχημάτων και αναγκάζεται να νοικιάσει τον απαραίτητο αριθμό οχημάτων ώστε να μπορέσει να καλύψει την εναπομένουσα ζήτηση. Αντιλαμβανόμαστε ότι τα

ιδιόκτητα οχήματα πάντα θα καταλήγουν στην αποθήκη σε αντίθεση με τα ενοικιαζόμενα τα οποία ανεξαιρέτως των επιστροφών τους στην αποθήκη λόγω της μη εξάντλησης των χρονικών τους περιθωρίων δε θα καταλήξουν ποτέ στην αποθήκη αλλά στον τελευταίο πελάτη της διαδρομής τους. Επίσης όσο περισσότερες επιστροφές για επαναφορτώσεις κάνουν τα ιδιόκτητα οχήματα τόσο λιγότερα ενοικιαζόμενα οχήματα χρειάζεται η εταιρεία ελαχιστοποιώντας σημαντικά το κόστος της. Η αντικειμενική συνάρτηση του προβλήματος είναι η εξής:

Αντικειμενική συνάρτηση :

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

Όπου οι μεταβλητές μας αναλύονται στον παρακάτω πίνακα:

X_{ij}^k	$= \begin{cases} \mathbf{1}, & \boldsymbol{\alpha\nu\ \tau o\ \acute{o}\chi\eta\mu\alpha\ k\ \pi\acute{\alpha}\epsilon\iota\ \alpha\pi'\tau o\nu\ \pi\epsilon\lambda\acute{\alpha}\tau\eta\ i\ \sigma\tau o\ j} \\ \mathbf{0}, & \boldsymbol{\alpha\lambda\lambda\iota\acute{\omega}\varsigma} \end{cases}$
u_i	**Συνεχής μεταβλητή η οποία χρησιμοποιείται σαν άνω φράγμα για την χωρητικότητα του οχήματος**
q_i	**Ζήτηση του πελάτη i**
h_i	**Συνεχής μεταβλητή η οποία χρησιμοποιείται σαν άνω φράγμα για την απόσταση που μπορεί να διανύσει ένα όχημα**

F_k	**Κόστος οχήματος k**
H_k	**Μέγιστη απόσταση οχήματος k**
Q_k	**Μέγιστη χωρητικότητα οχήματος k**
C_{ij}	**Κόστος/Απόσταση μεταξύ πελάτη i και πελάτη j**
Nu	**Μέγιστος αριθμός ιδιόκτητων οχημάτων**
K	**Αριθμός οχημάτων**
N	**Αριθμός πελατών**

Πίνακας 2.1 Επεξήγηση μεταβλητών της αντικειμενικής συνάρτησης του Ανοιχτού-Κλειστού Προβλήματος Δρομολόγησης Οχημάτων με Ιδιόκτητα και Ενοικιαζόμενα Οχήματα και Πολλαπλές Επιστροφές στην Αποθήκη

Υπό τους περιορισμούς:

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

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

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

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

$h_0 = 0$ (5)

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

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

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

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

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

Ο περιορισμός (1) επισημαίνει ότι όλοι οι πελάτες πρέπει να εξυπηρετηθούν από ένα και μόνο όχημα, ο περιορισμός (2) δηλώνει ότι εφόσον ένα όχημα καταλήξει σε έναν πελάτη και τον εξυπηρετήσει οφείλει αμέσως μετά να φύγει από αυτόν ενώ οι περιορισμοί (3), (4) αποτελούν περιορισμούς Miller Tucker Zemlin (MTZ) και εξασφαλίζουν τόσο την απουσία υπόκυκλων των οχημάτων όσο και την αποτροπή των οχημάτων να ξεπεράσουν τα όρια χωρητικότητας. Έπειτα οι περιορισμοί (5), (6), (7), (8) απαγορεύουν στα οχήματα να ξεπεράσουν τα χρονικά περιθώρια που τους αντιστοιχούν σε μία διαδρομή και τέλος ο περιορισμός (9) υποδηλώνει ο αριθμός των ιδιόκτητων οχημάτων έχει ανώτατο όριο Nu.

Κεφάλαιο 3 – Αλγόριθμοι Επίλυσης Προβλημάτων Εφοδιαστικής Αλυσίδας

3.1 Απλοί Ευρετικοί Αλγόριθμοι (Heuristics)

Ένας ευρετικός αλγόριθμος είναι μια μέθοδος επίλυσης προβλημάτων που χρησιμοποιεί μια πρακτική προσέγγιση για να βρει μια λύση δίνοντας μεγαλύτερη βαρύτητα στην ταχύτητα επίλυσης ενός προβλήματος συνδυαστικής βελτιστοποίησης. Είναι επίσης γνωστοί ως αλγόριθμοι "του βασικού κανόνα" και χρησιμοποιούνται συχνά όταν μια ακριβής λύση δεν είναι εφικτή ή όταν η ακριβής λύση είναι πολύ χρονοβόρα για να υπολογιστεί. Οι ευρετικοί αλγόριθμοι χρησιμοποιούνται συνήθως στην τεχνητή νοημοσύνη, τη μηχανική μάθηση και άλλα πεδία όπου η εύρεση μιας βέλτιστης λύσης είναι δύσκολη. Έχουν σχεδιαστεί για να λειτουργούν γρήγορα και να παρέχουν μια καλή λύση στις περισσότερες περιπτώσεις, αλλά δεν είναι εγγυημένο ότι θα βρουν την καλύτερη λύση και εφαρμόζονται κυρίως σε NP-Hard προβλήματα κυρίως με τη μέθοδο κατασκευής μίας αρχικής λύσης. Ωστόσο, για να γίνει αποδεκτή μία λύση ενός ευρετικού αλγορίθμου πρέπει να ικανοποιεί κάποια βασικά κριτήρια όπως η ποιότητα λύσης, δηλαδή η απόκλιση της λύσης του αλγορίθμου από τη βέλτιστη λύση του προβλήματος να είναι σχετικά μικρή και η ευχέρεια εύρεσης λύσεων. Μπορούμε να διαχωρίσουμε τους ευρετικούς αλγορίθμους σε τρεις διαφορετικές κατηγορίες σύμφωνα με τη λογική στην οποία στηρίζονται και αυτές είναι:

1. Οι αλγόριθμοι απληστίας (Greedy algorithms)
2. Οι προσεγγιστικοί αλγόριθμοι (Approximation algorithms)
3. Οι αλγόριθμοι τοπικής αναζήτησης (Local search algorithms)

Στην παρούσα διπλωματική εργασία θα κάνουμε χρήση αλγόριθμων απληστίας και τοπικής αναζήτησης.

3.1.1 Αλγόριθμοι Απληστίας (Greedy Algorithms)

Ένας άπληστος αλγόριθμος είναι ένας απλός, διαισθητικός αλγόριθμος που κάνει την τοπικά βέλτιστη επιλογή σε κάθε στάδιο με την ελπίδα να βρεθεί ένα ολικό ελάχιστο ή μέγιστο, αναλόγως τη φύση του προβλήματος και ανήκει στην κατηγορία των μυωπικών αλγορίθμων. Δημιουργεί μια λύση επιλέγοντας επαναληπτικά την τοπικά βέλτιστη επιλογή παράγοντας αποτέλεσμα σε πολυωνυμικό χρόνο . Ονομάζεται "άπληστος" επειδή γενικά επιλέγει τη λύση που φαίνεται καλύτερη εκείνη τη στιγμή χωρίς να λαμβάνει υπόψη την επίδραση που θα έχει στις μελλοντικές επιλογές με αποτέλεσμα πολλές φορές να μην μπορεί να εντοπίσει το ολικό ελάχιστο. Οι άπληστοι αλγόριθμοι δεν είναι πάντα η καλύτερη μέθοδος για την επίλυση ενός προβλήματος, αλλά είναι συχνά ιδιαίτερα χρήσιμοι. Οι δημοφιλέστεροι αλγόριθμοι απληστίας είναι οι παρακάτω:

- Αλγόριθμοι Πλησιέστερου Γείτονα (Nearest Neighborhood Algorithm).
- Αλγόριθμος της διαδικασίας εισαγωγής κόμβων (Nearest Insertion Algorithm).
- Αλγόριθμος του Christofides για το πρόβλημα του πλανόδιου πωλητή.
- Ο αλγόριθμος εξοικονομήσεων των Clarke and Write.
- Ο αλγόριθμος σαρώματος των Gillet & Miller

3.1.2 Προσεγγιστικοί Αλγόριθμοι

Ένας προσεγγιστικός αλγόριθμος είναι ένας αλγόριθμος που εντοπίζει μια λύση που βρίσκεται κοντά στη βέλτιστη λύση, αλλά όχι απαραίτητα την καλύτερη δυνατή λύση, σε εύλογο χρονικό διάστημα. Η ποιότητα της προσέγγισης μετριέται από το λόγο της λύσης που βρέθηκε από τον προσεγγιστικό αλγόριθμο προς τη βέλτιστη λύση, γνωστός ως λόγος προσέγγισης. Οι αλγόριθμοι προσέγγισης χρησιμοποιούνται συχνά για προβλήματα που είναι υπολογιστικά δυσεπίλυτα, όπως προβλήματα NP-hard, όπου η ακριβής λύση δεν είναι δυνατή σε εύλογο χρονικό διάστημα.

3.2 Αλγόριθμοι Τοπικής Αναζήτησης (Local Search)

Οι αλγόριθμοι τοπικής αναζήτησης αποτελούν μια οικογένεια αλγορίθμων βελτιστοποίησης που λειτουργούν κάνοντας επαναληπτικές μικρές αλλαγές σε μια τυχαία αρχική λύση με βασικό στόχο να βρεθεί μια καλύτερη λύση με μικρότερο κόστος. Βασίζεται στη μέθοδο δοκιμής και σφάλματος η οποία αποτελεί την αρχαιότερη μέθοδο βελτιστοποίησης. Ο αλγόριθμος ξεκινά με μια αρχική λύση και σε κάθε επανάληψη, δημιουργεί ένα σύνολο γειτονικών λύσεων κάνοντας μικρές αλλαγές στην τρέχουσα λύση χωρίς να επηρεάζονται από τη μνήμη των λύσεων. Στη συνέχεια, ο αλγόριθμος επιλέγει την καλύτερη γειτονική λύση ως τη νέα τρέχουσα λύση, κάνει αντικατάσταση της λύσης και συνεχίζει τη διαδικασία μέχρι να φτάσει σε ένα τοπικό βέλτιστο ή να ικανοποιηθεί ένα κριτήριο διακοπής έως ότου η λύση να μη βελτιώνεται περεταίρω. Οι αλγόριθμοι τοπικής αναζήτησης αν και δεν είναι συστηματικοί έχουν τα εξής χαρακτηστιστικά:

1. Απαιτούν ελάχιστη μνήμη

2. Εντοπίζουν συχνά λογικές λύσεις σε άπειρους συνεχείς χώρους καταστάσεων

Οι αλγόριθμοι τοπικής αναζήτησης βρίσκουν μεγάλη χρησιμότητα στα προβλήματα δρομολόγησης οχημάτων αφού καλούνται να ελαχιστοποιήσουν το συνολικό κόστος των διαδρομών. Μπορούν να διαχωριστούν σε δύο διαφορετικά είδη επίλυσης όπου το πρώτο είδος έχει ως στόχο τη βελτιστοποίηση της δρομολόγησης των οχημάτων σε μία διαδρομή και το δεύτερο όπου στοχεύουν στην βελτίωση μεταξύ 2 ή περισσότερων διαδρομών με την ανάθεση των πελατών σε διαφορετικά οχήματα.

Παρακάτω, παρουσιάζεται η δομή του γενικού αλγορίθμου της τοπικής αναζήτησης.

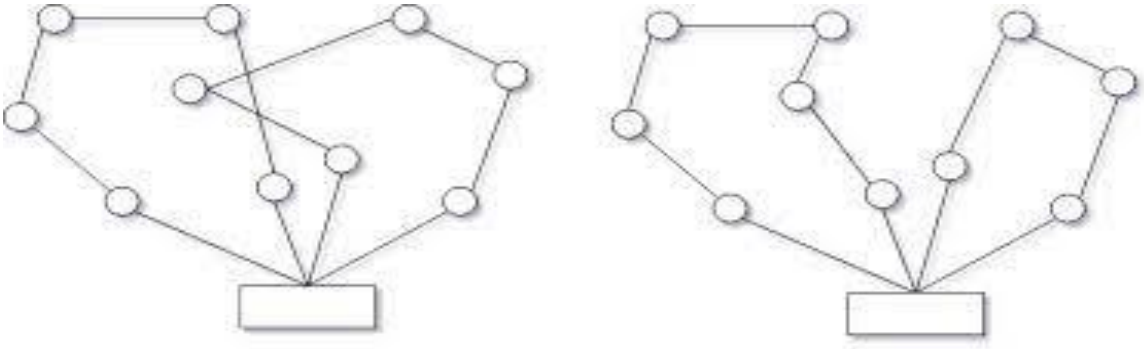
```
Local search

begin
t έστω μία αρχική λύση
do while βελτιώνεται η λύση,
improved(t)
t = improved(t)
return t
end
```

Εικόνα 3.1 Δομή αλγορίθμων Τοπικής Αναζήτησης

Κάποιες από τις πιο διαδεδομένες μεθόδους είναι η «1-0 relocate», η «1-1 Exchange» , η «2-Opt» , η «3-Opt» και η «Swap». Στη παρούσα διπλωματική εργασία θα χρησιμοποιήσουμε τους αλγορίθμους «1-0 relocate», «1-1 Exchange» και «2-Opt».

3.2.1 Μέθοδος 1-0 Relocate

Ο αλγόριθμος τοπικής αναζήτησης 1-0 Relocate είναι ένας αλγόριθμος ο οποίος προτάθηκε από τον Waters ως μία διαδικασία ανταλλαγής κόμβων. Πιο συγκεκριμένα μέσω αυτού του αλγορίθμου διαγράφεται ένας πελάτης από μία διαδρομή ενός οχήματος και επανατοποθετείται σε μία άλλη. Σε περίπτωση όπου προκύψει μικρότερο κόστος η αλλαγή αποθηκεύεται και μονιμοποιείται.

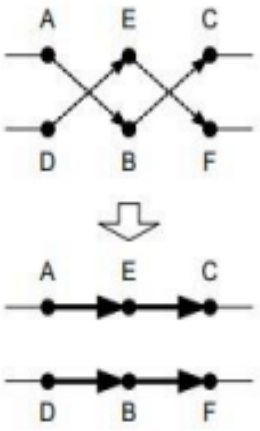


Εικόνα 3.3 Αλγόριθμος 1-0 Relocate

3.2.2 Μέθοδος 1-1 Exchange

Ο αλγόριθμος τοπικής αναζήτησης 1-1 Exchange προτάθηκε από τον Waters και έχει αρκετές ομοιότητες με τον αλγόριθμο 1-0 Relocate που αναλύσαμε παραπάνω. Ωστόσο, η βασική διαφορά στο συγκεκριμένο αλγόριθμο είναι ότι σε αυτή τη περίπτωση δε γίνεται διαγραφή ενός πελάτη-κόμβου από μία διαδρομή ενός οχήματος με σκοπό να προσθέσει σε μία άλλη διαδρομή ενός άλλου οχήματος, αλλά πραγματοποιείται μία ταυτόχρονη αλλαγή 2 πελατών από διαφορετικές διαδρομές με σκοπό την ελαχιστοποίηση του κόστους. Ο αλγόριθμος

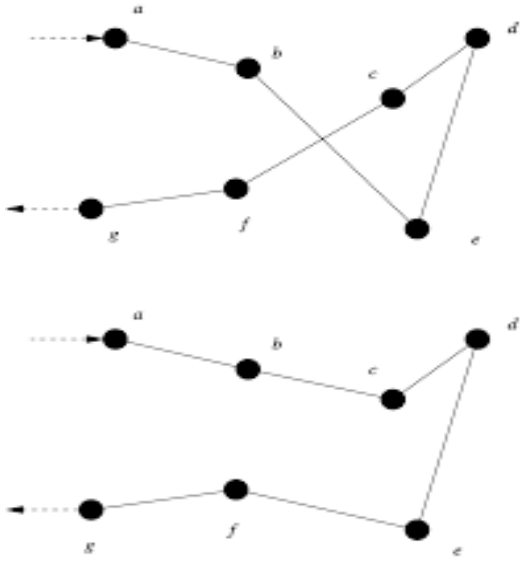
σταματάει όταν δεν μπορεί να γίνει πλέον κάποια άλλη αλλαγή ενώ, σε περίπτωση που προκύψει κάποια αλλαγή που επιφέρει μικρότερο κόστος αποθηκεύεται.



Εικόνα 3.2 Αλγόριθμος 1-1 Exchange

3.2.3 Μέθοδος 2-Opt

Ο αλγόριθμος 2-Opt προτάθηκε από τον Croes το 1958 σαν μια μέθοδος για τη βελτίωση της επίλυσης του Προβλήματος του Πλανόδιου Πωλητή (TSP) και άλλων παρόμοιων προβλημάτων βελτιστοποίησης. Λειτουργεί με την επαναληπτική εναλλαγή των άκρων-κόμβων σε μια διαδρομή με στόχο τη μείωση της συνολικής απόστασης της διαδρομής αυτής. Ο αλγόριθμος ξεκινά με μια αρχική περιήγηση και σε κάθε βήμα, εξετάζει όλες τις πιθανές εναλλαγές δύο άκρων-κόμβων και επιλέγει την εναλλαγή που οδηγεί στη συντομότερη περιήγηση. Η διαδικασία επαναλαμβάνεται έως ότου δεν μπορούν να γίνουν άλλες βελτιώσεις. Ο αλγόριθμος 2-Opt είναι ένας αλγόριθμος τοπικής αναζήτησης και είναι γνωστό ότι παράγει καλές λύσεις για μικρού έως μεσαίου μεγέθους περιπτώσεις TSP, VRP προβλημάτων, αλλά μπορεί να είναι αργός για μεγάλες περιπτώσεις.



Εικόνα 3.4 Αλγόριθμος 2-Opt

3.3 Μεθευρετικοί Αλγόριθμοι (Metaheuristics)

Οι απλοί ευρετικοί αλγόριθμοι συχνά δεν μας αποφέρουν τη βέλτιστη λύση στο πρόβλημα μας διότι είναι πολύ πιθανό να εγκλωβιστούν σε κάποιο τοπικό ελάχιστο με αποτέλσμα να μην μπορεί να βελτιωθεί άλλο μία λυση. Για αυτό το λόγο, επινοήθηκαν οι μεθευρετικοί αλγόριθμοι οι οποίοι αποτελούν έναν συνδιασμό των αλγορίθμων τοπικής αναζήτησης και υψηλότερου επιπέδου στρατηγικών έτσι ώστε να διευρύνουν το φάσμα των λύσεων σε μη εξερευνημένες περιοχές γύρω από τη «γειτονιά» μίας αρχικής μας λύσης ή επιπλέον και σε περιοχές που η λύση μας ήταν χειρότερη από πριν. Βάσει αυτής της διαδικασίας, ένας μεθευρετικός αλγόριθμος σταματάει πλεόν να «παγιδεύεται» σε τοπικά ελάχιστα και συγκλίνει ακόμα περισσότερο στη βέλτιστη λύση αφού η αποτελεσματικότητα τους βασίζεται αρκετά στην αυξημένη τυχαιότητα που τους χαρακτηρίζει. Υπάρχουν κάποιοι μεθευρετικοί

αλγόριθμοι οι οποίοι αξιοποιούν μία και μόνο αρχική λύση και υπάρχουν και αυτοί που χρησιμοποιούν έναν πληθυσμό λύσεων. Αυτό σημαίνει ότι οι πρώτοι αναζητώντας πιο έντονα στην περιοχή μίας συγκεκριμένης αρχικής λύσης θα δράσουν αποτελεσματικότερα στα πλαίσια της αναζήτησης της συγκεκριμένης γειτονιάς ενώ οι δεύτεροι που χρησιμοποιούν έναν πληθυσμό λύσεων θα εξερευνήσουν παραπάνω χώρους λιγότερο ενδελεχώς. Αξίζει να σημειωθεί ότι η ποιότητα των λύσεων που παράγουν οι μεθευρετικοί αλγόριθμοι είναι ανώτερη συγκριτικά με τους απλούς ευρετικούς αλγορίθμους για αυτό κιόλας η πλειοψηφία των προβλημάτων συνδιαστικής βελτιστοποίησης λύνονται μέσω των μεθευρετικών αλγορίθμων. Οι δημοφηλέστεροι μεθευρετικοί αλγόριθμοι είναι οι εξής:

- Η περιορισμένη αναζήτηση (Tabu search). Η διαδικασία άπληστης τυχαιοποιημένης προσαρμοστικής αναζήτησης (Greedy randomized adaptive search procedure).
- Ο αλγόριθμος της διασκορπισμένης αναζήτησης (Scatter search) .
- Η προσομοιωμένη ανόπτηση (Simulated annealing)
- Η περιορισμένη αναζήτηση (Tabu search).
- Η μεταβλητή αναζήτηση γειτονιάς (Variable neighborhood search)
- Τα νευρωνικά δίκτυα (Neural nets).

Στη παρούσα διπλωματική εργασία θα χρησιμοποιήσουμε τον μεθευρετικό αλγόριθμο της προσομοιωμένης ανόπτησης (Simulated Annealing) ο οποίος θα αναλυθεί εκτενώς στην επόμενη παράγραφο.

3.3.1 Προσομοιωμένη Ανόπτηση (Simulated Annealing)

Η προσομοιωμένη ανόπτηση αποτελεί μια πιθανολογική μέθοδο επίλυσης προβλημάτων συνδυαστικής βελτιστοποίησης. Η μέθοδος αυτή επινοήθηκε από τους Kirkpatrick, Gelett και

Vecchi το 1983 με κύριο σκοπό την εύρεση του ολικού ελαχίστου σε μια συνάρτηση κόστους με πληθώρα τοπικών ελαχίστων. Το όνομα και η έμπνευση προέρχονται από την ανόπτηση στη μεταλλουργία και τη θερμοδυναμική, μια τεχνική όπου ένα υλικό θερμαίνεται έως ότου να ξεπεράσει το σημείο τήξεως του και εν συνεχεία ψύχεται με έναν αργό ρυθμό ψύξης στοχεύοντας έτσι, αφού τελειώσει η διαδικασία της ψύξης, το στερεό να βρεθεί στο χαμηλότερο στάδιο ενέργειας δηλαδή στην πιο ευσταθή κατάσταση όπου η κινητικότητα των ατόμων έχει μειωθεί αρκετά. Συνεπώς ο αλγόριθμος της προσομοιωμένης ανόπτησης προσπαθεί, αντιγράφοντας το φυσικό φαινόμενο της ανόπτησης, να επιτρέψει την κίνηση κάποιου κριτηρίου στον εφικτό χώρο αναζήτησης ούτως ώστε όταν σταματήσει τη λειτουργία του να έχει καταλήξει στο ολικό ελάχιστο.

Στην πράξη ο αλγόριθμος προσομοιωμένης ανόπτησης εκμεταλλεύεται μία γειτονιά νέων καταστάσεων $N(s)$, που μπορεί να καταλήξει κάποιος από μία τρέχουσα κατάσταση s. Η διαδικασία του αλγορίθμου ξεκινάει λαμβάνοντας μια αρχική κατάσταση s_0 και μέσω μίας τυχαίας διαταραχής (στη παρούσα διπλωματική εργασία η τυχαία διαταραχή είναι ο αλγόριθμος τοπικής αναζήτησης 1-1 Exchange) του συστήματος παράγεται μία νέα λύση s'. Το αποτέλεσμα της αντικειμενικής συνάρτησης είναι το $\delta = f(s')$ - $f(s_0)$ όπου στην περίπτωση που το $\delta < 0$ τότε η καινούρια κατάσταση είναι πάντα αποδεκτή ενώ αν το $\delta \geq 0$ τότε η νέα κατάσταση είναι αποδεκτή βάσει της πιθανότητας $p(\delta) = e^{-\frac{\delta}{kt}}$. Η πιθανότητα αυτή βασίζεται στο νόμο της στατιστικής θερμοδυναμικής και το t δηλώνει τη θερμοκρασία, το δ την ποσότητα της ενέργειας μίας κατάστασης και τέλος το k αποτελεί τη σταθερά Boltzmann η οποία μπορεί να παραληφθεί διότι δε χρησιμεύει στη συνδυαστική βελτιστοποίηση και η πιθανότητα γίνεται $p(\delta) = e^{-\frac{\delta}{t}}$. Ο λόγος για τον οποίο μπορεί να γίνει αποδεκτή, βάσει της παραπάνω πιθανότητας, μία γειτονική λύση χειρότερη από τη τρέχουσα είναι για να μπορέσει

ο αλγόριθμος να απεγκλωβιστεί από κάποιο τοπικό ελάχιστο της αντικειμενικής συνάρτησης. Επίσης, όσο η θερμοκρασία μειώνεται σταδιακά η πιθανότητα αποδοχής μία «κακής» λύσης μειώνεται με τη σειρά της με το πέρασμα των επαναλήψεων του αλγορίθμου καταλήγοντας εύλογα στο συμπέρασμα ότι όσο πιο υψηλή είναι η θερμοκρασία και το σύστημα δεν βρίσκεται σε ενεργειακή ισορροπία τόσο πιο πιθανό είναι να γίνει αποδεκτή μία λύση με χειρότερη τιμή στην αντικειμενική συνάρτηση ως την νέα κατάσταση του προβλήματος. Ο ρυθμός μείωσης της θερμοκρασίας(ψύχρανση) εκφράζεται από μία συνάρτηση $\alpha(t)$ αναπαρίσταται με τέσσερις διαφορετικούς τρόπους οι οποίοι είναι οι εξής:

- Εκθετική μείωση, όπου $a(t) = t \times e^{-a \times i}$ ή $\alpha(t) = t \times a^{i}$ ή $a(t) = t \times a$ όπου $\alpha = 0.95$ ή $\alpha = 0.98$
- Καθορισμένου μήκους συνάρτηση $T(i) = T_0 - a \times i$ όπου $0.8 \leq \alpha \leq 0.98$
- Προσαρμοστικό πρόγραμμα ψύχρανσης, όπου η επιλογή του t είναι δυναμική
- Λογαριθμική προσέγγιση

Η αρχική θερμοκρασία t_0 θα πρέπει να είναι τόσο υψηλή έτσι ώστε να επιτρέπεται η κίνηση του αλγορίθμου σε όλες τις πιθανές καταστάσεις διότι σε διαφορετική περίπτωση είναι πολύ πιθανό η τελική μας λύση να είναι σχεδόν ίδια με την αρχική μας λύση. Ωστόσο, σε περίπτωση που η θερμοκρασία εκκίνησης είναι πολύ υψηλή η κίνηση σε γειτονιές στο χώρο μετατρέπεται, κυρίως στις αρχικές επαναλήψεις, σε μία τυχαία αναζήτηση και θα παραμείνει τυχαία έως ότου η θερμοκρασία αγγίξει τα επίπεδα όπου ο αλγόριθμος θα ξεκινήσει να δρα στα πλαίσια της προσομοιωμένης ανόπτησης. Η τελική θερμοκρασία θα περιμέναμε να φτάσει στο μηδέν όμως αυτό θα κάνει τον αλγόριθμο μας να τρέχει για ένα πολύ μεγάλο χρονικό διάστημα, συνεπώς το κριτήριο διακοπής ψύξης μπορεί να είναι είτε σε μία θερμοκρασία πολύ κοντά στο

μηδέν ή στην θερμοκρασία αυτήν όπου το σύστημα «παγώσει» και δεν γίνονται αποδεκτές ούτε καλύτερες αλλά ούτε χειρότερες λύσεις.

Τα βασικότερα κριτήρια διακοπής τους αλγορίθμου, όπου υπάρχει σύγκλιση στο ολικό ελάχιστο είναι τα παρακάτω:

- Στην περίπτωση που η θερμοκρασία έχει φτάσει σε ένα καθορισμένο χαμηλό σημείο εξαιρουμένου του μηδενός.
- Η αναλογία των αποδεχτών κινήσεων σε γειτονικές περιοχές είναι κάτω από μία δεδομένη τιμή.
- Ύστερα από έναν καθορισμένο αριθμό επαναλήψεων όπου δεν έχει υπάρξει κάποια αποδοχή στην αντικειμενική συνάρτηση.
- Ύστερα από έναν προκαθορισμένο μέγιστο αριθμό επαναλήψεων που έχουν ολοκληρωθεί.

Τέλος, μπορούμε να ρυθμίσουμε κάποιους βασικούς παράγοντες και παράμετρος κατά τη διάρκεια της διαδικασίας του αλγορίθμου αποσκοπώντας στην ταχύτερη εκτέλεση του προγράμματος αλλά και στην εύρεση καλύτερων αποτελεσμάτων. Οι παράμετροι αυτοί είναι το πεδίο αναζήτησης της λύσης, ο αριθμός των εσωτερικών επαναλήψεων και η σωστή επιλογή της συνάρτησης μείωσης της θερμοκρασίας. Ο ψευδοκώδικας που αναπαριστά με μεγαλύτερη ακρίβεια τον αλγόριθμο της προσομοιωμένης ανόπτησης είναι ο παρακάτω:

select μία αρχική λύση s_0

select μία αρχική θερμοκρασία t_0

select μία συνάρτηση μείωσης της θερμοκρασίας $a(t)$

repeat

repeat

Τυχαία επιλογή μιας γειτονιάς $s \in N(s_0)$

$\delta = f(s)-f(s_0)$

if $\delta<0$ **then**

$s_0=s$

else

δημιουργούμε τυχαία x, ομοιόμορφα κατανεμημένα στην ακτίνα (0,1)

if $x<e^{-\delta/t}$ **then**

$s_0=s$

endif

endif

until ο μέγιστος αριθμός επαναλήψεων ολοκληρωθεί

$t=a(t)$

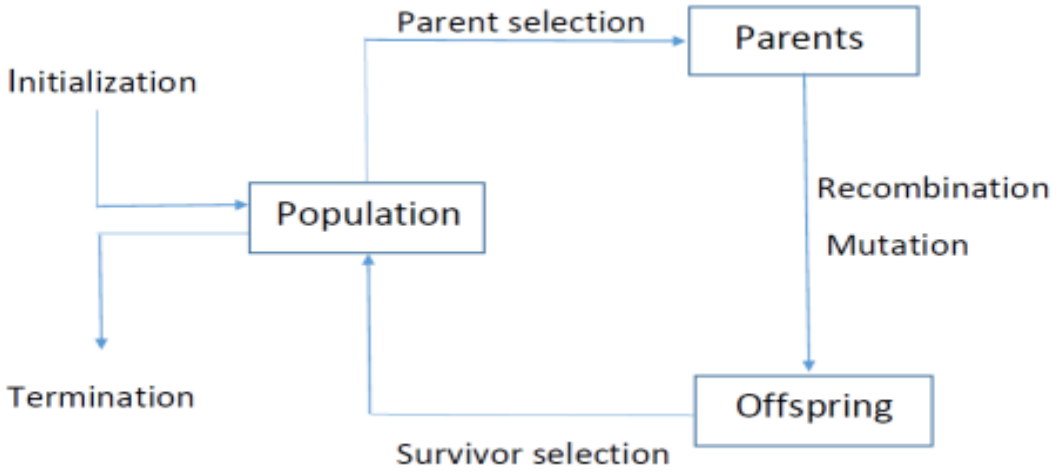
until κάποιο κριτήριο τερματισμού ικανοποιηθεί

3.4 Εξελικτικοί Αλγόριθμοι

Οι Εξελικτικοί αλγόριθμοι(Evolutionary Algorithms-EA) αποτελούν μεθευρετικές μεθόδους βελτιστοποίησης οι οποίοι στην ουσία βασίζονται στη μίμηση της διαδικασίας της φυσικής εξέλιξης και της βιολογίας. Αν και οι ρίζες των συγκεκριμένων αλγορίθμων προέρχονται από τον Κάρολο Δαρβίνο και το έργο του “The Origin of Species” που δημοσιεύτηκε το 1859, στη πιο σύγχρονη εποχή, το 1960, ο δημιουργός του πρώτου γενετικού αλγόριθμου που αποτελεί τα θεμέλια των σημερινών εξελικτικών αλγορίθμων είναι ο John Holland. Οι εξελικτικοί αλγόριθμοι μιμούνται τις φυσικές διαδικασίες της επιλογής, της μετάλλαξης, της αναπαραγωγής και της διασταύρωσης και τις εκμεταλλεύονται ως μηχανισμούς αναζήτησης αποσκοπώντας στην εύρεση της καλύτερης λύσης σε προβλήματα συνδυαστικής βελτιστοποίησης η οποία προέρχεται από την οργανωμένη ανταλλαγή πληροφοριών(γονιδίων ή μετάλλαξη). Δηλαδή η επιβίωση του ικανότερου στη φύση συγχέεται με την εύρεση της καλύτερης λύσης μέσα σε ένα πληθυσμό πιθανών λύσεων ενός προβλήματος που ονομάζονται άτομα. Σκοπός αυτής της στοχαστικής επαναληπτικής διαδικασίας είναι η ένωση δύο διαφορετικών λύσεων με στόχο τη δημιουργία μίας νέας καλύτερης λύσης. Για να καθορίσουμε τη δομή ενός εξελικτικού αλγορίθμου πρέπει πρώτα να αναλύσουμε ένα σύνολο διαδικασιών οι οποίες είναι:

1. Αρχικοποίηση
2. Αντιπροσώπευση (Καθορισμός των ατόμων)
3. Συνάρτηση Αξιολόγησης
4. Πληθυσμός
5. Μηχανισμός επιλογών γονέων
6. Τελεστές διασταύρωσης και μετάλλαξης

7. Μηχανισμός επιλογής επιβίωσης
8. Κριτήρια τερματισμού



Εικόνα 3.5 Δομή Εξελικτικών Αλγορίθμων

Οι βασικότεροι από τους εξελικτικούς αλγορίθμους είναι οι εξής:

- Οι Γενετικοί Αλγόριθμοι (Genetic Algorithms)
- Οι Υβριδικοί Γενετικοί Αλγόριθμοι ή Μιμητικοί (Hybrid Genetic Algorithms or Memetic Algorithms)
- Οι Γενετικοί Αλγόριθμοι Πολλαπλών Πληθυσμών-Νησιών (Island Genetic Algorithms)
- Η Διαφορική Εξέλιξη (Differential Evolution)

3.5 Αλγόριθμοι Εμπνευσμένοι από τη Φύση

Οι αλγόριθμοι εμπνευσμένοι από τη φύση ανήκουν στην κατηγορία των μεθευρετικών αλγορίθμων βελτιστοποίησης αλλά και στην κατηγορία των αλγορίθμων Υπολογιστικής Νοημοσύνης. Η φύση έχει την ικανότητα να λύνει μόνη της τα προβλήματα που παρουσιάζονται μπροστά της αφού διαθέτει κάποια χαρακτηριστικά όπως αυτά της αυτό-βελτίωσης, της αυτό-διδασκαλίας, της αυτό-θεραπείας και της αυτό-επεξεργασίας (Yang,2018). Συνεπώς, μέσα από τη μελέτη των φαινομένων της φύσης και της συμπεριφοράς ορισμένων ζωντανών οργανισμών ο άνθρωπος κατάφερε να τα μοντελοποιήσει και να δημιουργήσει αλγορίθμους με σκοπό την επίλυση δύσκολων προβλημάτων βελτιστοποίησης. Σύμφωνα με τον Fister Jr. (2013) οι εμπνευσμένοι από τη φύση αλγόριθμοι μπορούν να χωριστούν σε τέσσερις διαφορετικές κατηγορίες που είναι οι εξής:

- Βιο-εμπνευσμένοι αλγόριθμοι (Bio-inspired Algorithms), οι οποίοι βασίζονται σε βιολογικές διαδικασίες.
- Φυσικο-Χημικοί αλγόριθμοι (Physical Phenomena and laws of science), οι οποίοι βασίζονται στους νόμους της φυσικής και της χημείας.
- Αλγόριθμοι Σμήνους (Swarm Intelligence), οι οποίοι προσομοιώνουν τη νοημοσύνη σμήνους.
- Άλλοι (Others), οι οποίοι δεν ανήκουν σε καμία από τις προηγούμενες κατηγορίες

Οι πιο δημοφιλείς εμπνευσμένοι από τη φύση αλγόριθμοι είναι οι παρακάτω:

- Ο αλγόριθμος βελτιστοποίησης σμήνους σωματιδίων (Particle Swarm Optimization)
- Ο αλγόριθμος βελτιστοποίησης αποικίας μυρμηγκιών (Ant colony optimization)
- Ο αλγόριθμος τεχνητής αποικίας μελισσών (Artificial bee colony optimization)
- Ο αλγόριθμος της πυγολαμπίδας (Fire-fly algorithm)

- Ο αλγόριθμος της νυχτερίδας (Bat algorithm)
- Ο αλγόριθμος βελτιστοποίησης μπάμπουρων (Bumble bees mating algorithm)
- Αλγόριθμος αναζήτησης κούκων (Cuckoo search algorithm)

Κεφάλαιο 4 – Περιγραφή και Ανάλυση του Προβλήματος

4.1 Εισαγωγή στο Πρόβλημα

Στη παρούσα διπλωματική εργασία το πρόβλημα το οποίο επιλύεται είναι το ανοιχτό-κλειστό πρόβλημα δρομολόγησης οχημάτων με ενοικιαζόμενα και ιδιόκτητα οχήματα και πολλαπλές επιστροφές στην αποθήκη (Close-Open Vehicle Routing Problem). Στο συγκεκριμένο πρόβλημα, σε αντίθεση με τα OVRP και CVRP ένα όχημα, σε περίπτωση που δεν ξεπεράσει τα χρονικά όρια που του αναλογούν, έχει τη δυνατότητα να επιστρέψει στην αποθήκη-αφετηρία, να ξαναφορτώσει και να συνεχίσει σε για μία νέα διαδρομή προς τους πελάτες-κόμβους μέχρι να εξαντληθούν τα χρονικά όρια που μπορεί να είναι εν κινήσει. Τα οχήματα σε περίπτωση που είναι ιδιόκτητα οφείλουν να τερματίσουν τις διαδρομές τους στην αποθήκη ενώ τα ενοικιαζόμενα δεν είναι υποχρεωμένα να τερματίσουν στην αποθήκη. Είναι σημαντικό να σημειωθεί ότι τα ιδιόκτητα οχήματα έχουν μικρότερο κόστος από τα ενοικιαζόμενα. Στόχος μας είναι μέσω ευρετικών και μεθευρετικών αλγορίθμων να ελαχιστοποιήσουμε το συνολικό κόστος των διαδρομών που θα πραγματοποιήσουν τα οχήματα για τη βέλτιστη εξυπηρέτηση των πελατών βάσει των περιορισμών του προβλήματος. Ο κώδικας χωρίζεται σε τρία στάδια, όπου το πρώτο στάδιο είναι η εύρεση μίας αρχικής λύσης μέσω της χρήσης του αλγορίθμου του Πλησιέστερου Γείτονα, το δεύτερο στάδιο είναι η βελτιστοποίηση με τη χρήση των

αλγορίθμων τοπικής αναζήτησης 2-Opt και 1-0 Relocate, και το τρίτο στάδιο είναι η τελική βελτιστοποίηση με τη χρήση του μεθευρετικού αλγορίθμου της Προσομοιωμένης Ανόπτησης όπου για την υλοποίηση του θα χρησιμοποιηθεί σαν τυχαία διαταραχή του συστήματος ο αλγόριθμος τοπικής αναζήτησης 1-1 Exchange. Η γλώσσα προγραμματισμού που χρησιμοποιήθηκε για την υλοποίηση του κώδικα είναι η Python 3.10.

4.2 Δεδομένα και Επεξεργασία

Αρχικά για την υλοποίηση του κώδικα είναι απαραίτητο να εισάγουμε, με την εντολή “import”, τις απαραίτητες βιβλιοθήκες ώστε να μπορούμε να χρησιμοποιήσουμε κάποιες αναγκαίες συναρτήσεις για την ολοκλήρωση του κώδικα. Οι βιβλιοθήκες που κάναμε “import” είναι οι εξής :

- Numpy, για τη χρήση πινάκων
- Pandas, για να μπορεί το πρόγραμμα να διαβάσει δεδομένα σε μορφή Excel
- Scipy, για τη χρήση αριθμητικών αλγορίθμων και γενικότερα τη χρήση εργαλείων γραμμικής άλγεβρας
- Operator, για τη χρήση μαθηματικών και συγκριτικών συμβόλων
- Random, για τη χρήση συναρτήσεων που παράγουν τυχαίους αριθμούς και συνδυασμούς τυχαίων αριθμών
- Math, για τη χρήση μαθηματικών συναρτήσεων
- Matplotlib, για τη γραφική απεικόνιση των αποτελεσμάτων του κώδικα

Έπειτα, τα απαραίτητα δεδομένα(τα οποία ήταν αποθηκευμένα σε ένα αρχείο Excel) για τη λύση του συγκεκριμένου προβλήματος είναι τα παρακάτω:

Πλήθος Πελατών	**N**
Οι γεωγραφικές συντεταγμένες των πελατών	**XYa**
Η ζήτηση του κάθε πελάτη	**demand**
Η μέγιστη χωρητικότητα του κάθε οχήματος	**Q_max**
Ο μέγιστος διαθέσιμος χρόνος του κάθε οχήματος	**T_max**
Ο αριθμός ιδιόκτητων οχημάτων	**Nu**
Ο χρόνος εξυπηρέτησης ενός πελάτη	**s_time**
Ο χρόνος καθυστέρησης εκφόρτωσης ενός οχήματος	**delay_time**
Σταθερό κόστος ιδιόκτητου οχήματος	F_0
Σταθερό κόστος ενοικιαζόμενου οχήματος	F_1

Πίνακας 4.1 Επεξήγηση των δεδομένων του προβλήματος

Χρειάζεται να σημειωθεί ότι ο κόμβος που αντιπροσωπεύει την αποθήκη-αφετηρία έχει μηδενική ζήτηση και ότι ο αριθμός των ενοικιαζόμενων οχημάτων δεν έχει κάποιο όριο αφού εξαρτάται από το πλήθος των πελατών που πρέπει να εξυπηρετηθούν. Έπειτα, για να υπολογιστεί η απόσταση μεταξύ δύο πελατών-κόμβων χρησιμοποιήσαμε τον τύπο της ευκλείδειας απόστασης αφού θεωρούμε τις αποστάσεις μεταξύ των κόμβων ως ευθείες. Στο

συγκεκριμένο πρόβλημα οι αποστάσεις μεταξύ δύο κόμβων αντιπροσωπεύουν το κόστος μετάβασης από τον έναν πελάτη στον άλλον. Ο τύπος της ευκλείδειας απόστασης είναι ο εξής:

- **Euclidean distance** = $\sqrt{(X_2 - X_1)^2 + (Y_2 - Y_1)^2}$ **, όπου (**$X_{i,}, Y_i$**) είναι η συντεταγμένη του σημείου** i**.**

Έπειτα αφού υπολογίσουμε όλες τις αποστάσεις μεταξύ όλων των κόμβων δημιουργούμε έναν πίνακα που τον ονομάζουμε "costos", $N \times N$ που αρχικά είναι μηδενικός και στη συνέχεια καταχωρούμε στον πίνακα αυτόν τις αποστάσεις. Ο πίνακας αυτός είναι συμμετρικός με όλα τα στοιχεία της κύριας διαγωνίου του να είναι μηδενικά εφόσον δεν μπορεί ένα όχημα να παραμείνει στον ίδιο κόμβο πέρα από την αποθήκη αλλά και με όλα τα στοιχεία της πρώτης στήλης του πίνακα να είναι μηδενικά αφού τα ενοικιαζόμενα οχήματα δεν χρειάζεται να τερματίσουν τη διαδρομή τους στην αποθήκη. Ωστόσο, προγραμματιστικά έχει ληφθεί υπόψιν το κόστος επιστροφής των ιδιόκτητων οχημάτων να συμπεριλαμβάνεται στο συνολικό κόστος των διαδρομών καθώς και το σταθερό κόστος των ιδιόκτητων οχημάτων F_0 = 50 αλλά και το σταθερό κόστος των ενοικιαζόμενων οχημάτων F_1 = 150. Τέλος αξίζει να σημειωθεί ότι το συγκεκριμένο πρόβλημα δρομολόγησης οχημάτων δεν έχει αναλυθεί ενδελεχώς στην παγκόσμια βιβλιογραφία για αυτόν το λόγο τα δεδομένα που χρησιμοποιήθηκαν έχουν δημιουργηθεί για το Πρόβλημα Δρομολόγησης Οχημάτων Περιορισμένης Χωρητικότητας με αποτέλεσμα να μην υπάρχει ο περιορισμός της μέγιστης απόστασης που μπορεί να διανύσει ένα όχημα. Συνεπώς, χρησιμοποιήθηκε η προσέγγιση των Ran Liu και Zhibin Jiang στη δημοσίευση "The close-open mixed routing problem" στην οποία όλες οι μονάδες των μεταβλητών παραμένουν ίδιες, ο χρόνος εξυπηρέτησης των πελατών και ο χρόνος καθυστέρησης εκφόρτωσης είναι μηδενικός ενώ η μέγιστη απόσταση που μπορεί να διανύσει ένα όχημα παράγεται από την παρακάτω σχέση:

$$\textbf{T\_max} = \mathbf{0,2} \times \mathbf{2} \times \boldsymbol{max}_{iev}(\boldsymbol{c}_{0i}) + \mathbf{0,8} \times \boldsymbol{r}_{cw}$$

4.3 Εύρεση Αρχικής Λύσης – Αλγόριθμος Πλησιέστερου Γείτονα

Ο αλγόριθμος του Πλησιέστερου γείτονα ανήκει στην κατηγορία των ευρετικών αλγορίθμων απληστίας και αποτελεί έναν ιδανικό αλγόριθμο για την εύρεση αρχικής λύσης σε προβλήματα συνδυαστικής βελτιστοποίησης. Στο συγκεκριμένο πρόβλημα δρομολόγησης οχημάτων αποτελεί το πρώτο στάδιο του κώδικα και σκοπός του αλγορίθμου είναι, σύμφωνα με τους περιορισμούς χωρητικότητας, χρόνου και πλήθους ιδιόκτητων οχημάτων, να αναζητηθούν και να βρεθούν οι εφικτές διαδρομές για την εξυπηρέτηση όλων των πελατών. Η διαδικασία που ακολουθεί ο αλγόριθμος είναι η εξής:

‘Ένα όχημα ξεκινάει από την αποθήκη-αφετηρία και στη συνέχεια επισκέπτεται τον πελάτη-κόμβο που είναι πλησιέστερος σε αυτήν. Εφόσον αυτός ο πελάτης-κόμβος εξυπηρετηθεί το όχημα συνεχίζει και κατευθύνεται στον επόμενο πλησιέστερο πελάτη που δεν έχει επισκεφτεί ως τώρα και αυτή η διαδικασία συνεχίζεται έως ότου εξυπηρετηθούν όλοι οι πελάτες. Όταν ένα όχημα πραγματοποιήσει την διαδρομή του λόγω των περιορισμών χωρητικότητας και χρόνου αναγκάζεται να γυρίσει στην αποθήκη σε περίπτωση που είναι ιδιόκτητο αλλιώς εάν είναι ενοικιαζόμενο δεν επιστρέφει σε αυτήν. Στην περίπτωση που παραβιαστεί μόνο ο περιορισμός της χωρητικότητας και όχι ο χρονικός περιορισμός το όχημα, ιδιόκτητο ή μη επιστρέφει στην αποθήκη, φορτώνει νέο εμπόρευμα και συνεχίζει μέχρι να εξαντληθούν τα χρονικά του όρια όπου η τελική κατάληξη και πάλι εξαρτάται από το αν είναι ιδιόκτητο ή ενοικιαζόμενο. Ο βασικός επαναληπτικός βρόγχος του αλγορίθμου έχει ως στόχο να ταξινομήσει τους πελάτες βάσει το διάνυσμα της απόστασης (χρόνου) και εκτελείται μέχρι να ελεγχθούν όλοι οι πελάτες. Σε κάθε βήμα του αλγορίθμου ελέγχονται οι περιορισμοί και σε περίπτωση παραβίασης κάποιου περιορισμού ένα νέο όχημα ξεκινάει από την αποθήκη-αφετηρία για την εκτέλεση μίας νέας διαδρομής.

Οι μεταβλητές, τα διανύσματα και οι πίνακες που ορίσαμε για την εκτέλεση του αλγορίθμου αναφέρονται διεξοδικά στον παρακάτω πίνακα:

i	**Πλησιέστερος πελάτης**
c	**Μετρητής κόμβων-πελατών**
m	**Μετρητής στήλης μονοπατιού οχήματος**
QPath[ ]	**Διάνυσμα αποθήκευσης της ζήτησης κάθε διαδρομής**
TPath[ ]	**Διάνυσμα αποθήκευσης της χρονικής διάρκειας κάθε διαδρομής**
Path[ ]	**Πίνακας αποθήκευσης των διαδρομών όλων των οχημάτων**
forthgo	**Όχημα μίας διαδρομής**
flag	**Μετρητής που στην περίπτωση που ισούται με 1 το όχημα είναι ιδιόκτητο ενώ σε περίπτωση που ισούται με 0 το όχημα είναι ενοικιαζόμενο**
min_Y	**Ελάχιστη απόσταση από έναν κόμβο σε έναν άλλον**
Komvos	**Επόμενος πελάτης**

Πίνακας 4.2 Επεξήγηση των μεταβλητών της συνάρτησης του Πλησιέστερου Γείτονα

Αφού αναλύσαμε παραπάνω τη λειτουργία του αλγορίθμου του Πλησιέστερου Γείτονα, θα περιγράψουμε τους περιορισμούς που διέπουν το πρόβλημα, οι οποίοι είναι:

- T_Path[ ] ≤ T_max , όπου στο διάνυσμα T_Path[ ] αθροίζονται οι συνολικές αποστάσεις δηλαδή ο συνολικός χρόνος που ένα όχημα μπορεί να διανύσει. Στη περίπτωση που το όχημα είναι ιδιόκτητο στο διάνυσμα αυτό προστίθεται και η απόσταση(χρόνος) που θα κάνει το όχημα για να επιστρέψει στην αποθήκη με τη βοήθεια του μετρητή flag που αναφέραμε παραπάνω.
- Q_Path[ ] ≤ Q_max, όπου στο διάνυσμα Q_Path αθροίζεται η συνολική ζήτηση που έχει εξυπηρετήσει ένα όχημα και μας εξασφαλίζει ότι ένα όχημα δε θα ξεπεράσει ποτέ το επιτρεπτό όριο ζήτησης. Το συγκεκριμένο διάνυσμα μηδενίζεται στη περίπτωση που σε ένα όχημα περισσέψει χρόνος, επιστρέψει στην αποθήκη και ξαναφορτώσει με σκοπό να ξεκινήσει μία νέα διαδρομή μέχρι να φτάσει στο επιτρεπτό χρονικό όριο.
- Αριθμός ιδιόκτητων οχημάτων ≤ Nu , όπου αυτός ο περιορισμός μας εξασφαλίζει ότι θα χρησιμοποιηθούν ακριβώς Nu ιδιόκτητα οχήματα.

Αφού ορίσουμε ως σημείο εκκίνησης κάθε διαδρομής την αποθήκη και θεωρήσουμε ότι η αν η μεταβλητή “forthgo" $\leq Nu$ τότε “flag” = 0 χρησιμοποιούμε για την εύρεση της ελάχιστης ευκλείδειας απόστασης μεταξύ δύο πελατών-κόμβων αλλά και τη θέση των κόμβων αυτών τις παρακάτω εντολές:

1. **min_Y = np.amin(costos[0,:])**
2. **komvos = np.argmin(costos[0,:])**

Αφού λοιπόν βρεθεί ο πλησιέστερος κόμβος ο οποίος ικανοποιεί τους περιορισμούς του προβλήματος προστίθεται στη διαδρομή του οχήματος και τότε i = komvos. Ο επαναληπτικός βρόγχος τερματίζει όταν ελεγχθούν όλοι οι πελάτες. Όταν εξυπηρετείται ένας πελάτης αντικαθιστούμε στον πίνακα path[ ] την απόσταση μεταξύ δύο πελατών με inf έτσι ώστε να μην μπερδευτεί ο αλγόριθμος και εξυπηρετήσει ξανά έναν πελάτη. Όταν έχουν εξυπηρετηθεί

όλοι οι πελάτες ο πίνακας μας, θα έχει σε κάθε κελί inf για αυτό πρέπει να του ξαναπεράσουμε τις νέες αποστάσεις των διαδρομών. Για τον υπολογισμό του κόστους της κάθε διαδρομής, του συνολικού κόστους όλων των διαδρομών καθώς για την γραφική απεικόνιση τους χρησιμοποιήθηκαν οι παρακάτω βοηθητικές συναρτήσεις:

1. **Row_Cost_Calculator(row, costos)**, για τον υπολογισμό της κάθε διαδρομής ξεχωριστά.
2. **Cost_Calculator(path**, **costos),** για τον υπολογισμό όλων των διαδρομών συνολικά.
3. **Plot_coords(cords, path),** για την γραφική απεικόνιση των αποτελεσμάτων.

Με τη χρήση των παραπάνω συναρτήσεων καταλήγουμε στην αρχική εφικτή λύση.

4.4 Βελτιστοποίηση Αρχικής Λύσης – Αλγόριθμοι Τοπικής Αναζήτησης

Όπως αναφέραμε και στο Κεφάλαιο 3 οι αλγόριθμοι τοπικής αναζήτησης αποτελούν το δεύτερο στάδιο του κώδικα, αυτό της βελτίωσης της αρχικής λύσης που βρήκαμε από τον αλγόριθμο απληστίας του Πλησιέστερου Γείτονα. Οι αλγόριθμοι τοπικής αναζήτησης χρησιμοποιήθηκαν σαν συναρτήσεις, πέρα από τον 1-1 Exchange αφού τους χρειαστήκαμε σε διαφορετικά σημεία του κώδικα μας. Οι αλγόριθμοι αυτοί είναι οι 2-Opt, 1-1 Exchange και 1-0 relocate. Οι αλγόριθμοι 2-Opt και 1-0 Relocate χρησιμοποιήθηκαν αμέσως μετά τη δημιουργία αρχικής λύσης ενώ ο αλγόριθμος 1-1 Exchange χρειάστηκε να αποτελέσει την τυχαία διαταραχή του συστήματος κατά την υλοποίηση του μεθευρετικού αλγορίθμου της προσομοιωμένης ανόπτησης. Τα δεδομένα των αλγορίθμων είναι ο πίνακας των διαδρομών της αρχικής λύσης, το κόστος κάθε διαδρομής ξεχωριστά αλλά και το συνολικό κόστος όλων των διαδρομών και η συνολική ζήτηση της κάθε διαδρομής.

1. Ο αλγόριθμος 2-Opt λειτουργεί σαν συνάρτηση η οποία είναι η: **two_opt(path, costos)**. Οι επαναλήψεις του αλγορίθμου είναι τόσες ώστε σε μία διαδρομή ενός οχήματος να αλλάξουν θέση όλοι οι κόμβοι μεταξύ τους εκτός της αποθήκης-αφετηρίας και του τελευταίου κόμβου έως ότου να βρεθεί μία διαδρομή με μικρότερο κόστος, και αυτό συμβαίνει για όλες τις διαδρομές των οχημάτων. Σε περίπτωση όπου σε μία διαδρομή ο αλγόριθμος δεν μπορεί να βρει μικρότερο κόστος ο αλγόριθμος σταματάει. Να σημειωθεί ότι οι αλλαγές γίνονται μεταξύ της ίδιας διαδρομής και όχι μεταξύ διαφορετικών διαδρομών. Όταν ο αλγόριθμος πραγματοποιήσει την αλλαγή 2 κόμβων μεταξύ τους και οι περιορισμοί δεν παραβιάζονται ελέγχει αν το κόστος της διαδρομής είναι μικρότερο από το προηγούμενο κόστος τότε η συγκεκριμένη αλλαγή αποθηκεύεται και ο αλγόριθμος συνεχίζει ψάχνοντας για περαιτέρω αλλαγές ώσπου να τερματίσει.
2. Και ο αλγόριθμος 1-0 Relocate λειτουργεί σαν συνάρτηση η οποία είναι η: **One_zero_Relocate(path, costos)**. Στόχος του συγκεκριμένου αλγορίθμου είναι να μειώσουμε τις συνολικές διαδρομές των οχημάτων και να προσεγγίσουμε τον ελάχιστο αριθμό διαδρομών. Αρχικά για τη λειτουργία του αλγορίθμου χρειάστηκε να μετατρέψουμε τις σειρές του πίνακα path σε λίστες ώστε να μπορέσουμε να χρησιμοποιήσουμε την εντολή “pop(i)” για τη διαγραφή ενός κόμβου και την εντολή “append(i)” για την μετατόπιση του σε μία άλλη διαδρομή. Ο αλγόριθμος εφαρμόζεται μεταξύ διαφορετικών διαδρομών και δεν επανατοποθετεί ποτέ την αποθήκη και τα μηδενικά κελιά μίας λίστας. Οι επαναλήψεις του αλγορίθμου είναι τόσες ώστε να διαγραφούν και να επανατοποθετηθούν όλοι οι κόμβοι από όλες τις λίστες σε όλες τις πιθανές θέσεις. Στη περίπτωση όπου η διαγραφή ενός κόμβου και η μετατόπιση του σε μία άλλη διαδρομή, πρώτον δεν παραβιάζει τους περιορισμούς του προβλήματος και

δεύτερον αποφέρει μικρότερο κόστος από αυτό που προϋπήρχε τότε, η αλλαγή αποθηκεύεται. Ο αλγόριθμος τερματίζει όταν ελεγχθούν όλες οι πιθανές αλλαγές. Τέλος αφού γίνουν οι αλλαγές μετατρέπουμε τις λίστες ξανά σε έναν πίνακα με όλες τις νέες μας διαδρομές.

3. Σε αντιδιαστολή με τους παραπάνω αλγορίθμους τοπικής αναζήτησης που χρησιμοποιήθηκαν σαν συναρτήσεις ο αλγόριθμος 1-1 Exchange εφαρμόστηκε σαν την τυχαία διαταραχή συστήματος για την υλοποίηση του μεθευρετικού αλγορίθμου της προσομοιωμένης ανόπτησης. Η φιλοσοφία του συγκεκριμένου αλγορίθμου είναι να ανταλλάξουμε όλους τους κόμβους-πελάτες διαφορετικών διαδρομών μεταξύ τους και στη περίπτωση που ικανοποιούνται οι απαραίτητοι περιορισμοί και το συνολικό κόστος μίας διαδρομής είναι μικρότερο από το προηγούμενο τότε η λύση αποθηκεύεται. Οι επαναλήψεις του αλγορίθμου είναι τόσες ώστε να ανταλλαχθούν όλοι οι κόμβοι μεταξύ τους για όλες τις διαδρομές πέρα από την αποθήκη-αφετηρία. Στο συγκεκριμένο αλγόριθμο υπάρχει η πιθανότητα ο αλγόριθμος να εγκλωβιστεί σε τοπικό ελάχιστο με αποτέλεσμα μετά από έναν αριθμό επαναλήψεων ο αλγόριθμος να τερματίσει αφού δεν υπάρχει περιθώριο βελτιστοποίησης της λύσης.

4.5 Εύρεση Τελικής Λύσης – Αλγόριθμος Προσομοιωμένης Ανόπτησης

Στο τρίτο στάδιο του κώδικα, υλοποιείται ο μεθευρετικός αλγόριθμος της προσομοιωμένης ανόπτησης, με τον οποίο θα παράξουμε και την τελική λύση του προβλήματος μας. Ο αλγόριθμος καλείται ως συνάρτηση στο πρόγραμμα η οποία είναι: **Simulated_annealing (path, initial_cost)**. Αρχικά, έξω από τον επαναληπτικό βρόγχο της προσομοιωμένης ανόπτησης

ορίζουμε τις μεταβλητές, τα διανύσματα και τους μετρητές που είναι απαραίτητοι για την υλοποίηση του κώδικα

Comb_list	**Λίστα με όλους τους πιθανούς συνδυασμούς ανταλλαγής κόμβων**
CurrentPath	**Πίνακας με τις διαδρομές που δημιουργήθηκαν με τον αλγόριθμο NN**
BestSolution	**Η καλύτερη λύση που παράχθηκε μετά τον αλγόριθμο του NN**
bestCost	**Το κόστος της αρχικής μας λύσης μετά τον αλγόριθμο του NN**
Tstart	**Η θερμοκρασία εκκίνησης του αλγορίθμου SA**
Tmin	**Η θερμοκρασία τερματισμού του αλγορίθμου SA**
α	**Μεταβλητή ρυθμού μείωσης θερμοκρασίας**
f	**Μετρητής για τις συνολικές επαναλήψεις που εκτελεί ο αλγόριθμος SA**
f1	**Μετρητής που εκφράζει το σύνολο των επαναλήψεων που ο αλγόριθμος SA αποδέχθηκε μία χειρότερη λύση**
f2	**Μετρητής που εκφράζει το σύνολο των επαναλήψεων που ο αλγόριθμος SA εντόπισε μία βέλτιστη λύση**
f3	**Μετρητής που εκφράζει το σύνολο των ανταλλαγών που έκανε ο αλγόριθμος 1-1 Exchange σαν τυχαία διαταραχή του αλγορίθμου SA**
mincost	**Η αρχική μας λύση**
cost	**Η λύση μετά την τυχαία διαταραχή 1-1 Exchange**

Πίνακας 4.2 Επεξήγηση των μεταβλητών της συνάρτησης της Προσομοιωμένης Ανόπτησης (SA)

Αφού ορίστηκαν οι μεταβλητές ξεκινάει ο επαναληπτικός βρόγχος της προσομοιωμένης ανόπτησης. Ο αλγόριθμος επαναλαμβάνεται όσο η θερμοκρασία εκκίνησης **“Tstart”** είναι μεγαλύτερη από τη θερμοκρασία τερματισμού **“Tmin”**. Σε πρώτη φάση υλοποιείται ο αλγόριθμος τοπικής αναζήτησης 1-1 Exchange ο οποίος αποτελεί την τυχαία διαταραχή του αλγορίθμου. Στην λίστα **“comb_list”** αποθηκεύονται όλοι οι πιθανοί συνδυασμοί κόμβων που μπορούν να γίνουν ανταλλαγές διαδρομών. Οι ανταλλαγές των κόμβων γίνονται αυστηρά από διαφορετικές διαδρομές. Σε περίπτωση που ικανοποιούνται οι περιορισμοί χωρητικότητας και χρόνου υπολογίζεται το νέο κόστος και αποθηκεύεται στη μεταβλητή **cost** ενώ οι νέες διαδρομές μετά τις αλλαγές αποθηκεύονται στον πίνακα **“temp_Path”**. Στη δεύτερη φάση του αλγορίθμου υπολογίζουμε τη διαφορά **“δ”** η οποία προκύπτει από τη σχέση **“delta = cost – mincost”**. Σε περίπτωση όπου **δ** $<$ **0**, δηλαδή η λύση μας μετά την υλοποίηση της τυχαίας διαταραχής είναι μικρότερο από την αρχική μας λύση (**“cost < micost”**), τότε **“mincost = cost”** και οι διαδρομές μας αποθηκεύονται στον πίνακα **“currentPath”** και αναλόγως προσαρμόζονται και τα διανύσματα χωρητικότητάς της κάθε διαδρομής**.** Αντίθετα αν προκύψει ότι **“delta** $\geq$ **0”** τότε η λύση μας είναι χειρότερη από την προηγούμενη. Κατά την εκτέλεση των αλγορίθμων τοπικής αναζήτησης μία τέτοια λύση θα είχε απορριφθεί αλλά ο αλγόριθμος της προσομοιωμένης ανόπτησης θέλοντας να απεγκλωβιστεί από ένα τοπικό ελάχιστο δέχεται αυτή τη λύση με μία πιθανότητα αποδοχής **“P”** η οποία ονομάστηκε **“accept_proba”** και υπολογίστηκε από τον τύπο **“math.exp (-delta / T) ”** όπου “T” είναι η τρέχουσα θερμοκρασία του αλγορίθμου και υπολογίζεται από τον τύπο **“**$\boldsymbol{T = T}$**start**$-\boldsymbol{f \times a}$**”**. Για να γίνει αποδεκτή η πιθανότητα **“accept_proba”** πρέπει να είναι μεγαλύτερη από μία τιμή **“X”** η οποία παράγεται μέσω μίας γεννήτριας τυχαίων αριθμών ομοιόμορφα κατανεμημένων στην ακτίνα μεταξύ του 0 και του 1 και προκύπτει από την εντολή **“random.uniform(0,1)”.** Συνεπώς στην περίπτωση όπου η πιθανότητα **“accept_proba”** είναι μικρότερη από την τιμή

“X” τότε **“mincost = cost”** , “currentPath = Temp_Path” και αναπροσαρμόζονται τα διανύσματα χωρητικότητας των διαδρομών. Ο αλγόριθμος τερματίζει όταν η θερμοκρασία **“T”** γίνει μικρότερη από τη θερμοκρασία τερματισμού **“Tmin”**.

Η συνάρτηση που χρησιμοποιήθηκε για την εφαρμογή του αλγορίθμου της προσομοιωμένης ανόπτησης περιέχει και τους αλγορίθμους τοπικών αναζητήσεων “1-0 Relocate” και “2-Opt” σε ένα διαφορετικό βρόγχο επαναλήψεων όπου ουσιαστικά ελέγχουμε αν το αποτέλεσμα μετά τις τοπικές αναζητήσεις είναι καλύτερο από αυτό πριν τους αλγορίθμους τοπικών αναζητήσεων. Αφού λοιπόν γίνει αυτός ο έλεγχος αποθηκεύουμε το καλύτερο αποτέλεσμα που προκύπτει κατά τη διαδικασία της προσομοιωμένης ανόπτησης και των τοπικών αναζητήσεων και η συνάρτηση ολοκληρώνεται επιστέφοντας την βέλτιστη λύση.

Κεφάλαιο 5 – Αποτελέσματα και Συμπεράσματα

5.1 Παρουσίαση Αποτελεσμάτων και Συμπεράσματα

Τα δεδομένα που χρησιμοποιήθηκαν για την εξαγωγή των αποτελεσμάτων του προβλήματος είναι τα : (Augerat et al.,1995) dataset A,B,P, Christofides and Eilon dataset E for CVRP και Daniele Vigo dataset D for DCVRP και αναφέρονται σε 17 διαφορετικά παραδείγματα μικρής κλίμακας όπου το κάθε ένα από αυτά έχει διαφορετικά δεδομένα και διαφορετικούς περιορισμούς. Για την εξαγωγή αποτελέσματος κάθε διαφορετικού παραδείγματος-σεναρίου πραγματοποιήθηκε ένα σύνολο αρκετών επαναλήψεων του προγράμματος αποσκοπώντας στη βέλτιστη λύση αφού τα αποτελέσματα από τον αλγόριθμο της προσομοιωμένης ανόπτησης είναι πιθανό να διαφέρουν ελάχιστα κάθε φορά καθώς στην διαταραχή που πραγματοποιείται η επιλογή των συνδυασμών των κόμβων που θα ανταλλαχθούν κάθε φορά είναι τυχαία ενώ σε

ορισμένες περιπτώσεις χρειάστηκε να τροποποιήσουμε τις τιμές κάποιων παραμέτρων όπως τη θερμοκρασία εκκίνησης και τη θερμοκρασία τερματισμού της προσομοιωμένης ανόπτησης. Στον παρακάτω πίνακα (Πίνακας 5.1) γίνεται αντιληπτό ότι η αρχική λύση (Initial Solution) που παράχθηκε από τον αλγόριθμο του Πλησιέστερου Γείτονα σε σχέση με την τελική λύση βρίσκεται αρκετά κοντά, αφού η μέση απόκλιση των αρχικών λύσεων με τις τελικές είναι 3.24%. Το γεγονός αυτό δηλώνει ότι ο αλγόριθμος του Πλησιέστερου Γείτονα αν και είναι ένας "κοντόφθαλμος" αλγόριθμος απληστίας εξήγαγε ιδιαίτερα ικανοποιητικά αποτελέσματα τα οποία έχουν κατά μέσο όρο απόκλιση +11.3% με τις μέχρι στιγμής παγκοσμίως καλύτερες λύσεις που υπάρχουν στη βιβλιογραφία. Αυτό οφείλεται στη φύση των δεδομένων που επεξεργαστήκαμε. Σε δεύτερη φάση κατά την υλοποίηση των αλγορίθμων τοπικών αναζητήσεων έγινε αρχικά η χρήση του αλγορίθμου "1-0 Relocate" ο οποίος διέγραφε τυχαία έναν πελάτη από μία διαδρομή και τον εισήγαγε σε μία άλλη αποσκοπώντας στην μείωση του συνολικού κόστους και την εξάλειψη περιττών διαδρομών μικρής κλίμακας. Ο συγκεκριμένος αλγόριθμος βοηθάει στη διαμόρφωση του τελικού συνόλου των διαδρομών εφόσον είναι αρκετά πιο ευέλικτος από τους υπόλοιπους αλγορίθμους τοπικής αναζήτησης που χρησιμοποιήθηκαν για αυτό χρησιμοποιήθηκε πρώτος. Έπειτα με τη χρήση του αλγορίθμου τοπικής αναζήτησης "2 Opt" έγινε πιο στοχευμένη βελτιστοποίηση του ολικού κόστους αφού ο συγκεκριμένος αλγόριθμος ανταλλάζει τυχαία τη θέση δύο κόμβων πάντα από την ίδια διαδρομή οι οποίες είναι ήδη διαμορφωμένες από τον αλγόριθμο "1-0 Relocate". Σε τελική φάση χρησιμοποιήθηκε ο μεθευρετικός αλγόριθμος της Προσομοιωμένης Ανόπτησης όπου για την λειτουργία του έγινε η χρήση του αλγορίθμου τοπικής αναζήτησης "1-1 Exchange" σαν διαταραχή του συστήματος.

Ο συγκεκριμένος αλγόριθμος επιλέχθηκε σαν διαταραχή του συστήματος της Προσομοιωμένης Ανόπτησης επειδή γίνεται ακόμα πιο στοχευμένη ανταλλαγή πελατών, αυτή

τη φορά όμως από όλες τις διαδρομές των οχημάτων. Αυτό δημιουργεί ευνοϊκές συνθήκες για την αποφυγή του αλγορίθμου να εγκλωβιστεί σε τοπικά ελάχιστα. Στον παρακάτω πίνακα (Πίνακας 5.1) παρουσιάζονται αναλυτικά τα αποτελέσματα του προγράμματος. Στην πρώτη στήλη αναγράφονται τα παραδείγματα πάνω στα οποία υλοποιήθηκε ο κώδικας. Ο τρόπος με τον οποίο αποτυπώνονται τα παραδείγματα είναι ένας κωδικός π.χ. B-n51-k7 όπου το B αντιστοιχεί στο data set, το n51 είναι ο αριθμός των κόμβων-πελατών που πρέπει να εξυπηρετηθούν και το k7 δηλώνει τον ελάχιστο αριθμό διαδρομών. Στην δεύτερη και την τρίτη στήλη δηλώνονται ο περιορισμός χωρητικότητας (Qmax) και ο χρονικός περιορισμός (Tmax) αντίστοιχα ενώ στην τέταρτη και την πέμπτη στήλη παρουσιάζονται ο χρόνος εξυπηρέτησης ενός πελάτη (Service Time) και ο χρόνος καθυστέρησης εκφόρτωσης ενός οχήματος (Delay Time). Στην έκτη και στην έβδομη στήλη αναγράφονται οι αρχικές (Initial Solution) και οι τελικές λύσεις (Final Solution) κάθε παραδείγματος ενώ στην τελευταία στήλη αποτυπώνεται η απόκλιση μεταξύ της αρχικής με την τελική λύση (Deviation IS-FS).

Examples	Qmax	Tmax	ST	DT	Initial Solution	Final Solution	Deviation IS-FS(%)
A-n32-k5	100	246.89	0	0	1141.71	1133.91	-0.69
A-n33-k5	100	176.68	0	0	1120.51	1099.60	-1.87
A-n33-k6	100	175.83	0	0	1468.39	1420.17	-3.29
A-n34-k5	100	201.73	0	0	1089.83	1076.66	-1.21
A-n36-k5	100	236.53	0	0	1165.68	1128.37	-3.21
B-n50-k8	100	232.11	0	0	1973.51	1914.59	-2.99
B-n51-k7	100	186.69	0	0	1622.87	1622.87	0.00
B-n52-k7	100	158.17	0	0	1077.41	1077.41	0.00
B-n56-k7	100	186.22	0	0	1172.96	1151.96	-1.80
B-n57-k7	100	197.65	0	0	1858.66	1799.35	-3.20
E-n22-k4	6000	110.47	0	0	944.83	911.97	-3.48
E-n30-k3	4500	215.12	0	0	1075.50	960.61	-10.69
E-n33-k4	8000	262.48	0	0	1194.55	1132.14	-5.23
E-n51-k5	160	135.77	0	0	1086.77	1057.42	-2.71
E-n76-k10	140	129.29	0	0	1701.20	1594.58	-6.27
P-n21-k2	160	132.10	0	0	591.63	552.92	-6.55
P-n22-k8	3000	109.71	0	0	1159.01	1136.66	-1.93
Average							**-3.24**

Πίνακας 5.1 Σύγκριση Αρχικών – Τελικών Αποτελεσμάτων

Η σύγκριση των τελικών λύσεων με τις ευρέως διαδεδομένες βέλτιστες λύσεις στην παγκόσμια βιβλιογραφία παρουσιάζονται στον παρακάτω πίνακα(Πίνακας 5.2) όπου παρατηρείται μέση απόκληση των τελικών λύσεων με τις παγκοσμίως καλύτερες +8.06%.. Στην πρώτη στήλη αποτυπώνονται οι κωδικοί των παραδειγμάτων ενώ στη δεύτερη στήλη βρίσκονται τα αποτελέσματα που παρήγαγε ο solver CPLEX ο οποίος για να εξάγει αποτελέσματα απαιτεί να του δοθεί ένα κάτω φράγμα με ένα συγκεκριμένο κόστος και τερματίζει τη λειτουργία του όταν συναντήσει το κάτω φράγμα ή όταν περάσει ένα προκαθορισμένο χρονικό διάστημα που στη συγκεκριμένη περίπτωση το διάστημα αυτό είναι 48 ώρες. Στην τρίτη στήλη καταγράφονται τα καλύτερα αποτελέσματα που υπάρχουν στην βιβλιογραφία (Best Known Solution) ενώ στην τέταρτη στήλη αναφέρεται το σύνολο των οχημάτων που χρησιμοποιήθηκαν για την εξαγωγή της καλύτερης λύσης. Για παράδειγμα αν για μία διαδρομή χρησιμοποιήθηκαν 3 ιδιόκτητα οχήματα και συνολικά χρειάστηκαν 6 τότε ο αριθμός των οχημάτων συμβολίζεται ως (3,6). Στην πέμπτη στήλη καταγράφεται η τελική λύση του προγράμματος μετά το πέρας του αλγορίθμου της προσομοιωμένης ανόπτησης ενώ στην έκτη και έβδομη στήλη παρουσιάζονται τα οχήματα που χρησιμοποιήθηκαν και η απόκλιση της τελικής μας λύσης σε σχέση με την καλύτερη υπαρκτή λύση αντίστοιχα. Τέλος στην όγδοη στήλη καταγράφεται ο μέσος χρόνος 10 επαναλήψεων που χρειάστηκε ο κώδικας για να υλοποιηθεί.

Examples	CPLEX	BKS	Vehicles BKS	FS	Vehicles FS	Deviation FS-BKS(%)	Time (sec)
A-n32-k5	1063.53	1063.53	(3,5)	1133.91	(3,4)	7.01	5.94
A-n33-k5	1011.15	994.91	(3,5)	1099.60	(3,5)	9.62	7.87
A-n33-k6	1171.35	1169.29	(3,6)	1420.17	(3,6)	17.7	7.96
A-n34-k5	1109.61	1102.17	(3,5)	1076.66	(3,4)	-2.32	4.51
A-n36-k5	1106.55	1096.35	(3,5)	1128.37	(3,4)	1.02	8.28
B-n50-k8	1756.27	1739.95	(4,8)	1914.59	(4,8)	9.13	20.57
B-n51-k7	1586.59	1494.54	(4,7)	1622.87	(4,8)	7.91	28.17
B-n52-k7	1236.50	1202.7	(4,7)	1077.41	(4,5)	-10.5	20.17
B-n56-k7	1172.21	1168.22	(4,7)	1151.96	(4,5)	-0.88	7.81
B-n57-k7	1801.78	1549.4	(4,7)	1799.35	(4,8)	13.9	37.00
E-n22-k4	700.94	700.93	(2,4)	911.97	(2,5)	23.2	3.55
E-n30-k3	733.24	730.83	(2,3)	960.61	(2,4)	24.0	5.51
E-n33-k4	1076.90	1057.05	(2,4)	1132.14	(2,4)	6.64	3.34
E-n51-k5	933.37	921.28	(3,5)	1057.42	(3,5)	12.88	21.03
E-n76-k10	1847.55	1750.26	(5,10)	1594.58	(5,8)	-8.91	55.52
P-n21-k2	383.46	383.46	(1,2)	552.92	(1,3)	30.7	3.42
P-n22-k8	1183.87	1183.7	(5,8)	1136.66	(5,7)	-3.98	2.10
Average						**+8.06**	

Πίνακας 5.2 Σύγκριση Τελικών – Παγκοσμίως Καλύτερων Αποτελεσμάτων

Για την εύρεση των παραπάνω αποτελεσμάτων χρησιμοποιήθηκε ο η γραμμική, η εκθετική και η λογαριθμική προσέγγιση μείωσης της θερμοκρασίας στον αλγόριθμο της προσομοιωμένης ανόπτησης και οι μαθηματικοί τύποι των ρυθμών ψύξης είναι οι εξής:

- **Γραμμική Προσέγγιση:** $\boldsymbol{T(f) = T_{start} - f \times a}$
- **Εκθετική Προσέγγιση:** $\boldsymbol{T(f) = T_{start} \times a^{f}}$
- **Λογαριθμική Προσέγγιση:** $\boldsymbol{T(f) = T_{start}/ln(f + 2)}$

Η γραμμική προσέγγιση είναι εκτελεστικά απλή και αποτελεσματική σε πολλές διαφορετικές περιπτώσεις, ειδικά για προβλήματα με γνωστό φάσμα βέλτιστων λύσεων όπως το πρόβλημα μας. Ωστόσο , δεν είναι η καλύτερη μέθοδος για όλα τα προβλήματα βελτιστοποίησης διότι είναι για μεγάλο πλήθος δεδομένων αρκετά χρονοβόρα ενώ η επιλογή των σωστών τιμών των παραμέτρων της αρχικής-τελικής θερμοκρασίας και της μεταβλητής ρυθμού ψύξης να χρειάζονται αρκετές δοκιμές.

Η εκθετική προσέγγιση εξαρτάται σε μεγάλο βαθμό από τη μεταβλητή ρυθμού ψύξης α $(0 < \alpha < 1)$ καθώς όσο πιο μικρή είναι η τιμή της τόσο πιο αργός είναι και ο ρυθμός ψύξης ενώ παρατηρείται μία πιο σταδιακή σύγκλιση προς τη βέλτιστη λύση. Η εκθετική προσέγγιση μπορεί να συγκλίνει αρκετά γρήγορα στο ολικό ελάχιστο, είναι ιδιαίτερα απλή και ευέλικτη και προτιμάται για την εύρεση βέλτιστης λύσης σε μεγάλης κλίμακας προβλήματα συνδυαστικής βελτιστοποίησης.

Τέλος, στη λογαριθμική προσέγγιση η θερμοκρασία μειώνεται πιο αργά καθώς αυξάνεται ο αριθμός των επαναλήψεων, γεγονός που επιτρέπει στον αλγόριθμο να εξερευνήσει τον χώρο αναζήτησης πιο διεξοδικά σε υψηλότερες θερμοκρασίες και να επικεντρωθεί στη βελτίωση της καλύτερης λύσης σε χαμηλότερες θερμοκρασίες. Αυτό συμβαίνει διότι όσο η θερμοκρασία πλησιάζει το μηδέν, ο αλγόριθμος επικεντρώνεται περισσότερο στη βελτίωση της καλύτερης

λύσης που έχει βρεθεί μέχρι στιγμής, παρά στην εξερεύνηση του χώρου αναζήτησης. Τέλος, παρατηρήθηκε ότι σε σχέση με τις υπόλοιπες προσεγγίσεις που επιλέξαμε ο μέσος χρόνος εκτέλεσης του προγράμματος ήταν ο μικρότερος όπου σε 17 παραδείγματα ο μέσος χρόνος ήταν 4.79 δευτερόλεπτα.

Προγραμματιστικά, και οι τρεις προσεγγίσεις του ρυθμού ψύξης επέφεραν αποτελέσματα με αμελητέες διαφορές, αλλά κυριάρχησαν τα αποτελέσματα της γραμμικής προσέγγισης, πέρα από τέσσερα παραδείγματα λόγω της μικρής κλίμακας των παραδειγμάτων αλλά και λόγω του γνωστού φάσματος των τελικών λύσεων. Στον παρακάτω πίνακα παρουσιάζονται τα αποτέλεσματα και ο χρόνος εκτέλεσης όλων των παραδειγμάτων σύμφωνα με την προσέγγιση του ρυθμού ψύξης που χρησιμοποιήθηκε.

Examples	FS(Linear	Time	FS(Exponential	Time	FS(Logarithmic	Time

	Approximation)	(sec)	Approximation)	(sec)	Approximation)	(sec)
A-n32-k5	1133.91	5.94	1133.91	3.05	1133.91	2.60
A-n33-k5	1099.6	7.87	1114.59	3.47	1112.23	3.11
A-n33-k6	1420.17	7.96	1436.74	3.33	1438.49	2.88
A-n34-k5	1085.19	8.14	1076.66	4.51	1085.17	3.33
A-n36-k5	1128.37	8.28	1128.37	3.92	1128.37	3.41
B-n50-k8	1914.59	20.57	1914.59	9.06	1914.59	6.94
B-n51-k7	1622.87	28.17	1622.87	11.33	1622.87	8.07
B-52-k7	1077.41	20.17	1077.41	7.66	1077.41	5.26
B-n56-k7	1166.22	19.25	1151.96	7.81	1168.24	6.02
B-n57-k7	1799.35	37.00	1799.35	15.60	1799.35	10.50
E-n22-k4	911.97	3.55	917.40	1.95	918.26	1.70
E-n30-k3	960.61	5.51	992.81	2.95	992.81	2.84
E-n33-k4	1141.82	8.74	1141.82	3.83	1132.14	3.34
E-n51-k5	1057.42	21.03	1074.82	9.02	1078.53	6.56
E-n76-k10	1594.58	55.52	1654.47	23.52	1604.26	15.77
P-n21-k2	552.92	3.42	552.92	2.09	552.92	1.65
P-n22-k8	1142.60	3.30	1136.66	2.10	1139.60	1.82

Πίνακας 5.3 Σύγκριση τελικών αποτελσμάτων των διαφορετικών προσεγγίσεων ρυθμού ψύξης της προσομοιωμένης ανόπτησης

Συμπερασματικά, τα τελικά αποτελέσματα που παρήγαγε ο αλγόριθμος είναι αρκετά ικανοποιητικά αφού η μέση απόκλισή τους από τις καλύτερες παγκόσμιες λύσεις είναι μόλις 8,06%. Πολύ σημαντικό ρόλο στη μείωση του συνολικού κόστους έπαιξε η δυνατότητα των οχημάτων να εκτελούν πολλαπλές διαδρομές στη περίπτωση που δεν είχε ακόμα παραβιαστεί ο χρονικός περιορισμός τους με αποτέλεσμα να μειωθούν έτσι τα συνολικά δρομολόγια και συνεπώς και το ολικό κόστος. Ωστόσο, σε περιπτώσεις που οι κόμβοι-πελάτες ήταν ελάχιστοι ο αλγόριθμος έτεινε να χρησιμοποιήσει παραπάνω ενοικιαζόμενα οχήματα με αποτέλεσμα να αυξάνεται το ολικό κόστος. Εν κατακλείδι, ο αλγόριθμος της προσομοιωμένης ανόπτησης με τη γραμμική προσέγγιση ψύξης λειτούργησε ιδιαίτερα αποτελεσματικά για τη βελτιστοποίηση του συγκεκριμένου προβλήματος δρομολόγησης οχημάτων.

5.2 Γραφική Απεικόνιση Αποτελεσμάτων

Στη συγκεκριμένη παράγραφο παρουσιάζονται γραφικά τα αποτελέσματα από ορισμένα παραδείγματα όπου η λύση του αλγορίθμου ήταν ιδιαίτερα ικανοποιητική. Σε κάθε παράδειγμα αναφέρεται ο κωδικός του ,η προσέγγιση ρυθμού ψύξης που επιλέχθηκε, ο χρονικός περιορισμός, ο περιορισμός χωρητικότητας και ο χρόνος εξυπηρέτησης ενώ κάτω από το γράφημα παρουσιάζεται και ο αντίστοιχος πίνακας διαδρομών των οχημάτων.

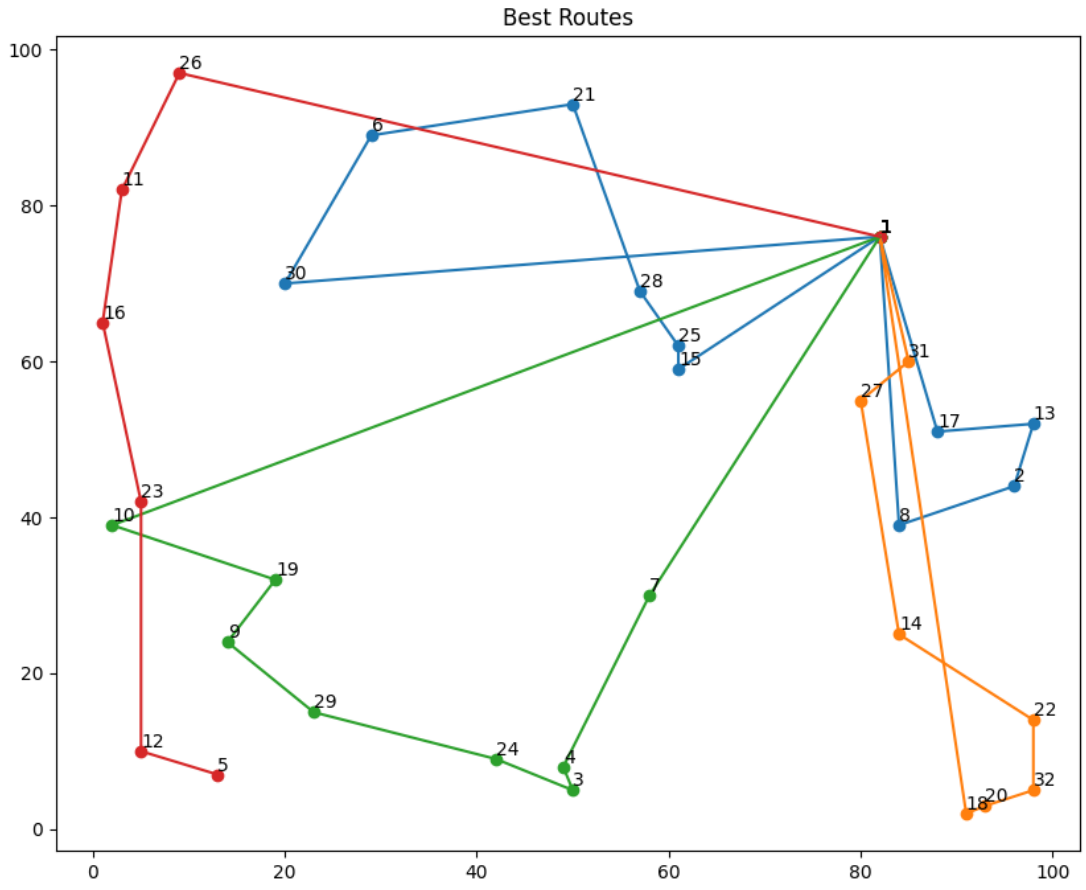
Παράδειγμα A-n32-k5

Γραμμική Προσέγγιση

Tmax = 246.89

Qmax = 100

Service Time = 0



Διαδρομές:

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

COST = 1133.91

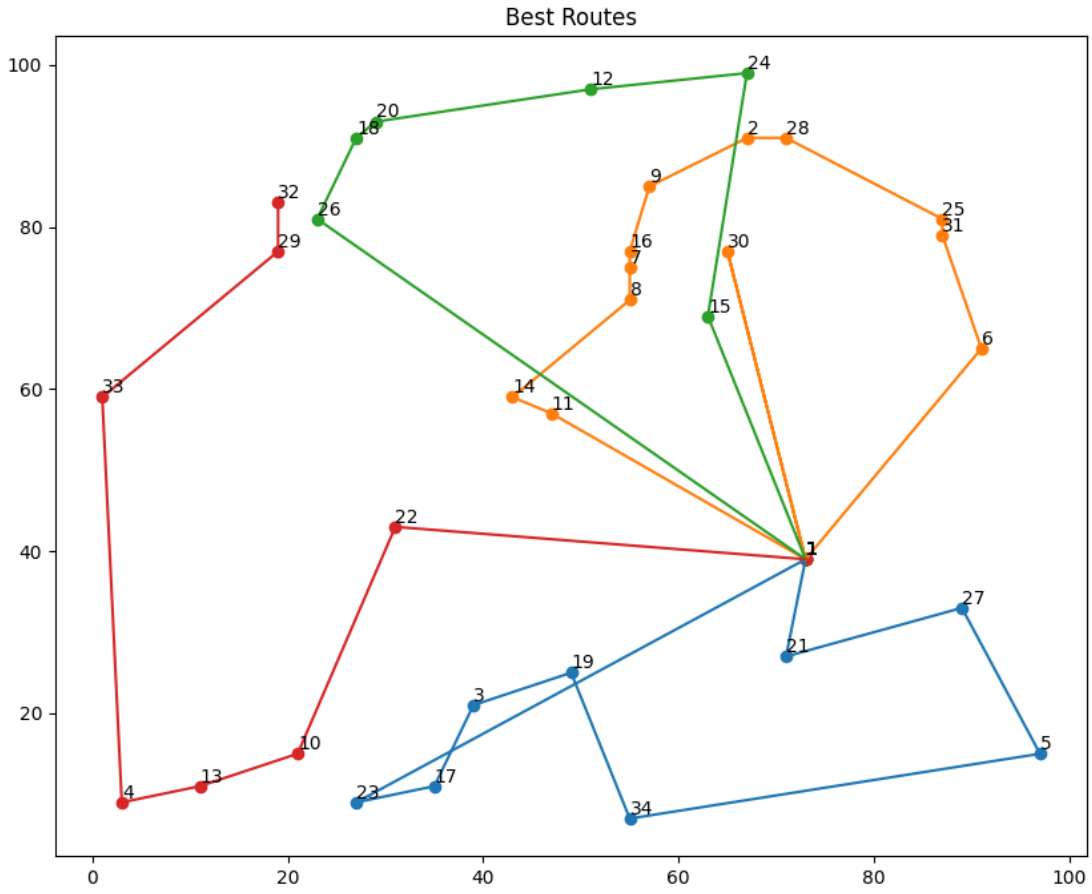
Παράδειγμα A-n34-k5

Εκθετική Προσέγγιση

Tmax = 201.73

Qmax = 100

Service Time = 0



Διαδρομές:

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

COST = 1076.66

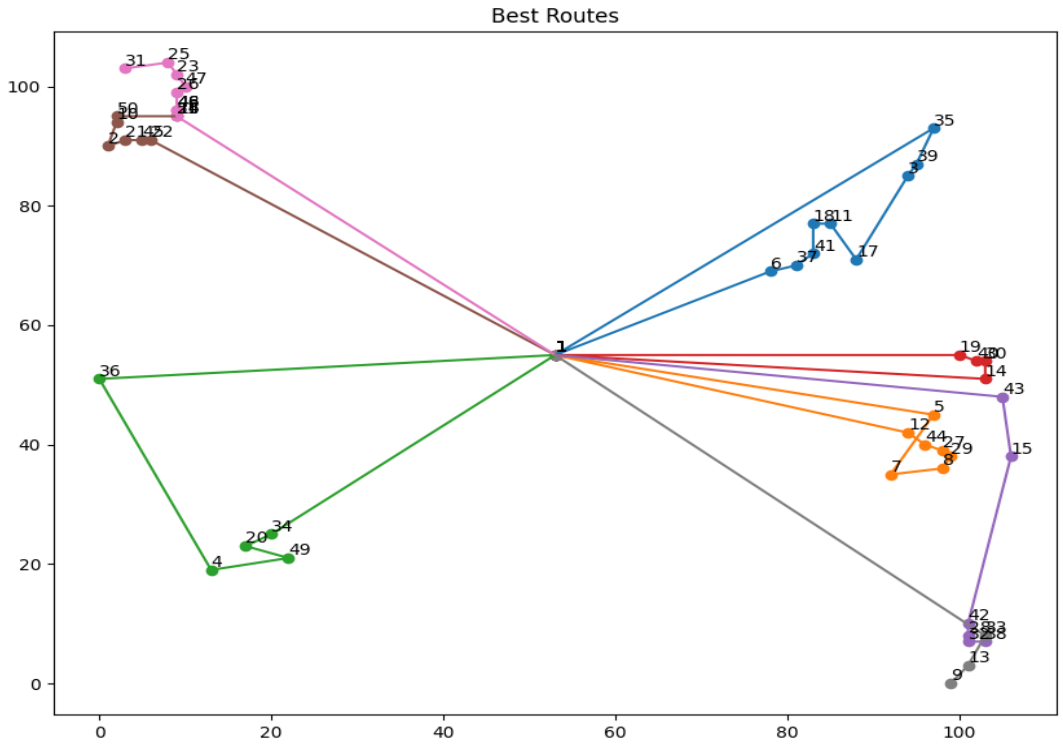
Παράδειγμα B-n51-k7

Γραμμική Προσέγγιση

Tmax = 186.69

Qmax = 100

Service time = 0



Διαδρομές:

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

COST = 1622.87

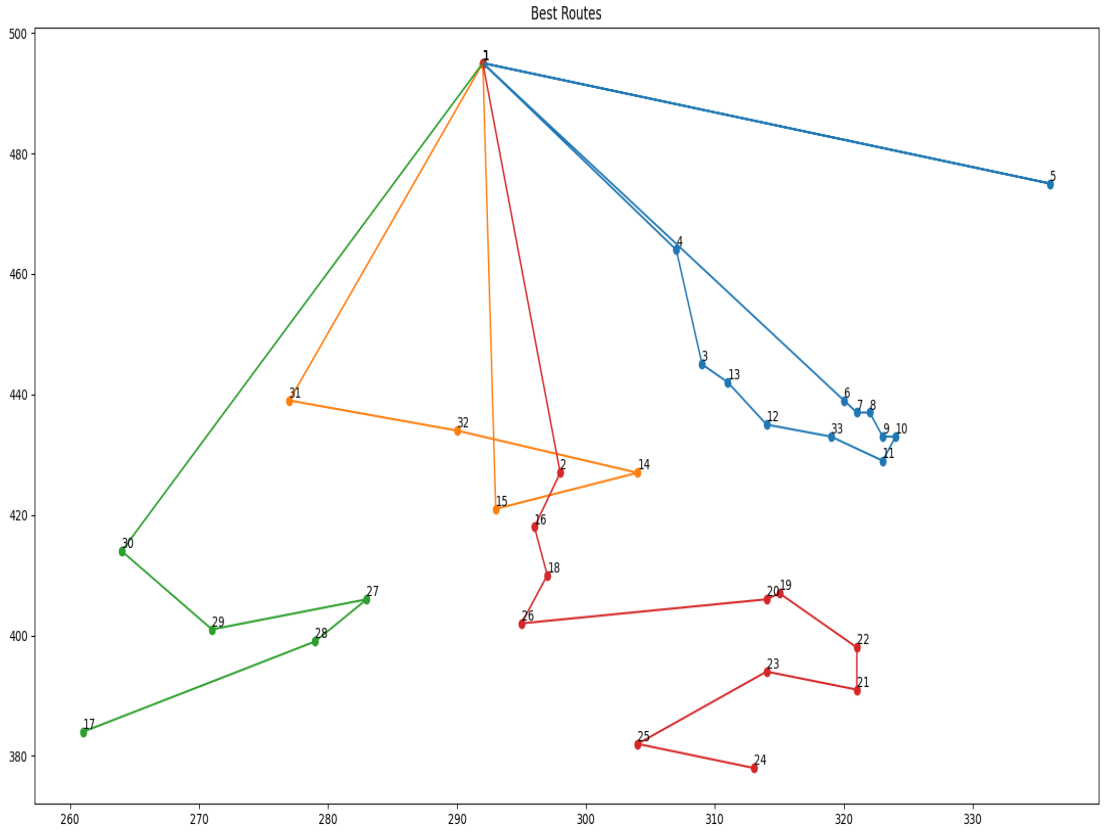
Παράδειγμα E-n33-k4

Λογαριθμική Προσέγγιση

Tmax = 262.48

Qmax = 8000

Service time = 0



Διαδρομές:

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

COST = 1132.14

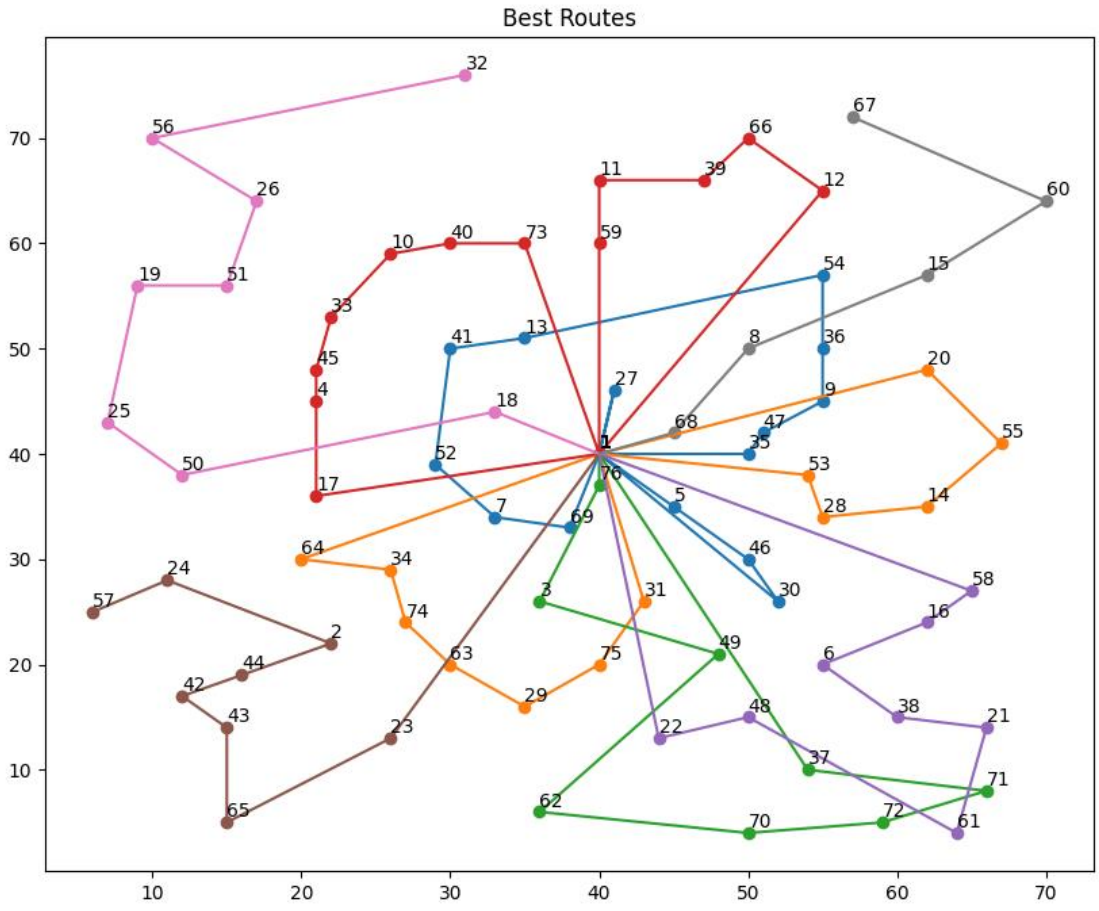
Παράδειγμα E-n76-k10

Γραμμική Προσέγγιση

Tmax = 129.29

Qmax = 140

Service time = 0



Διαδρομές:

1	**69**	**7**	**52**	**41**	**13**	**54**	**36**	**9**	**47**	**35**	**1**	**27**	**1**	**5**	**46**	**30**	**1**
1	53	28	14	55	20	1	31	75	29	63	74	34	64	1	0	0	0
1	76	3	49	62	70	72	71	37	1	0	0	0	0	0	0	0	0
1	17	4	45	33	10	40	73	1	59	11	39	66	12	1	0	0	0
1	58	16	6	38	21	61	48	22	1	0	0	0	0	0	0	0	0
1	23	65	43	42	44	2	24	57	0	0	0	0	0	0	0	0	0
1	18	50	25	19	51	26	56	32	0	0	0	0	0	0	0	0	0
1	68	8	15	60	67	0	0	0	0	0	0	0	0	0	0	0	0

COST = 1594.58

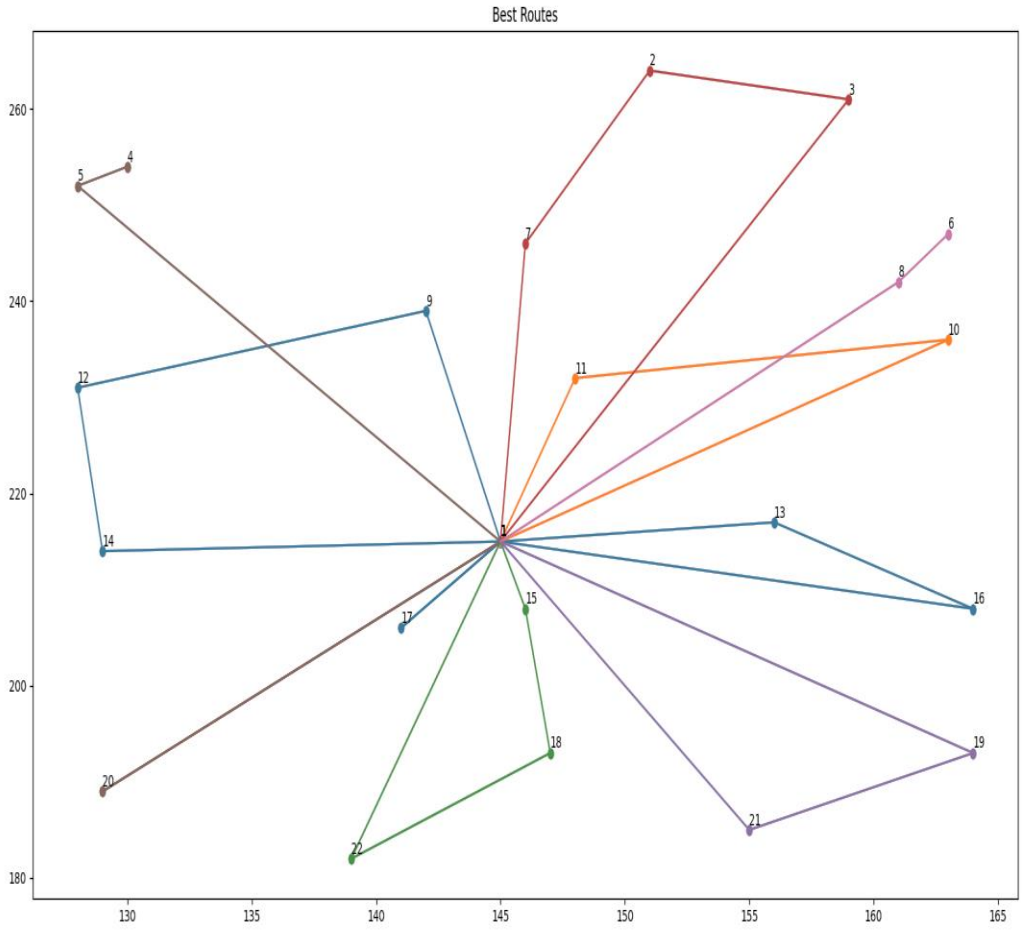
Παράδειγμα P-n22-k8

Εκθετική Προσέγγιση

Tmax = 109.71

Qmax = 3000

Service time = 0



Διαδρομές:

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

COST = 1136.66

Βιβλιογραφία

[1] Christopher, M. (2011). Logistics & Supply Chain Management (Financial Times) (4th ed.). Ft Pr.

[2] Liu, R., & Jiang, Z. (2012). The close–open mixed vehicle routing problem. European Journal of Operational Research, 220(2), 349–360. https://doi.org/10.1016/j.ejor.2012.01.061

[3] Βιδάλης, Μ. (2017). Εφοδιαστική (Logistics): Μια ποσοτική προσέγγιση (2nd ed.). ΚΛΕΙΔΑΡΙΘΜΟΣ.

[4] Μαρινάκης, Ι., Μαρινάκη, Μ., & Μυγδαλάς, Α. (2019). Προβλήματα Δρομολόγησης Οχημάτων στη Διαχείριση της Εφοδιαστικής Αλυσίδας: Μοντελοποίηση & Αλγόριθμοι Επίλυσης (1st ed.). NewTech Pub.

[5] Kraslawski, A., & Turunen, I. (2013). 23rd European Symposium on Computer Aided Process Engineering (Volume 32) (Computer Aided Chemical Engineering, Volume 32) (1st ed.). Elsevier.

[6] Erdinc, O. (2017). Optimization in Renewable Energy Systems: Recent Perspectives. Elsevier Gezondheidszorg.

[7] Arora, J. S. (2014). Introduction to Optimum Design. Academic Press.

[8] Emmanouil E. Zachariadis, Chris T. Kiranoudis (2010). A Strategy for Reducing the Computational Complexity of Local Search-Based Methods, and its Application to the Vehicle Routing Problem, Department of Process Analysis and Plant Design, National Technical University of Athens.

[9] https://en.wikipedia.org/wiki/2-opt

[10] Tzanetos, A., Vassiliadis, V. and Dounias, G. (2018). Boosting the performance of hybrid Nature-Inspired algorithms: Application from the financial optimization domain. Logic Journal of the IGPL. [online] https://doi.org/10.1093/jigpal/jzy048.

[11] Σιδεριάδης Κ. (2016) , Μελέτη περίπτωσης ενός προβλήματος δρομολόγησης οχημάτων με περιορισμένη χωρητικότητα (CVRP), Μεταπτυχιακή Διατριβή, Τμήμα Πληροφορικής, ΠΑ.ΠΕΙ.

[12] Bertsimas, D., & Tsitsiklis, J. (1993). Simulated annealing. Statistical Science, Vol. 8(10–15).

[13] Christopher M. (2005). “Logistics and supply chain management”.PrenticeHall Inc

[14] Laudon, K. and Laudon, J. (2014). Πληροφοριακά Συστήματα Διοίκησης, Εκδόσεις Κλειδάριθμός, Αθήνα.