



ΠΑΝΕΠΙΣΤΗΜΙΟ ΘΕΣΣΑΛΙΑΣ
ΠΟΛΥΤΕΧΝΙΚΗ ΣΧΟΛΗ
ΤΜΗΜΑ ΜΗΧΑΝΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ

ΔΡΟΜΟΛΟΓΗΣΗ ΣΤΟΛΟΥ ΟΧΗΜΑΤΩΝ : ΜΕΛΕΤΗ ΠΕΡΙΠΤΩΣΗΣ

ΥΠΟ

ΠΕΡΟΥΛΑΚΗ ΛΥΔΙΑ

Διπλωματούχου Μηχανικού Παραγωγής και Διοίκησης Πολυτεχνείου Κρήτης 2022

Μεταπτυχιακή Εργασία

Υπεβλήθη για την εκπλήρωση μέρους των
απαιτήσεων για την απόκτηση του
Μεταπτυχιακού Διπλώματος Ειδίκευσης

Βόλος, 2025

© 2025 Περουλάκη Λυδία

Η έγκριση της μεταπτυχιακής εργασίας από το Τμήμα Μηχανολόγων Μηχανικών της Πολυτεχνικής Σχολής του Πανεπιστημίου Θεσσαλίας δεν υποδηλώνει αποδοχή των απόψεων του συγγραφέα (Ν. 5343/32 αρ. 202 παρ. 2).

Εγκρίθηκε από τα Μέλη της Τριμελούς Εξεταστικής Επιτροπής:

Πρώτος Εξεταστής (Επιβλέπων)	Δρ. Σαχαρίδης Γεώργιος Αναπληρωτής Καθηγητής, Τμήμα Μηχανολόγων Μηχανικών, Πανεπιστήμιο Θεσσαλίας
Δεύτερος Εξεταστής	Δρ. Κοζανίδης Γεώργιος Αναπληρωτής Καθηγητής, Τμήμα Μηχανολόγων Μηχανικών, Πανεπιστήμιο Θεσσαλίας
Τρίτος Εξεταστής	Δρ. Λυμπερόπουλος Γεώργιος Καθηγητής, Τμήμα Μηχανολόγων Μηχανικών, Πανεπιστήμιο Θεσσαλίας.

ΕΥΧΑΡΙΣΤΙΕΣ

Φτάνοντας στο τέλος αυτού του κύκλου σπουδών, αισθάνομαι βαθιά την ανάγκη να εκφράσω την ευγνωμοσύνη μου σε όλους όσους συνέβαλαν στην ολοκλήρωση αυτής της εργασίας.

Πρώτα απ' όλα, θα ήθελα να ευχαριστήσω θερμά τον κύριο Σαχαρίδη, τον επιβλέποντα καθηγητή μου, για την καθοδήγηση, τις χρήσιμες παρατηρήσεις και πληροφορίες και την πολύτιμη βοήθειά του καθ' όλη τη διάρκεια της εκπόνησης της παρούσας εργασίας.

Επίσης, ένα μεγάλο ευχαριστώ στην οικογένειά μου και στους φίλους μου για την αδιάκοπη στήριξη και την ενθάρρυνσή τους, που αποτέλεσαν πηγή δύναμης σε κάθε βήμα αυτής της προσπάθειας.

ΔΡΟΜΟΛΟΓΗΣΗ ΣΤΟΛΟΥ ΟΧΗΜΑΤΩΝ : ΜΕΛΕΤΗ ΠΕΡΙΠΤΩΣΗΣ

ΠΕΡΟΥΛΑΚΗ ΛΥΔΙΑ

Τμήμα Μηχανολόγων Μηχανικών, Πανεπιστήμιο Θεσσαλίας, 2025

Επιβλέπων Καθηγητής: Δρ. Γεώργιος Σαχαρίδης,
Αναπληρωτής Καθηγητής Επιχειρησιακής Έρευνας

Περίληψη

Στο σημερινά εξαιρετικά ανταγωνιστικό περιβάλλον, η ανάπτυξη ενός αποτελεσματικού συστήματος logistics αποτελεί βασικό στοιχείο για την επιτυχία των επιχειρήσεων. Ένα από τα κύρια ζητήματα που καλούνται οι επιχειρήσεις να επιλύσουν, είναι η αναζήτηση βέλτιστων διαδρομών που θα εξυπηρετούν τους πελάτες με τον πιο αποδοτικό και οικονομικό τρόπο. Το Πρόβλημα Δρομολόγησης Οχημάτων (VRP) αποτελεί ένα πρόβλημα συνδυαστικής βελτιστοποίησης, όπου πραγματοποιείται αναζήτηση του βέλτιστου συνόλου διαδρομών για έναν στόλο οχημάτων, εξασφαλίζοντας την εξυπηρέτηση πελατών και την ελαχιστοποίηση των δαπανών υπό καθορισμένους περιορισμούς.

Στόχος της παρούσας εργασίας είναι η εύρεση των βέλτιστων διαδρομών μιας εμπορικής εταιρίας διανομών που καλείται να καλύψει ο στόλος οχημάτων της, με σκοπό την κάλυψη της ζήτησης του συνόλου των πελατών ελαχιστοποιώντας την συνολική απόσταση που διανύουν τα οχήματα χωρίς να παραβιάζονται οι περιορισμοί που έχουν τεθεί στο υπό μελέτη πρόβλημα.

Για την προσέγγιση της εύρεσης των βέλτιστων διαδρομών, εφαρμόστηκαν δυο μέθοδοι υλοποίησης. Πρώτον, εξετάστηκαν οι λύσεις που προέκυψαν με τη χρήση βιβλιοθηκών του OR-TOOLS της Google υλοποιώντας τη μέθοδο Path Cheapest Arc η οποία αποτελεί μια τεχνική για την εύρεση αρχικής λύσης. Αυτή η μέθοδος βρίσκει την πιο φθηνή καμπύλη για να συνδέσει τους πελάτες με βάση το μικρότερο κόστος. Στη συνέχεια, για να βελτιωθεί η συνολική απόσταση των διαδρομών, εφαρμόστηκε ο μεθευρετικός αλγόριθμος Guided Local Search, ο οποίος επιδιώκει τη βελτίωση της αρχικής λύσης. Κατά την δεύτερη προσέγγιση,

εκτελέστηκε ο αλγόριθμος Πλησιέστερης Εισαγωγής ο οποίος λειτουργεί με την εισαγωγή του πλησιέστερου πελάτη σε κάθε βήμα, επεκτείνοντας τη διαδρομή έως ότου καλυφθούν όλοι οι πελάτες.

Τέλος πραγματοποιήθηκε μια σύγκριση των δυο προσεγγίσεων για την εύρεση της πιο αποτελεσματικής μεθόδου. Τα αποτελέσματα έδειξαν ότι οι λύσεις που προέκυψαν με τη χρήση του OR-TOOLS ήταν γενικά πιο αποτελεσματικές, με μικρότερες συνολικές αποστάσεις, ειδικά σε πιο σύνθετες διαδρομές. Ο αλγόριθμος Nearest Insertion, αν και πρόκειται για γρήγορο και απλό αλγόριθμο στην εκτέλεσή του, συχνά παγιδεύεται σε τοπικά ελάχιστα, με αποτέλεσμα να μην επιτυγχάνει τις ίδιες βέλτιστες λύσεις. Ωστόσο, η απόδοσή του σε απλούστερα σενάρια ήταν ικανοποιητική, με διαφορές της τάξης του 10% συγκριτικά με το OR-TOOLS.

Συμπερασματικά, η χρήση των βιβλιοθηκών OR-TOOLS είναι πιο κατάλληλη για σύνθετες περιπτώσεις, ενώ ο Nearest Insertion προσφέρει μια ταχεία λύση για λιγότερο απαιτητικές συνθήκες.

FLEET VEHICLE ROUTING – CASE STUDY

PEROULAKI LYDIA

Department of Mechanical Engineering, University of Thessaly, 2025

Supervisor: Dr George Sacharidis
Associate Professor of Operational Research

Abstract

In today's highly competitive environment, the development of an effective logistics system is a key element for business success. One of the main issues that companies have to solve is the search for optimal routes to serve customers in the most efficient and cost-effective way. The Vehicle Routing Problem (VRP) is a combinatorial optimization problem, where a search for the optimal set of routes for a fleet of vehicles is performed, ensuring customer service and minimizing costs under specified constraints.

The aim of this study is to find the optimal routes for a distribution company that its vehicle fleet is required to cover, in order to meet the demand of all customers by minimizing the total distance travelled by the vehicles without violating the constraints imposed in the problem.

Two implementation methods were applied to the approach of finding the optimal routes. First, the solutions obtained using Google's OR-TOOLS libraries were examined by implementing the Path Cheapest Arc method which is a technique for finding an initial solution. This method finds the cheapest arc to connect customers based on the lowest cost. Then, to improve the overall path distance, the Guided Local Search algorithm was applied which seeks to improve the initial solution. In the second approach, the Nearest Insertion algorithm was performed which works by importing the nearest customer at each step, extending the route until all customers are covered.

Finally, a comparison of the two methods was performed to find the most efficient method. The results showed that the solutions obtained using OR-TOOLS were generally more efficient,

with shorter overall distances, especially on more complex routes. The Nearest Insertion algorithm, although is a fast and simple algorithm to execute, often gets trapped in local minima, resulting in not achieving the same optimal solutions. However, its performance in simpler scenarios was satisfactory, with differences of 10% compared to OR-TOOLS.

In conclusion, the use of OR-TOOLS libraries is more suitable for complex cases, while Nearest Insertion offers a fast solution for less demanding conditions.

Πίνακας περιεχομένων

Περίληψη	4
Abstract.....	6
1 ΚΕΦΑΛΑΙΟ: ΕΙΣΑΓΩΓΗ	15
1.1 ΔΙΑΧΕΙΡΙΣΗ ΕΦΟΔΙΑΣΤΙΚΗΣ ΑΛΥΣΙΔΑΣ ΚΑΙ LOGISTICS	16
1.1.1. ΕΦΟΔΙΑΣΤΙΚΗ ΑΛΥΣΙΔΑ.....	17
1.1.2. LOGISTICS	18
1.1.3. ΔΙΑΦΟΡΕΣ SUPPLY CHAIN MANAGEMENT & LOGISTICS.....	21
1.2 ΛΕΙΤΟΥΡΓΙΕΣ ΕΦΟΔΙΑΣΤΙΚΗΣ ΑΛΥΣΙΔΑΣ.....	22
1.3 ΔΙΚΤΥΑ ΔΙΑΝΟΜΗΣ.....	25
1.3.1. THIRD PARTY LOGISTICS.....	27
2 ΚΕΦΑΛΑΙΟ: ΔΡΟΜΟΛΟΓΗΣΗ ΟΧΗΜΑΤΩΝ	28
2.1 ΤΟ ΠΡΟΒΛΗΜΑ ΤΟΥ ΠΛΑΝΟΔΙΟΥ ΠΩΛΗΤΗ - TSP	29
2.1.1 ΜΟΝΤΕΛΟΠΟΙΗΣΗ TSP	30
2.2 ΤΟ ΠΡΟΒΛΗΜΑ ΔΡΟΜΟΛΟΓΗΣΗΣ ΟΧΗΜΑΤΩΝ - VRP	31
2.2.1 ΟΡΙΣΜΟΣ ΤΟΥ ΠΡΟΒΛΗΜΑΤΟΣ	32
2.2.2 ΤΟ ΠΕΡΙΟΡΙΣΜΕΝΗΣ ΧΩΡΗΤΙΚΟΤΗΤΑΣ ΠΡΟΒΛΗΜΑ ΔΡΟΜΟΛΟΓΗΣΗΣ ΟΧΗΜΑΤΩΝ (CVRP) & ΜΟΝΤΕΛΟΠΟΙΗΣΗ	35
2.2.3 ΤΟ ΠΡΟΒΛΗΜΑ ΔΡΟΜΟΛΟΓΗΣΗΣ ΣΕ ΔΕΔΟΜΕΝΑ ΧΡΟΝΙΚΑ ΠΕΡΙΘΩΡΙΑ (VRPTW)	37
2.2.4 ΤΟ ΠΡΟΒΛΗΜΑ ΠΑΡΑΛΑΒΗΣ ΚΑΙ ΠΑΡΑΔΟΣΗΣ (VRPPD).....	38
2.2.5 ΥΠΑΡΞΗ ΠΟΛΛΑΠΛΩΝ ΑΠΟΘΗΚΩΝ (MD-VRP)	39
2.2.6 ΤΟ ΠΡΟΒΛΗΜΑ ΔΡΟΜΟΛΟΓΗΣΗΣ ΟΧΗΜΑΤΩΝ ΜΕ BACKHAULS ΚΑΙ LINEHAULS.....	40
3 ΚΕΦΑΛΑΙΟ: ΑΛΓΟΡΙΘΜΟΙ ΕΠΙΛΥΣΗΣ ΠΡΟΒΛΗΜΑΤΩΝ ΣΥΝΔΙΑΣΤΙΚΗΣ ΒΕΛΤΙΣΤΟΠΟΙΗΣΗΣ.....	42
3.1 ΑΚΡΙΒΕΙΣ ΑΛΓΟΡΙΘΜΟΙ	43
3.2 ΕΥΡΕΤΙΚΟΙ ΑΛΓΟΡΙΘΜΟΙ	43
3.3 ΜΕΘΕΥΡΕΤΙΚΟΙ ΑΛΓΟΡΙΘΜΟΙ	44
4 ΚΕΦΑΛΑΙΟ: ΜΕΛΕΤΗ ΠΕΡΙΠΤΩΣΗΣ.....	49
4.1 ΠΛΗΡΟΦΟΡΙΕΣ ΕΤΑΙΡΙΑΣ 3PL	49

4.2	ΔΙΑΧΕΙΡΙΣΗ ΔΕΔΟΜΕΝΩΝ	51
4.3	ΜΟΝΤΕΛΟΠΟΙΗΣΗ ΠΡΟΒΛΗΜΑΤΟΣ	54
4.4	ΕΠΙΛΥΣΗ ΠΡΟΒΛΗΜΑΤΟΣ.....	57
5	ΚΕΦΑΛΑΙΟ: ΕΠΙΛΥΣΗ VRP ΜΕ ΧΡΗΣΗ GOOGLE OR-TOOLS	59
5.1	ΕΠΙΛΟΓΕΣ ΓΙΑ ΤΗΝ ΕΠΙΛΥΣΗ ΔΡΟΜΟΛΟΓΗΣΗΣ	60
5.1.1	ΡΑΘΗ CHEAPEST ARC.....	62
5.1.2	GUIDED LOCAL SEARCH.....	63
5.2	ΥΠΟΛΟΓΙΣΜΟΣ ΑΠΟΣΤΑΣΕΩΝ.....	65
5.3	ΚΩΔΙΚΑΣ	69
5.3.1	ΔΗΜΙΟΥΡΓΙΑ ΔΕΔΟΜΕΝΩΝ	69
5.3.2	ΔΗΜΙΟΥΡΓΙΑ ΜΟΝΤΕΛΟΥ	71
5.3.3	ΔΗΜΙΟΥΡΓΙΑ ΣΥΝΑΡΤΗΣΗΣ ΕΠΙΣΤΡΟΦΗΣ	73
5.3.4	ΔΗΜΙΟΥΡΓΙΑ ΚΟΣΤΟΥΣ ΤΑΞΙΔΙΟΥ	73
5.3.5	ΟΡΙΣΜΟΣ ΔΙΑΣΤΑΣΗΣ ΑΠΟΣΤΑΣΗΣ	74
5.3.6	ΔΗΜΙΟΥΡΓΙΑ ΣΥΝΑΡΤΗΣΗΣ ΕΠΙΣΤΡΟΦΗΣ ΖΗΤΗΣΗΣ ΚΑΙ ΠΕΡΙΟΡΙΣΜΟΥ ΧΩΡΗΤΙΚΟΤΗΤΑΣ	75
5.3.7	ΟΡΙΣΜΟΣ ΠΑΡΑΜΕΤΡΩΝ ΑΝΑΖΗΤΗΣΗΣ.....	77
5.3.8	ΟΡΙΣΜΟΣ ΣΥΝΑΡΤΗΣΗΣ ΕΚΤΥΠΩΣΗΣ.....	78
6	ΚΕΦΑΛΑΙΟ: ΕΠΙΛΥΣΗ VRP ΜΕ ΧΡΗΣΗ ΤΟΥ ΑΛΓΟΡΙΘΜΟΥ ΠΛΗΣΙΕΣΤΕΡΗΣ ΕΙΣΑΓΩΓΗΣ	79
6.1	ΑΛΓΟΡΙΘΜΟΣ ΠΛΗΣΙΕΣΤΕΡΟΥ ΓΕΙΤΟΝΑ.....	79
6.2	ΑΛΓΟΡΙΘΜΟΣ ΠΛΗΣΙΕΣΤΕΡΗΣ ΕΙΣΑΓΩΓΗΣ	80
6.3	ΚΩΔΙΚΑΣ	83
6.3.1	ΔΗΜΙΟΥΡΓΙΑ ΔΕΔΟΜΕΝΩΝ	83
6.3.2	ΔΗΜΙΟΥΡΓΙΑ ΣΥΝΑΡΤΗΣΗΣ NEAREST INSERTION.....	85
7	ΚΕΦΑΛΑΙΟ: ΑΠΟΤΕΛΕΣΜΑΤΑ & ΣΥΜΠΕΡΑΣΜΑΤΑ.....	89
7.1	ΑΠΟΤΕΛΕΣΜΑΤΑ ΕΡΕΥΝΑΣ.....	90
7.2	ΠΡΟΤΑΣΕΙΣ ΓΙΑ ΜΕΛΛΟΝΤΙΚΗ ΕΡΕΥΝΑ	104
8	ΒΙΒΛΙΟΓΡΑΦΙΑ	109

ΚΑΤΑΛΟΓΟΣ ΠΙΝΑΚΩΝ

Πίνακας 1.1: Διαφορές και ομοιότητες μεταξύ Logistics και Διαχείρισης Εφοδιαστικής Αλυσίδας.....	21
Πίνακας 7.1: Σύγκριση αποτελεσμάτων μεταξύ αλγορίθμων που εκτελέστηκαν με Or-TOOLS και Nearest Insertion.....	90
Πίνακας 7.2: Αποτελέσματα δρομολόγησης οχήματος 1 από τους κώδικες.	92
Πίνακας 7.3: Αποτελέσματα δρομολόγησης οχήματος 2 από τους κώδικες.	95
Πίνακας 7.4: Αποτελέσματα δρομολόγησης οχήματος 3 από τους κώδικες.	98
Πίνακας 7.5: Αποτελέσματα δρομολόγησης οχήματος 4 από τους κώδικες.	101

ΚΑΤΑΛΟΓΟΣ ΣΧΗΜΑΤΩΝ

Εικόνα 1.1: Διαδικασία της εφοδιαστικής αλυσίδας, από την απόκτηση των πρώτων υλών μέχρι την παράδοση του τελικού προϊόντος στον καταναλωτή.	18
Εικόνα 2.1: Διαγραμματική συσχέτιση μεταξύ κοινών προβλημάτων VRP [6].	29
Εικόνα 3.1: Αλγόριθμοι επίλυσης για παραλλαγές στο Vehicle Routing Problem [13].	42
Εικόνα 4.1: Αλλαγή ειδικών προϊόντων που ανήκουν στη λίστα ProductName σε συγκεκριμένες ποσότητες.	52
Εικόνα 4.2: Υπολογισμός συνολικής ζήτησης ανά πελάτη, ανά υποκατάστημα ανά ημέρα.	53
Εικόνα 4.3: Δημιουργία νέου αρχείου βάση πελατών που υπάρχουν διαθέσιμες οι συντεταγμένες τους.	53
Εικόνα 4.4: Εισαγωγή νέας στήλης, όπου ο πελάτης χαρακτηρίζεται από μοναδικό CustomerID.	54
Εικόνα 5.1: Πρότυπο ψευδοκώδικα για τον αλγόριθμο Guided Local Search [24].	64
Εικόνα 5.2: Πίνακας αποστάσεων υπολογισμένος βάση πραγματικών διαδρομών με χρήση API key.	67
Εικόνα 5.3: Πίνακας αποστάσεων υπολογισμένος με χρήση γεωγραφικών συντεταγμένων.	67
Εικόνα 5.4: Πίνακας που περιλαμβάνει τη διαφορά μεταξύ των αποστάσεων που υπολογίστηκαν βάση των πραγματικών διαδρομών και με χρήση γεωγραφικών συντεταγμένων.	67
Εικόνα 5.5: Δημιουργία δεδομένων μέσω της συνάρτησης create_data_model.	69
Εικόνα 5.6: Υπολογισμός του πίνακα αποστάσεων, μέσω της συνάρτησης calculate_distance_matrix.	70
Εικόνα 5.7: Υπολογισμός πίνακα απόστασης και δημιουργία λεξικού δεδομένων.	71
Εικόνα 5.8: Κύριο τμήμα κώδικα, εισαγωγή ημ/νίας από τον χρήστη και δημιουργία μοντέλου δρομολόγησης.	72
Εικόνα 5.9: Δημιουργία συνάρτησης επιστροφής.	73
Εικόνα 5.10: Συνάρτηση αξιολόγησης κόστους τόξου.	74
Εικόνα 5.11: Συνάρτηση επανάκλησης ζήτησης.	75
Εικόνα 5.12: Περιορισμός ελαχίστου φορτίου.	76

Εικόνα 5.13: Περιορισμός ελαχίστου αριθμού πελατών προς εξυπηρέτηση.	77
Εικόνα 5.14: Παράμετροι αναζήτησης αλγορίθμου επίλυσης.	77
Εικόνα 5.15: Συνάρτηση εκτύπωσης αποτελεσμάτων αλγορίθμου.	78
Εικόνα 6.1: Ψευδοκώδικας για την περιγραφή του αλγορίθμου πλησιέστερης εισαγωγής. .	82
Εικόνα 6.2: Εισαγωγή δεδομένων προβλήματος, μέσω της συνάρτησης create_data_model.	84
Εικόνα 6.3: Δημιουργία λεξικού δεδομένων.	85
Εικόνα 6.4: Δημιουργία συνάρτησης εύρεσης καλύτερης θέσης εισαγωγής πελατών εντός της υπο-διαδρομής.	86
Εικόνα 6.5: Δημιουργία διαδικασίας επαναλήψεων εκτέλεσης αλγορίθμου εύρεσης πλησιέστερης εισαγωγής πελατών εντός της υπο-διαδρομής.	87
Εικόνα 7.1: Διάγραμμα απεικόνισης δρομολόγησης οχήματος 1 με χρήση OR-TOOLS.	93
Εικόνα 7.2: Διάγραμμα απεικόνισης δρομολόγησης οχήματος 1 με χρήση Nearest Insertion.	93
Εικόνα 7.3: Διάγραμμα απεικόνισης δρομολόγησης οχήματος 1 με χρήση OR-TOOLS μέσω Badger Maps.	94
Εικόνα 7.4: Διάγραμμα απεικόνισης δρομολόγησης οχήματος 1 με χρήση Nearest Insertion μέσω Badger Maps.	94
Εικόνα 7.5: Διάγραμμα απεικόνισης δρομολόγησης οχήματος 2 με χρήση OR-TOOLS.	96
Εικόνα 7.6: Διάγραμμα απεικόνισης δρομολόγησης οχήματος 2 με χρήση Nearest Insertion.	96
Εικόνα 7.7: Διάγραμμα απεικόνισης δρομολόγησης οχήματος 2 με χρήση OR-TOOLS μέσω Badger Maps.	97
Εικόνα 7.8: Διάγραμμα απεικόνισης δρομολόγησης οχήματος 2 με χρήση Nearest Insertion μέσω Badger Maps.	97
Εικόνα 7.9: Διάγραμμα απεικόνισης δρομολόγησης οχήματος 3 με χρήση OR-TOOLS.	99
Εικόνα 7.10: Διάγραμμα απεικόνισης δρομολόγησης οχήματος 3 με χρήση Nearest Insertion.	100
Εικόνα 7.11: Διάγραμμα απεικόνισης δρομολόγησης οχήματος 3 με χρήση OR-TOOLS μέσω Badger Maps.	100

Εικόνα 7.12: Διάγραμμα απεικόνισης δρομολόγησης οχήματος 3 με χρήση Nearest Insertion μέσω Badger Maps.	101
Εικόνα 7.13: Διάγραμμα απεικόνισης δρομολόγησης οχήματος 4 με χρήση OR-TOOLS.	102
Εικόνα 7.14: Διάγραμμα απεικόνισης δρομολόγησης οχήματος 4 με χρήση Nearest Insertion.	103
Εικόνα 7.15: Διάγραμμα απεικόνισης δρομολόγησης οχήματος 4 με χρήση OR-TOOLS μέσω Badger Maps.	103
Εικόνα 7.16: Διάγραμμα απεικόνισης δρομολόγησης οχήματος 4 με χρήση Nearest Insertion μέσω Badger Maps.	104

1 ΚΕΦΑΛΑΙΟ: ΕΙΣΑΓΩΓΗ

Στο σημερινό έντονα ανταγωνιστικό επιχειρηματικό περιβάλλον, οι εταιρείες βρίσκονται αντιμέτωπες με μια σειρά προκλήσεων που απαιτούν την άμεση βελτιστοποίηση των διαδικασιών τους. Τα ζητήματα που τίθενται υπό βελτιστοποίηση περιλαμβάνουν τη μείωση του κόστους παραγωγής, την παροχή προϊόντων υψηλής ποιότητας και προδιαγραφών, την προσφορά προσιτών τιμών και την ανάγκη για ταχύτερες παραδόσεις. Όλα αυτά τα ζητήματα φέρουν τις επιχειρήσεις στο να προσαρμοστούν στις νέες συνθήκες της αγοράς, γεγονός που τις ωθεί στην αναζήτηση βέλτιστων πρακτικών ώστε να καινοτομούν στη διαχείριση των πόρων και των διαδικασιών τους. Όπως παρατηρείται άλλωστε και τα τελευταία χρόνια, η εφαρμογή νέων τεχνολογικών μέσων, διαδικασιών και εφαρμογών για τη βελτιστοποίηση της λειτουργίας των εταιριών έχει καταστεί ουσιαστική. Αυτό στοχεύει είτε στη μεγιστοποίηση του κέρδους είτε στην ελαχιστοποίηση των εξόδων. Ως εκ τούτου, η ικανότητα καινοτομίας και η στόχευση στη συνεχή βελτιστοποίηση αποτελούν αναγκαίες προϋποθέσεις για την επιβίωση και την ανάπτυξη μιας επιχείρησης στο σύγχρονο επιχειρηματικό περιβάλλον.

Τα συστήματα μεταφοράς αποτελούν ένα από τα σημαντικότερα μερίδια που δαπανώνται στην εφοδιαστική αλυσίδα και αποτελούν βασικό κριτήριο για την οικονομία ενός οργανισμού ή μιας επιχείρησης ενώ συμβάλλουν ουσιαστικά στην υλοποίηση των επιχειρηματικών στόχων που θέτονται, αποτελώντας πυλώνα του παγκόσμιου εμπορίου. Η μείωση του κόστους μεταφοράς, του χρόνου ταξιδιού και η ανάγκη για άμεση εύρεση μέσων μεταφοράς τα οποία θα είναι και φιλικότερα προς το περιβάλλον, καθιστούν επιτακτική την ανάπτυξη πιο οργανωμένων και αποτελεσματικών συστημάτων μεταφοράς.

Το Πρόβλημα Δρομολόγησης Οχημάτων (VRP) θεωρείται σημαντικό ερευνητικό κεφάλαιο λόγω της αποτελεσματικότητάς του στον χειρισμό πραγματικών προβλημάτων δρομολόγησης. Η εξέλιξη της τεχνολογίας και η ανάπτυξη μεθόδων διαχείρισης του VRP συνέβαλαν στη συνεχή βελτίωση της διαχείρισης και στην αποτελεσματική μεταφορά φορτίου, στην ταχεία παράδοση και στη βέλτιστη ποιότητα εξυπηρέτησης, μειώνοντας παράλληλα το κόστος μεταφοράς και συμβάλλοντας στην εξοικονόμηση ενέργειας.

Σε αυτή την εργασία, γίνεται εστίαση στην ανάπτυξη και την επίλυση του βασικού προβλήματος δρομολόγησης οχημάτων σε πραγματικά δίκτυα διανομής μιας επιχείρησης, όπου ο στόχος είναι να ελαχιστοποιηθεί το κόστος διαδρομής και να βρεθεί η συντομότερη διαδρομή που καλούνται να ακολουθήσουν τα οχήματα. Ειδικότερα, μελετάται το πρόβλημα του προγραμματισμού στόλου οχημάτων με περιορισμούς χωρητικότητας για εταιρεία διανομής προϊόντων. Στην ανάπτυξη του οδικού δικτύου χρησιμοποιήθηκαν γεωγραφικά δεδομένα από το Google Maps, ενώ η λύση πραγματοποιήθηκε με τη χρήση του δωρεάν λογισμικού OR-Tools σε γλώσσα προγραμματισμού Python και σύγκριση της λύσης που λαμβάνουμε μέσω αυτού με τον αλγόριθμο πλησιέστερης εισαγωγής.

Από τα αποτελέσματα της μελέτης φάνηκε ότι η ακρίβεια και η αποδοτικότητα της δρομολόγησης των οχημάτων επηρεάζεται από τους περιορισμούς που τίθενται στο πρόβλημα, τα χαρακτηριστικά του οδικού δικτύου, τη γεωγραφική θέση των περιοχών εξυπηρέτησης και τις τεχνολογικές υποδομές που υπάρχουν διαθέσιμες στην εταιρεία.

1.1 ΔΙΑΧΕΙΡΙΣΗ ΕΦΟΔΙΑΣΤΙΚΗΣ ΑΛΥΣΙΔΑΣ ΚΑΙ LOGISTICS

Στο περιβάλλον των επιχειρήσεων, συχνά παρατηρούμε ότι οι όροι Διοίκηση Εφοδιαστικής Αλυσίδας και τα Logistics χρησιμοποιούνται σαν ισοδύναμοι. Η άποψη ότι δεν υπάρχει διαφορά μεταξύ της Διοίκησης Εφοδιαστικής Αλυσίδας (Supply Chain Management) και των Logistics είναι αρκετά διαδεδομένη, αλλά οι δύο όροι ορίζονται διαφορετικά, παρόλο που υπάρχει μεγάλη συσχέτιση μεταξύ τους. Η αλυσίδα εφοδιασμού περιλαμβάνει όλα τα στάδια εκείνα που απαιτούνται από την απόκτηση και την αποστολή των αρχικών υλικών που χρειάζονται για την κατασκευή ενός προϊόντος μέχρι την τελική του παράδοση στο σημείο πώλησης. Τα logistics, από την άλλη, αποτελούν ένα κρίσιμο συστατικό της διαχείρισης της εφοδιαστικής αλυσίδας καθώς περιλαμβάνουν τη κυκλοφορία και την αποθήκευση αγαθών εντός της αλυσίδας αυτής.

Με άλλα λόγια, η διαχείριση εφοδιαστικής αλυσίδας καθορίζει τη στρατηγική και κατευθύνει τις καθημερινές δραστηριότητες που συμβαίνουν σε εργοστάσια, αποθήκες,

τοπικά κέντρα διανομής και άλλες εγκαταστάσεις. Τα logistics είναι μια πτυχή του supply chain που πραγματοποιεί ενέργειες όπως αποθήκευση ή παράδοση τελικών προϊόντων ή υπηρεσιών σε πελάτες, είτε είναι κατασκευαστής, διανομέας ή καταναλωτής. Ο στόχος των logistics είναι η έγκαιρη παράδοση των αγαθών και των υπηρεσιών στον πελάτη σε ανταγωνιστική τιμή. Ακολούθως, θα πραγματοποιηθεί μια αναλυτικότερη περιγραφή των δυο εννοιών.

1.1.1. ΕΦΟΔΙΑΣΤΙΚΗ ΑΛΥΣΙΔΑ

Η εφοδιαστική αλυσίδα έχει αναδειχθεί ως ένα κρίσιμο μέσο με το οποίο οι εταιρείες μπορούν να αποκτήσουν ανταγωνιστικό πλεονέκτημά πάνω σε ζητήματα που αφορούν τη λειτουργία τους εφόσον προβούν στην αποτελεσματική διαχείρισή τους. Ορίζοντας μια εφοδιαστική αλυσίδα, δεν νοείται μόνο η ροή των πρώτων υλών που μεταφέρεται από έναν προμηθευτή ή τον κατασκευαστή αυτών μέχρι τον καταναλωτή, αλλά επιπλέον και η διάδοση των διάφορων πληροφοριών μεταξύ όλων των μελών που σχετίζονται με την εκάστοτε αλυσίδα. Αποτελεί, δηλαδή, ένα σύστημα στο οποίο επιτρέπεται η ροή των προϊόντων, οι παρεχόμενες υπηρεσίες, πληροφορίες, ακόμη και τους οργανισμούς που εμπλέκονται σε κάθε στάδιο, από την πρώτη φάση της παραγωγής έως την παράδοση του τελικού προϊόντος στον καταναλωτή.

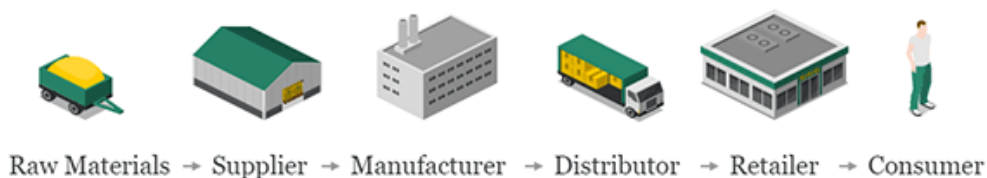
Τα μέλη που απαρτίζουν τη Διοίκηση Αλυσίδας Εφοδιασμού, όπως παραγωγοί, προμηθευτές, διανομείς κ.α. επιδιώκουν να ελέγχουν, να συντονίζουν και να βελτιστοποιούν όλες τις διαδικασίες που εμπλέκονται στην παραγωγή και διανομή προϊόντων, από την αρχική προμήθεια των υλικών μέχρι την τελική παράδοση των προϊόντων στους καταναλωτές. Μέσω της αποτελεσματικής διαχείρισης της εφοδιαστικής αλυσίδας υπάρχει βελτιστοποίηση στη διαχείριση των διαδικασιών, αρμονική συνεργασία μεταξύ προμηθευτών και πελατών ώστε να προσφέρεται στην επιχείρηση ανταγωνιστικό πλεονέκτημα προκειμένου να μπορούν να μειώσουν το κόστος, να βελτιώσουν την ταχύτητα και την αξιοπιστία των παραδόσεων και να ανταποκρίνονται καλύτερα στις απαιτήσεις των πελατών και της αγοράς. Η επίτευξη αυτού του στόχου, μπορεί να πραγματοποιηθεί όταν υπάρχει αυστηρός έλεγχος σε μια σειρά

διαδικασιών που περιλαμβάνει την απαιτούμενη ποσότητα παραγωγής, τη βέλτιστη διαχείριση στα αποθέματα, στην αλληλουχία παραλαβή-πώληση-διανομή και στη σωστή διαχείριση και εκμετάλλευση των επιστρεφόμενων προϊόντων.

Όσον αφορά στον ορισμό της Διαχείρισης της Εφοδιαστικής Αλυσίδας, όπως αυτός αναφέρεται από το Συμβούλιο Διαχείρισης Αλυσίδων Εφοδιασμού (Council of Supply Chain Management Professionals) [1]:

«Η διαχείριση της εφοδιαστικής αλυσίδας περιλαμβάνει τον προγραμματισμό και τη διαχείριση όλων των δραστηριοτήτων που εμπλέκονται στην προμήθεια, τη μετατροπή και όλες τις δραστηριότητες διαχείρισης της εφοδιαστικής αλυσίδας. Είναι σημαντικό ότι περιλαμβάνει επίσης συντονισμό και συνεργασία με τους εταίρους, οι οποίοι μπορεί να είναι προμηθευτές, μεσάζοντες, πάροχοι υπηρεσιών και πελάτες. Στην ουσία, η διαχείριση της εφοδιαστικής αλυσίδας ενσωματώνει τη διαχείριση προσφοράς και ζήτησης εντός και μεταξύ των εταιρειών.»

Supply Chain Diagram



Εικόνα 1.1: Διαδικασία της εφοδιαστικής αλυσίδας, από την απόκτηση των πρώτων υλών μέχρι την παράδοση του τελικού προϊόντος στον καταναλωτή.

1.1.2. LOGISTICS

Ο όρος Logistics, ο οποίος χρησιμοποιείται σε παγκόσμιο επίπεδο, ξεκίνησε να χρησιμοποιείται για στρατιωτικούς σκοπούς, γύρω στη δεκαετία του 1950 καθώς σχετιζόταν με τον τρόπο παραλαβής, αποθήκευσης και διανομής στρατιωτικού εξοπλισμού, απαιτούμενων προμηθειών και διαχείρισης προσωπικού.

Το 1964 τα Logistics εισήχθησαν για πρώτη φορά ως επιχειρηματική διαδικασία και ονομάστηκαν Business Logistics τα οποία τότε αφορούσαν μόνο τη φυσική διανομή των αγαθών. Η μελέτη και η πρακτική αυτής της διαδικασίας συζητήθηκαν μεταξύ της δεκαετίας του 1960 και του 1970 [2]. Όπως δίνεται και από τον Sheffi (1997) [3], η εφοδιαστική ασχολήθηκε μόνο με τη βελτιστοποίηση των τριών P:

1. Place (Τοποθεσία και προορισμός) - Προσθήκη αξίας θέσης σε αντικείμενα μετακινώντας τα από θέσεις χαμηλότερης αξίας σε θέσεις υψηλότερης αξίας για τον πελάτη.
2. Period & Pace (Time value) - Προσδίδοντας χρονική αξία στα στοιχεία με την αποθήκευσή τους και (i) μεταφέροντας τα στοιχεία από τις περιόδους που είναι διαθέσιμα (μετά την εξόρυξη, τη συγκομιδή ή την κατασκευή) στις περιόδους που τα απαιτούν οι πελάτες και (ii) καθιστώντας όλες τις διαδικασίες πιο αποτελεσματικές, τα αποθέματα επιτρέπουν την αποσύνδεση των διαδικασιών κατά μήκος της αλυσίδας αξίας μεταξύ τους, έτσι ώστε η καθεμία να μπορεί να λειτουργεί με τον βέλτιστο ρυθμό σύμφωνα με τα δικά της οικονομικά χαρακτηριστικά. Αυτό αντικατοπτρίζεται στη διαχείριση αποθεμάτων μαζί με τη ροή των αγαθών.
3. Pattern (Μορφές παραγγελίας) - Προσθήκη αξίας παραγγελίας στα είδη με τη διάταξή τους σε επιθυμητές ποσότητες και μοτίβα. Δημιουργία αξίας μέσω παραγγελίας εστιάζοντας στην επιθυμητή μορφή αγαθών.

Ο κλάδος των logistics λοιπόν αποτελείται από μια σειρά αλληλένδετων δραστηριοτήτων και λειτουργιών που έχουν σχεδιαστεί για να συμβάλουν στη βελτιστοποίηση της ροής των προϊόντων και να διασφαλίζουν ότι αυτά φτάνουν στον σωστό προορισμό εγκαίρως. Τα logistics αναφέρονται στον σχεδιασμό και την υλοποίηση της αποθήκευσης και της μεταφοράς αγαθών μεταξύ διαφορετικών σημείων της εφοδιαστικής αλυσίδας. Συγχρόνως, συντονίζουν τις εγκαταστάσεις, το προσωπικό, τον εξοπλισμό και άλλους πόρους ώστε να διασφαλίζεται ότι τα προϊόντα καταλαμβάνουν τον κατάλληλο χώρο μέσα στην αποθήκη για να τοποθετηθούν και να διανεμηθούν αποτελεσματικά.

Ο σχεδιασμός της ζήτησης, η μεταφορά και η διαχείριση του στόλου μιας εταιρίας, η διαχείριση αποθεμάτων, ο χειρισμός υλικών και η εκπλήρωση παραγγελιών είναι όλες οι διαδικασίες που απαρτίζουν την εφοδιαστική.

Ο ορισμός της διαχείρισης Logistics της CSCMP [1] αναφέρει ότι:

«Η διαχείριση logistics είναι εκείνο το μέρος της διαχείρισης της εφοδιαστικής αλυσίδας που σχεδιάζει, εφαρμόζει και ελέγχει την αποδοτική, αποτελεσματική προώθηση και αντιστρέφει τη ροή και την αποθήκευση αγαθών, υπηρεσιών και σχετικών πληροφοριών μεταξύ του σημείου προέλευσης και του σημείου κατανάλωσης, προκειμένου να ανταποκριθεί στις απαιτήσεις των πελατών.»

Υπάρχουν διάφοροι τύποι logistics που πρέπει να προσέχουν οι επιχειρήσεις.

- Inbound Logistics

Τα εισερχόμενα logistics αναφέρονται στη μεταφορά πρώτων υλών από μια εξωτερική πηγή σε έναν κατασκευαστή και επικεντρώνονται κυρίως στην πτυχή του εφοδιασμού της διαχείρισης της εφοδιαστικής αλυσίδας. Κατά τη διάρκεια αυτής της διαδικασίας, το κάθε αρμόδιο τμήμα παραγγέλλει τα εμπορεύματα, τα παραλαμβάνει, τα αποθηκεύει και τα μετατρέπει σε προϊόντα.

- Outbound Logistics

Το επόμενο στάδιο στην αλυσίδα εφοδιασμού είναι τα εξερχόμενα logistics, τα οποία περιλαμβάνουν τη μεταφορά και αποθήκευση προϊόντων από τον κατασκευαστή στον πελάτη. Κατά τη διάρκεια αυτής της διαδικασίας, οι δραστηριότητες περιλαμβάνουν συνήθως συσκευασία, αποστολή και αποθήκευση, με απώτερο στόχο να διασφαλιστεί ότι τα προϊόντα φτάνουν στον προορισμό τους εγκαίρως.

- Reverse Logistics

Η αντίστροφη εφοδιαστική είναι το τμήμα των logistics που συνεπάγεται τη διακίνηση και διαχείριση αγαθών και πόρων μετά την πώλησή τους και την παράδοσή τους στον πελάτη. Η αντίστροφη εφοδιαστική περιλαμβάνει την επιστροφή προϊόντων στον λιανοπωλητή ή τον κατασκευαστή για επισκευή, ανταλλαγή ή πίστωση στον πελάτη.

1.1.3. ΔΙΑΦΟΡΕΣ SUPPLY CHAIN MANAGEMENT & LOGISTICS

Για να εξηγηθούν περαιτέρω οι διαφορές και οι ομοιότητες μεταξύ των Logistics και της Διαχείρισης της Εφοδιαστικής Αλυσίδας, δημιουργήθηκε ο κάτωθι πίνακας.

Πίνακας 1.1: Διαφορές και ομοιότητες μεταξύ Logistics και Διαχείρισης Εφοδιαστικής Αλυσίδας.

LOGISTICS	SUPPLY CHAIN MANAGEMENT
Τα logistics αποτελούν μια από τις δραστηριότητες της διαχείρισης της εφοδιαστικής αλυσίδας και επικεντρώνονται σε δραστηριότητες που αφορούν τη μεταφορά, αποθήκευση και το απόθεμα.	Η διαχείριση της εφοδιαστικής αλυσίδας καλύπτει ένα ευρύ φάσμα δραστηριοτήτων, συμπεριλαμβανομένου του σχεδιασμού, της προμήθειας υλικών, της διαχείρισης της εργασίας και των εγκαταστάσεων, της παραγωγής και της παράδοσης αυτών των αγαθών και υπηρεσιών.
Συνήθως ασχολείται με καθημερινές και τακτικές δραστηριότητες. Επικεντρώνεται στην εκτέλεση καθηκόντων για τη διασφάλιση της ομαλής ροής, της ακριβούς διαχείρισης αποθεμάτων και της έγκαιρης εκπλήρωσης παραγγελιών.	Λαμβάνει μια μακροπρόθεσμη προοπτική και περιλαμβάνει τη λήψη στρατηγικών αποφάσεων, όπως ο σχεδιασμός του δικτύου, η επιλογή προμηθευτών και η πρόβλεψη της ζήτησης. Αντιμετωπίζει ολόκληρο τον κύκλο ζωής ενός προϊόντος ή μιας υπηρεσίας, από την προμήθεια πρώτων υλών έως την τελική παράδοση στον πελάτη.
Κύριος στόχος η εξασφάλιση της ικανοποίησης των πελατών. Συμβάλλει στην ικανοποίηση των πελατών διασφαλίζοντας ότι τα προϊόντα	Έμφαση στο ανταγωνιστικό πλεονέκτημα. Εστιάζει στην κατανόηση των αναγκών των πελατών, των προτύπων ζήτησης και της δυναμικής

παραδίδονται εγκαίρως και σε καλή κατάσταση. Επικεντρώνεται στην ικανοποίηση των προσδοκιών των πελατών όσον αφορά την εκτέλεση των παραγγελιών, την αξιοπιστία των μεταφορών και τη διαθεσιμότητα των προϊόντων.	της αγοράς. Στοχεύει στη δημιουργία αξίας για τους πελάτες με την παράδοση των σωστών προϊόντων, στο σωστό χρόνο, στις σωστές ποσότητες και στο σωστό κόστος.
Επικεντρώνεται στην απόδοση ως προς το κόστος και το χρόνο παράδοσης των αγαθών.	Επικεντρώνεται στην καινοτόμο προσέγγιση και τη μείωση του κόστους σε κάθε βήμα της διαδικασίας.
Τα logistics είναι μια δραστηριότητα που εντάσσεται στο πλαίσιο της αλυσίδας εφοδιασμού.	

1.2 ΛΕΙΤΟΥΡΓΙΕΣ ΕΦΟΔΙΑΣΤΙΚΗΣ ΑΛΥΣΙΔΑΣ

Η εφοδιαστική αλυσίδα αποτελείται από πληθώρα διαδικασιών και δραστηριοτήτων που επιτρέπουν τη ροή των προϊόντων ή των υπηρεσιών από το σημείο παραγωγής ή απόκτησής τους έως τον τελικό καταναλωτή. Ανάλογα με τον τομέα και τις ανάγκες της επιχείρησης, αυτές οι διαδικασίες μπορεί να διαφέρουν, αλλά μερικές κοινές και βασικές διαδικασίες περιλαμβάνουν:

- Προμήθεια ή αγορά:

Καθώς οι επιχειρήσεις αντιμετωπίζουν την ανάγκη για αποτελεσματική παραγωγή ή παροχή προϊόντων και υπηρεσιών, το τμήμα αυτό είναι υπεύθυνο ώστε να εξασφαλίζεται η σωστή ποσότητα προμήθειων των απαραίτητων πρώτων υλών ή προϊόντων για την επίτευξη αυτών των στόχων. Οι προμήθειες περιλαμβάνουν την εύρεση αξιόπιστων προμηθευτών, τη διαπραγμάτευση συμβάσεων και την εκτέλεση και οργάνωση των παραδόσεων-παραγγελιών. Μέσω της στρατηγικής διαχείρισης των προμηθειών, οι επιχειρήσεις μπορούν να επιτύχουν αποτελεσματική διαχείριση του κόστους, παραγωγή

ποιοτικών προϊόντων και διαχείριση χρόνου παράδοσης, καθώς και να αντιμετωπίσουν όσους ενδεχόμενους κινδύνους σχετίζονται με την προμήθεια.

- **Μεταφορά και διανομή:**

Οι διαδικασίες της μεταφοράς και διανομής αποτελούν κρίσιμο κομμάτι της εφοδιαστικής αλυσίδας, επιτρέποντας την ομαλή μετακίνηση των προϊόντων από το σημείο παραγωγής ή αποθήκευσης έως τον τελικό προορισμό τους. Η αποτελεσματική μεταφορά απαιτεί συστηματικό σχεδιασμό και διαχείριση των διαδικασιών μεταφοράς, συμπεριλαμβανομένης της επιλογής των κατάλληλων μεθόδων και μέσων μεταφοράς (όπως οδική, θαλάσσια, αεροπορική, σιδηροδρομική μεταφορά), τη διαχείριση του δικτύου μεταφορών και την παρακολούθηση των αποστολών. Η αποτελεσματική διαχείριση αυτής, συμβάλλει στη μείωση των χρόνων παράδοσης, την αυξημένη ασφάλεια των προϊόντων κατά τη μεταφορά τους και τη βελτίωση της ικανοποίησης των πελατών.

- **Αποθήκευση:**

Η κατάλληλη διαχείριση της αποθήκευσης περιλαμβάνει την επιλογή των κατάλληλων αποθηκευτικών χώρων, την οργάνωση των προϊόντων σύμφωνα με συγκεκριμένα κριτήρια, όπως η κατηγοριοποίηση και η σήμανση και τη διαχείριση του αποθεματικού με στόχο τη μείωση του κινδύνου εξάντλησης ή υπερπληρότητας. Κατά το στάδιο της αποθήκευσης περιλαμβάνονται ακόμη οι ενέργειες που αφορούν τη διαχείριση του απαιτούμενου χώρου όπως ο σωστός σχεδιασμός του χώρου παραλαβών, αποστολών, διαχείρισης αποθέματος, της αποθήκευσης, της διαδικασίας που περιλαμβάνει την προετοιμασία των παραγγελιών και αποστολής αυτών.

- **Διαχείριση αποθέματος:**

Κατά τη διαχείριση του αποθέματος πραγματοποιείται παρακολούθηση των επιπέδων αποθεμάτων, με στόχο τη διασφάλιση της συνεχούς διαθεσιμότητας των προϊόντων και την ελαχιστοποίηση του κόστους αποθήκευσης. Η αποτελεσματική διαχείριση του αποθέματος περιλαμβάνει την πρόβλεψη της ζήτησης, την λήψη παραγγελιών, την παρακολούθηση των επιπέδων αποθεμάτων και τη διατήρηση των

χαμηλότερων δυνατών επιπέδων αποθέματος που μπορούν να ανταπεξέλθουν στις απαιτήσεις των πελατών με το ελάχιστο δυνατό κόστος.

- Διαχείριση παραγγελιών:

Η διαχείριση και η παραλαβή παραγγελιών περιλαμβάνουν την επεξεργασία των παραγγελιών από τους πελάτες, την επιλογή των κατάλληλων προϊόντων και τη διαχείριση τους στο σύστημα παραγγελιών της εταιρείας. Η παραλαβή παραγγελιών αφορά την αποθήκευση, την αντιστοίχιση με τις παραγγελίες και την προετοιμασία για την αποστολή τους προς τους πελάτες.

- Εγκαταστάσεις / εξοπλισμός:

Αφορά τη διαχείριση των κινητών και ακινήτων περιουσιακών στοιχείων που χρησιμοποιούνται κατά τη διάρκεια των δραστηριοτήτων της επιχείρησης. Μεταξύ αυτών των περιουσιακών στοιχείων ανήκουν εξοπλισμός, μηχανήματα, οχήματα, κτίρια και άλλα στοιχεία που είναι αναγκαία για την παραγωγή ή την παροχή προϊόντων και υπηρεσιών. Περιλαμβάνουν την παρακολούθηση του χρόνου ζωής και της κατάστασης των περιουσιακών στοιχείων, τον προγραμματισμό των συντηρήσεων και αναβαθμίσεων, καθώς και την αξιολόγηση της απόδοσής τους.

- Διαχείριση επιστροφών:

Πολλές φορές η λειτουργία της εφοδιαστικής δεν ολοκληρώνεται τη στιγμή της παράδοσης της παραγγελίας στον πελάτη. Σε περιπτώσεις όπου υπάρχουν προβλήματα με τις παραγγελίες, όπως ελαττωματικά προϊόντα ή λανθασμένες ποσότητες (ελλειμματικές ή πλεονασματικές παραδόσεις) ή ακόμη και για επιστροφή ληγμένων προϊόντων, είναι απαραίτητο να γίνει η συλλογή, η καταγραφή και η επιστροφή τους. Προϊόντα όπως παλέτες, κιβώτια, τελάρα και δοχεία όπου γενικά χρησιμοποιούνται κατά την διανομή και παράδοση μπορούν να επιστραφούν για επαναχρησιμοποίηση πίσω στους προμηθευτές, ενώ άλλα υλικά, όπως μέταλλα, γυαλί, χαρτί, πλαστικά και έλαια, μπορούν να υποστούν ανακύκλωση. Οι δραστηριότητες που αφορούν την επιστροφή υλικών πίσω σε έναν οργανισμό ονομάζονται αντίστροφη εφοδιαστική ή αντίστροφη διανομή (reverse logistics).

- Συστήματα πληροφόρησης:

Ένα σύστημα πληροφόρησης αποτελεί σημαντικό ρόλο στην παροχή υπηρεσιών στους πελάτες. Για παράδειγμα, η δυνατότητα ειδοποίησης των πελατών για την ακριβή τοποθεσία των αγαθών και την ακριβή ώρα άφιξης (order tracking) μπορεί να βελτιώσει τις επιχειρηματικές δραστηριότητες και να αυξήσει την ικανοποίηση των πελατών από την παρεχόμενη υπηρεσία. Η χρήση εργαλείων για τον εντοπισμό, την πρόσβαση, την αποθήκευση, την ανάλυση, την ανάκτηση πληροφοριών και την υποστήριξη της λήψης αποφάσεων είναι κρίσιμη για τις λειτουργίες της εφοδιαστικής.

Γενικά υπάρχουν πολλές ακόμη δραστηριότητες που μπορούν να ενταχθούν οι οποίες σχετίζονται με τη ζήτηση, τις ανάγκες και τις προτεραιότητες του οργανισμού. Ενδεικτικά παραδείγματα είναι η εξυπηρέτηση των πελατών, οι προβλέψεις πωλήσεων για τη βελτιστοποίηση της διαχείρισης των αποθεμάτων, ο προγραμματισμός παραγωγής και η διασύνδεση με εξωτερικούς φορείς όπως προμηθευτές και συνεργάτες.

Μεταξύ όλων των σταδίων πρέπει να υπάρχει συντονισμός για τη δημιουργία μιας αποτελεσματικής ροής υλικών από το αρχικό σημείο είτε αφορά στην παραγωγή είτε στην προμήθεια έως τον τελικό προορισμό. Αυτή η συνεργασία μεταξύ των διαφόρων λειτουργικών ομάδων είναι κρίσιμη για τη διασφάλιση της ομαλής λειτουργίας της εφοδιαστικής αλυσίδας και την επίτευξη των στρατηγικών στόχων της επιχείρησης.

1.3 ΔΙΚΤΥΑ ΔΙΑΝΟΜΗΣ

Τα κανάλια διανομής στα logistics συνδέουν στοιχεία μεταξύ παραγωγής και κατανάλωσης, διασφαλίζοντας ότι τα αγαθά φτάνουν αποτελεσματικά στον τελικό πελάτη. Είναι στρατηγικά δίκτυα που υποστηρίζουν την αποτελεσματική διαχείριση της εφοδιαστικής αλυσίδας. Συγκεκριμένα, διασφαλίζουν ότι η διαχείριση της ομαλής ροής των προϊόντων ώστε να ανταποκρίνεται στις απαιτήσεις των πελατών. Η σημασία τους δεν μπορεί να υπερεκτιμηθεί καθώς επηρεάζουν άμεσα την ικανοποίηση των πελατών και τη συνολική επιτυχία των λειτουργιών της εφοδιαστικής αλυσίδας. Ένα σύστημα διανομής αναφέρεται σε

ένα σύνολο διαδικασιών, μεθόδων, μέσων και πρακτικών που χρησιμοποιούνται για τη μεταφορά αγαθών από τον αρχικό προμηθευτή μέχρι τον τελικό προορισμό. Αφορά είτε στη μεταφορά εμπορευμάτων είτε επιβατών από μία αρχική τοποθεσία (σημείο αναχώρησης) σε μία άλλη επιθυμητή (σημείο προορισμού), παρέχοντας πάντα στον πελάτη το όσο το δυνατόν καλύτερο επίπεδο εξυπηρέτησης.

Η εφοδιαστική φυσικά στοχεύει στην παράδοση με το ελάχιστο δυνατό κόστος, καθώς η διανομή αποτελεί το κύριο μέλημα των εταιρειών αλλά και του συνολικού κλάδου των logistics καθώς αποτελεί περίπου το 40-50% του συνολικού κόστους των logistics και το 4 με 10% της τελικής τιμής πώλησης των τελικών προϊόντων, σύμφωνα με το Global Trail Platform [4].

Γενικά τα κανάλια διανομής μπορούν να κατηγοριοποιηθούν σε [5]:

- Κανάλια άμεσης διανομής:

Αυτά τα κανάλια δίνουν τη δυνατότητα στους κατασκευαστές να πωλούν απευθείας στον τελικό καταναλωτή, παρακάμπτοντας τους μεσάζοντες. Αυτή η στρατηγική προωθεί μια άμεση σύνδεση με τον πελάτη, γεγονός που συχνά βελτιώνει την ικανοποίηση και την εξυπηρέτησή του.

- Κανάλια έμμεσης διανομής:

Με τη συμμετοχή ενός ή περισσότερων διαμεσολαβητών, όπως χονδρέμποροι ή καταστήματα λιανικής, μπορούν να επεκτείνουν τη γκάμα συναλλαγών των προϊόντων και να φτάσουν σε μια ευρύτερη αγορά και να επιτύχουν έναν καθορισμένο στόχο. Είναι ζωτικής σημασίας για επιχειρήσεις που δεν μπορούν να χειριστούν τη διανομή ανεξάρτητα λόγω κλίμακας, κόστους ή πολυπλοκότητας των logistics.

- Υβριδικά κανάλια διανομής:

Συνδυάζοντας στοιχεία άμεσων και έμμεσων μεθόδων, τα υβριδικά κανάλια προσφέρουν ευελιξία στη στρατηγική διανομής, επιτρέποντας στις επιχειρήσεις να προσαρμόσουν την προσέγγισή τους σε διαφορετικά τμήματα της αγοράς ή τύπους προϊόντων για βέλτιστη διείσδυση στην αγορά.

1.3.1. THIRD PARTY LOGISTICS

Το σύστημα του Τρίτου Παρόχου Υπηρεσιών Logistics (3PL) αποτελεί μια στρατηγική προσέγγιση στη διαχείριση της εφοδιαστικής αλυσίδας. Συγκεκριμένα, οι υπηρεσίες που παρέχονται από τους 3PL ανήκουν στην κατηγορία των εξωτερικών υπηρεσιών logistics όπου ορισμένες εταιρίες εμπιστεύονται σε έναν εξωτερικό πάροχο υπηρεσιών τις λειτουργίες logistics της εταιρείας τους. Οι υπηρεσίες που παρέχονται από έναν 3PL παρέχουν σε επιχειρήσεις τη δυνατότητα να εστιάσουν στον πυρήνα των δραστηριοτήτων τους, ενώ η ίδια η 3PL εταιρία αναλαμβάνει την κάλυψη λειτουργιών όπως την αποθήκευση των εμπορευμάτων, τη διαχείριση των αποθεμάτων, την εξυπηρέτηση των πελατών μέσω της σωστής διαχείρισης των παραγγελιών, της διανομής και την επιστροφή εμπορευμάτων.

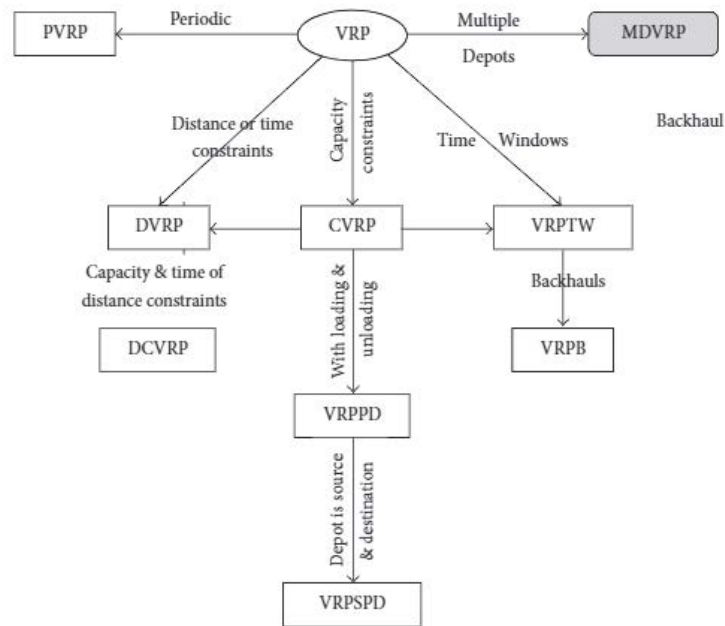
Μεταξύ των κύριων υπηρεσιών που προσφέρουν περιλαμβάνονται η συσκευασία και επισήμανση προϊόντων, η εκτέλεση παραγγελιών (picking and packing), η επισκευή και ανακατασκευή ελαττωματικών προϊόντων, η διαχείριση των επιστροφών και η αντικατάστασή τους σύμφωνα με τις πολιτικές των επιχειρήσεων καθώς και η παροχή εξειδικευμένων λύσεων για διάφορους τομείς ανάλογες με τις εκάστοτε συνθήκες που επικρατούν στην αγορά.

2 ΚΕΦΑΛΑΙΟ: ΔΡΟΜΟΛΟΓΗΣΗ ΟΧΗΜΑΤΩΝ

Στο πλέον ανταγωνιστικό περιβάλλον μεταξύ των εταιριών που διακρίνεται σήμερα, η ανάπτυξη ενός αποτελεσματικού συστήματος εφοδιαστικής αποτελεί ουσιώδες στοιχείο για την επιτυχία των επιχειρήσεων. Η βελτιστοποίηση της ροής προϊόντων ή πληροφοριών για να ικανοποιηθούν οι ανάγκες των πελατών είναι προτεραιότητα, με ένα από τα κύρια ζητήματα να είναι η αναζήτηση βέλτιστων διαδρομών που θα εξυπηρετούν τους πελάτες με τον πιο αποδοτικό και οικονομικό τρόπο. Το VRP αναλύεται ως μια διαδικασία συνδυαστικής βελτιστοποίησης, κατά την οποία αναζητείται το βέλτιστο σύνολο διαδρομών που πρέπει να ακολουθήσει ένας στόλος οχημάτων, ξεκινώντας και καταλήγοντας σε μία ή περισσότερες αποθήκες, εξασφαλίζοντας την εξυπηρέτηση πελατών και την ελαχιστοποίηση των δαπανών.

Καθημερινά προκύπτουν ποικίλες προκλήσεις στον σχεδιασμό δρομολόγησης, οι οποίες απαιτούν την εύρεση βέλτιστων λύσεων λαμβάνοντας υπόψη διαφορετικά δεδομένα και περιορισμούς. Αυτά τα προβλήματα περιλαμβάνουν την ανάγκη για βέλτιστη χρήση πόρων όπως χρόνος, κόστος και διαθεσιμότητα προϊόντων, με στόχο την αποδοτικότερη διαχείριση της αλυσίδας εφοδιασμού ή της διανομής εμπορευμάτων. Αυτό έχει οδηγήσει στην ανάπτυξη διαφορετικών ειδών προβλημάτων δρομολόγησης που αντιμετωπίζουν αυτά τα προβλήματα με διαφορετικούς τρόπους.

Σε αυτό το κεφάλαιο θα εξεταστεί το Πρόβλημα Δρομολόγησης Οχημάτων το οποίο αποτελεί ένα από τα κύρια ζητήματα που καλούνται να αντιμετωπίζουν οι επιχειρήσεις στον τομέα της εφοδιαστικής. Αρχικά, θα γίνει μια σύντομη αναφορά στο Πρόβλημα του Πλανόδιου Πωλητή (Travelling Salesman Problem, TSP), ένα απλούστερο πρόβλημα δρομολόγησης όπου ο πωλητής πρέπει να επισκεφτεί όλα τα σημεία πώλησης με τον ελάχιστο δυνατό κόστος. Έπειτα, παρουσιάζεται λεπτομερώς το VRP, περιγράφοντας τις διάφορες εκδοχές και τις πολυπλοκότητες που ενέχονται, όπως η προσθήκη χρονικών περιορισμών και οι περιορισμοί της χωρητικότητας των οχημάτων. Τέλος, εξετάζονται διάφορες επεκτάσεις του προβλήματος και προτείνονται λύσεις για την αποτελεσματική διαχείρισή του.



Εικόνα 2.1: Διαγραμματική συσχέτιση μεταξύ κοινών προβλημάτων VRP [6].

2.1 ΤΟ ΠΡΟΒΛΗΜΑ ΤΟΥ ΠΛΑΝΟΔΙΟΥ ΠΩΛΗΤΗ - TSP

Το πρόβλημα του πλανόδιου πωλητή (TSP) αποτελεί το πρωταρχικό πρόβλημα στη κατηγορία των προβλημάτων δρομολόγησης οχημάτων και είναι ένα από τα πιο γνωστά προβλήματα συνδυαστικής βελτιστοποίησης. Στόχος του TSP τίθεται η εύρεση της συντομότερης διαδρομής, είτε σε χρόνο, απόσταση ή κάποιου διαφορετικού κόστους, που καλείται να εκτελέσει ένας πωλητής ο οποίος ξεκινάει έχοντας μια αφετηρία (π.χ. αποθήκη, κέντρο διανομής) και αφού εξυπηρετήσει το πλήθος των πελατών που έχουν ανατεθεί σε αυτόν ακριβώς μια φορά, θα επιστρέψει πίσω στο σημείο εκκίνησής του.

Παρόλο που αυτό το πρόβλημα αποτελεί την απλούστερη μορφή των προβλημάτων δρομολόγησης έχοντας συγκεκριμένα δεδομένα και όχι ιδιαίτερα πολύπλοκους περιορισμούς, η επίλυσή του μπορεί είναι αρκετά δύσκολη και απαιτητική. Αυτό συμβαίνει σε περιπτώσεις, όπου για παράδειγμα, έχουμε ένα μεγάλο αριθμό πελατών (κόμβων) που πρέπει να εξυπηρετηθούν και προκύπτουν πολλές εναλλακτικές διαδρομές που μπορούν να αποτελέσουν τη βέλτιστη λύση. Για το λόγο αυτό, χρησιμοποιούνται ευρετικοί αλγόριθμοι, οι

οποίοι προσεγγίζουν τη βέλτιστη λύση με βάση συγκεκριμένες στρατηγικές. Αυτοί οι αλγόριθμοι βρίσκουν τον επόμενο κόμβο που θα επισκεφτούν τα οχήματα με βάση συγκεκριμένα κριτήρια όπως η απόσταση, ο χρόνος άφιξης, η προτεραιότητα της παράδοσης και με αυτόν τον τρόπο, επιτυγχάνεται μια λύση που είναι κοντά στη βέλτιστη, ενώ μειώνεται η απαιτούμενη χρονική και υπολογιστική πολυπλοκότητα.

2.1.1 ΜΟΝΤΕΛΟΠΟΙΗΣΗ TSP

Η μορφή του προβλήματος μπορεί να οριστεί ως εξής, χρησιμοποιώντας δυαδικές μεταβλητές απόφασης:

Δεδομένου ενός δικτύου που αποτελείται από n σημεία, ορίζουμε δυαδικές μεταβλητές απόφασης $x_{i,j}$ όπου:

- $x_{i,j} = 1$ αν στη διαδρομή υπάρχει απευθείας μετάβαση από τον κόμβο i στον κόμβο j
- $x_{i,j} = 0$ αλλιώς

ενώ η απόσταση από το κόμβο i στο κόμβο j ορίζεται ως $c_{i,j}$. Στόχος του προβλήματος είναι η εύρεση της ελάχιστης διαδρομής που ξεκινάει από τον κόμβο $i = 1$, θα επισκέπτεται ακριβώς μια φορά όλα τα σημεία του δικτύου και θα επιστρέφει πάλι στον κόμβο εκκίνησης.

Το πρόβλημα μορφοποιείται ως το ακόλουθο γραμμικό πρόβλημα:

$$\min \sum_{i=1}^n \sum_{j \neq i, j=1}^n c_{i,j} * x_{i,j} \quad \text{Εξ.2.1}$$

Υπό τους περιορισμούς,

$$\sum_{j:(i,j) \in A} x_{i,j} = 1, \quad \forall i = 1, 2, \dots, n \quad \text{Εξ.2.2}$$

$$\sum_{i:(i,j) \in A} x_{i,j} = 1, \quad \forall j = 1, 2, \dots, n \quad \text{Εξ.2.3}$$

$$U_i - U_j + n * x_{i,j} \leq n - 1, \quad 2 \leq i \neq j \leq n \quad \text{Εξ.2.4}$$

$$x_{i,j} \in \{0,1\}, i \in V, j \in V \quad \text{Εξ.2.5}$$

Οι εξισώσεις (Εξ.2.2) και (Εξ.2.3) διασφαλίζουν ότι από κάθε κόμβο ξεκινά και σε κάθε κόμβο τερματίζει μία μόνο διαδρομή. Ο τελευταίος περιορισμός επιβάλλει ότι υπάρχει μόνο μία διαδρομή που καλύπτει όλους τους κόμβους.

2.2 ΤΟ ΠΡΟΒΛΗΜΑ ΔΡΟΜΟΛΟΓΗΣΗΣ ΟΧΗΜΑΤΩΝ - VRP

Το Πρόβλημα Δρομολόγησης Οχημάτων αποτελεί μια κατηγορία προβλημάτων σχεδιασμού δρομολόγησης, τα οποία περιλαμβάνουν το Στατικό Πρόβλημα Δρομολόγησης Οχημάτων (SRP) και το Πρόβλημα Δυναμικής Δρομολόγησης Οχημάτων (DVRP). Οι εισροές στο VRP είναι είτε ντετερμινιστικές είτε γνωστές με βεβαιότητα είτε γνωστές με αβεβαιότητα είτε πιθανολογικές, δηλαδή ακολουθούν κάποια κατανομή πιθανοτήτων, σύμφωνα με τους Psaraftis, H. N., Wen, M., & Kontovas, C. A. (2016) [7].

Υπάρχουν διάφοροι τύποι προβλημάτων που βασίζονται στη δρομολόγηση και τον προγραμματισμό μεταφοράς οι οποίες περιλαμβάνουν την ανάθεση οχημάτων σε εργασίες παράδοσης ή παραλαβής και στοχεύουν στην ελαχιστοποίηση του κόστους ανάθεσης και του συνολικού κόστους δρομολόγησης. Αυτά τα προβλήματα είναι σημαντικά για τις βιομηχανίες μεταποίησης, τις εταιρείες παροχής υπηρεσιών και μεταφορών, όπως εργασίες ταχυμεταφορών, οι μεταφορές που εκτελούνται κατά παραγγελία και οι υπηρεσίες που παρέχουν τα ταξί.

Στο κλασικό VRP καθορίζεται ένα σύνολο διαδρομών, όπου στοχεύει στην ελαχιστοποίηση του συνολικού κόστους απόστασής τους. Η θέση εκκίνησης και η θέση τερματισμού της διαδρομής είναι οι ίδιες, που μπορεί να είναι μια αποθήκη, ένα κέντρο διανομής ή κάποιο άλλο σημείο εκκίνησης, διασφαλίζοντας την εξυπηρέτηση των απαιτήσεων σε κάθε κόμβο. Η χωρητικότητα των οχημάτων είναι περιορισμένη και κάθε κόμβος επισκέπτεται μόνο μία φορά από ένα όχημα. Υπάρχουν επίσης περιορισμοί που εφαρμόζονται στον μέγιστο χρόνο ταξιδιού σε ορισμένους τύπους VRP αλλά και άλλοι τύποι περιορισμών που δημιουργούν διαφορετικές κατηγορίες προβλημάτων όπως θα αναλυθούν στη συνέχεια του κεφαλαίου.

2.2.1 ΟΡΙΣΜΟΣ ΤΟΥ ΠΡΟΒΛΗΜΑΤΟΣ

Το Πρόβλημα Δρομολόγησης Οχημάτων αποτελεί επέκταση του προβλήματος του πλανόδιου πωλητή στις περιπτώσεις όπου ένα μόνο όχημα δεν δύναται να επισκεφθεί όλους τους πελάτες. Στο κλασικό πρόβλημα γίνεται αναζήτηση πολλαπλών κυκλικών διαδρομών, όπου όλες ξεκινάνε και τελειώνουν στον ίδιο κόμβο εκκίνησης αλλά κάθε μια επισκέπτεται ένα διαφορετικό σύνολο πελατών από κάποια άλλη. Όπως και στο TSP, στόχος του προβλήματος είναι η ελαχιστοποίηση του συνολικού κόστους των επιμέρους διαδρομών που εξυπηρετούν το σύνολο των πελατών.

Το Vehicle Routing Problem αποτελεί ένα σημαντικό πρόβλημα διανομής για πολλές επιχειρήσεις. Καθώς επιχειρήσεις σε διάφορους τομείς χρειάζονται να παραδίδουν αγαθά σε πολλούς πελάτες, η βέλτιστη οργάνωση των δρομολογίων των οχημάτων τους τίθεται κρίσιμης σημασίας για την ελαχιστοποίηση του κόστους και την αποτελεσματική εξυπηρέτηση των πελατών. Αναλύοντας το πρόβλημα, η πρώτη προτεραιότητα είναι ο υπολογισμός των βέλτιστων διαδρομών, οι οποίες είναι ικανές να καλύψουν όλες τις παραδόσεις με τον ελάχιστο αριθμό οχημάτων σε βέλτιστο χρόνο. Κατά την εκτέλεση των διαδρομών, ο συγκεκριμένος αριθμός οχημάτων πρέπει να είναι επαρκής ώστε να καλυφθεί η ζήτηση των πελατών, αλλά χωρίς περιττή χρήση αυτών.

Επίσης, είναι σημαντικό να ληφθούν υπόψη διάφοροι παράγοντες κατά τον υπολογισμό των δρομολογίων, όπως οι περιορισμοί σχετικά με τα χρονικά παράθυρα παράδοσης, οι περιορισμοί σχετικά με το φορτίο και οι περιορισμοί σχετικά με την πρόσβαση σε συγκεκριμένες περιοχές. Ένα πρόβλημα το οποίο θα αποτελείται από το συνδυασμό όλων αυτών των παραγόντων θα επιτρέψει τη δημιουργία εύρεσης βέλτιστων διαδρομών που θα παρέχουν αποτελεσματικότητα και οικονομία.

Το πρόβλημα αυτό, ξεκίνησε να εφαρμόζεται πρακτικά το 1959 από τους Dantzig και Ramser [8] έχοντας πραγματοποιήσει μια έρευνα σχετικά με την παράδοση βενζίνης σε σταθμούς εξυπηρέτησης, όπου χρησιμοποιώντας μία μέθοδο βασισμένη στις αρχές του Γραμμικού Προγραμματισμού κατέληξαν σε μία προσεγγιστικά βέλτιστη λύση.

Όπως θα παρουσιαστεί παρακάτω, το VRP μπορεί να διαμορφωθεί ανάλογα με τα χαρακτηριστικά και τις ιδιότητες του προβλήματος που ζητείται να επιλυθεί. Διαφορετικά χαρακτηριστικά μεταβάλλουν τους περιορισμούς και τους στόχους του προβλήματος και δημιουργούν παραλλαγές στο πρόβλημα δρομολόγησης. Ορισμένα ποιοτικά χαρακτηριστικά των προβλημάτων διανομής αναλύονται ακολούθως:

Χαρακτηριστικά των πελατών:

- Η γεωγραφική τους θέση (road graph), δηλαδή το σημείο πάνω σε ένα γράφημα διανομής που ανήκει ο συγκεκριμένος πελάτης.
- Η ποσότητα ζήτησης των προϊόντων που απαιτείται για την εξυπηρέτηση του κάθε πελάτη (demand), είτε αυτή αφορά παράδοση ή συλλογή αντικειμένων.
- Τα χρονικά περιθώρια (time windows) εντός μια ημέρας στα οποία πρέπει να ολοκληρωθεί η εξυπηρέτηση των πελατών.
- Ο χρόνος εξυπηρέτησης κάθε πελάτη (service time) ο οποίος συνδυάζει και τον χρόνο φορτό-εκφόρτωσης.

Φυσικά, οι πελάτες θα εξυπηρετηθούν και η μεταφορά των προϊόντων τους εντός του δικτύου θα πραγματοποιηθεί από τον στόλο των οχημάτων που θα χρησιμοποιηθεί από την εκάστοτε εταιρία που τα διαχειρίζεται. Η επιλογή και η διαχείριση του αριθμού και του τύπου των οχημάτων που θα χρησιμοποιηθούν εξαρτώνται από τις απαιτήσεις των πελατών. Ωστόσο, κάποια βασικά χαρακτηριστικά των οχημάτων είναι:

- Η αφετηρία των οχημάτων (depot) και αν δύναται να επιστρέψουν σε διαφορετικό σημείο από αυτό που ξεκίνησαν.
- Η χωρητικότητα του οχήματος (capacity) εκφρασμένη σε κάποια μονάδα π.χ. βάρος, όγκος, λίτρα, παλέτες που το όχημα μπορεί να φορτώσει.
- Αν τα οχήματα είναι χωρισμένα σε διάφορα τμήματα ανάλογα με τη χωρητικότητα ή το είδος του εμπορεύματος (ξηρό, ψυγείο ή προϊόντα κατάψυξης) που πρέπει να μεταφερθεί.
- Το σύνολο των διαδρομών που μπορούν να προσπελαστούν από το τρέχον όχημα.
- Ο διαθέσιμος εξοπλισμός που μπορεί να διαθέτει το όχημα για την διευκόλυνση των εργασιών φορτο-εκφόρτωσης (ανυψωτικό, παλετοφόρο).

- Το κόστος χρήσης που αναλογεί στον τύπο του οχήματος.

Οι οδηγοί εξίσου πρέπει να πληρούν μερικές απαιτήσεις που υπάρχουν όπως η διάρκεια και οι μέρες εργασίας, η διάρκεια διαλείμματος ακόμη και αν πραγματοποιούν υπερωρίες. Οι απαιτήσεις που ισχύουν για τους οδηγούς εφαρμόζονται στο μαθηματικό μοντέο και στα αντίστοιχα οχήματα που χρησιμοποιούν.

Γενικά ανάλογα με τα χαρακτηριστικά που έχουμε στο υπάρχον πρόβλημα τροποποιούνται και οι περιορισμοί που επηρεάζουν τη δρομολόγηση των οχημάτων. Ας εξετάσουμε μερικούς από αυτούς:

- Το όχημα δεν μπορεί να μεταφέρει περισσότερο φορτίο από τη χωρητικότητά του.
- Οι πελάτες έχουν προκαθορισμένα χρονικά παράθυρα κατά τα οποία πρέπει να γίνει η παράδοση ή η παραλαβή των αγαθών.
- Οι πελάτες μπορεί να έχουν συγκεκριμένες απαιτήσεις σχετικά με την παράδοση ή την παραλαβή των αγαθών.
- Ορισμένοι πελάτες μπορεί να πρέπει να εξυπηρετηθούν με συγκεκριμένη σειρά.
- Οι οδηγοί έχουν περιορισμένο χρόνο εργασίας και ανάπαυσης σύμφωνα με τους κανονισμούς ασφαλείας και τους νόμους περί οδικής ασφάλειας.
- Τα οχήματα μπορούν να μεταφέρουν περισσότερα από ένα προϊόντα, κάτι που αυξάνει την πολυπλοκότητα του προβλήματος, οπότε αυτοί οι περιορισμοί πρέπει να ληφθούν υπόψη κατά τον καθορισμό των διαδρομών.

Κάθε ένας από αυτούς τους περιορισμούς επηρεάζει τον τρόπο με τον οποίο σχεδιάζεται και λύνεται το πρόβλημα δρομολόγησης οχημάτων, καθώς απαιτείται η εύρεση διαδρομών που να τηρούν όλους αυτούς τους περιορισμούς. Οι στόχοι της επίλυσης του προβλήματος ορίζονται από την εκάστοτε επιχείρηση ανάλογα με τις απαιτήσεις των πελατών, σε ποιο αγοραστικό κοινό θέλει να απευθυνθεί καθώς και τι θέλει σε γενικό πλαίσιο να επιτύχει βάζοντας η ίδια τις προτεραιότητές της. Διατυπώνοντας τους στόχους της επίλυσης προβλημάτων δρομολόγησης οχημάτων:

- Ένας βασικός στόχος είναι η μείωση του συνολικού κόστους λειτουργίας του συστήματος δρομολόγησης οχημάτων, συμπεριλαμβανομένων των καυσίμων, του χρόνου οδήγησης, του κόστους συντήρησης και άλλων σχετικών λειτουργικών εξόδων.

- Ελαχιστοποίηση του συνολικά απαιτούμενου χρόνου για την παράδοση των αγαθών στους πελάτες.
- Μείωση του αριθμού οχημάτων που απαιτούνται για την κάλυψη της δεδομένης ζήτησης.
- Μέσω της εύρεσης της διαδρομής εκείνης που ελαχιστοποιεί την διανυθείσα απόσταση ή το χρόνο παράδοσης των προϊόντων δύναται να γίνει η εύρεση ενός συνόλου διαδρομών χαμηλού κόστους.

2.2.2 ΤΟ ΠΕΡΙΟΡΙΣΜΕΝΗΣ ΧΩΡΗΤΙΚΟΤΗΤΑΣ ΠΡΟΒΛΗΜΑ ΔΡΟΜΟΛΟΓΗΣΗΣ ΟΧΗΜΑΤΩΝ (CVRP) & ΜΟΝΤΕΛΟΠΟΙΗΣΗ

Το πρόβλημα δρομολόγησης οχημάτων γνωστό και ως πρόβλημα δρομολόγησης οχημάτων με περιορισμένη χωρητικότητα (Capacitated Vehicle Routing Problem), είναι μια πιο σύνθετη μορφή του προβλήματος του πλανόδιου πωλητή. Αντί να χρησιμοποιείται ένα μοναδικό όχημα για να επισκεφθεί όλους τους πελάτες, απαιτούνται περισσότερα οχήματα όταν ένα όχημα δεν επαρκεί. Σε αυτή την περίπτωση, σχεδιάζονται πολλές διαδρομές που ξεκινούν και τελειώνουν στο ίδιο σημείο και κάθε διαδρομή εξυπηρετεί μια ομάδα πελατών. Απαραίτητη προϋπόθεση για την επίλυση του προβλήματος αυτού είναι ότι κανένας πελάτης δεν πρέπει να εξυπηρετείται από περισσότερες από μία διαδρομές.

Όπως και στο πρόβλημα του πλανόδιου πωλητή, στόχος είναι να ελαχιστοποιηθεί το συνολικό κόστος των διαδρομών. Οι λόγοι που ένα μόνο όχημα δεν μπορεί να εξυπηρετήσει όλους τους πελάτες μπορεί να είναι πολλοί, όπως για παράδειγμα η συνολική ζήτηση των πελατών να υπερβαίνει τη χωρητικότητα του οχήματος, η ανάγκη να εξυπηρετηθούν οι πελάτες μέσα σε συγκεκριμένα χρονικά πλαίσια που δεν μπορούν να καλυφθούν από ένα μόνο όχημα, ή η απαίτηση για διαφορετικά προϊόντα που δεν μπορούν να μεταφερθούν μαζί, να αναγκάζουν τη χρήση επιπλέον οχημάτων.

Το βασικό πρόβλημα δρομολόγησης οχημάτων μπορεί να οριστεί με όρους θεωρίας γραφημάτων ως:

Έστω ότι έχουμε ένα πλήρη γράφημα $G = (V, A)$ όπου δίνεται $V = \{0, \dots, n\}$ το σύνολο των κόμβων έχοντας σαν κόμβο 0 το σημείο εκκίνησης μας, δηλαδή την αποθήκη, ενώ οι υπόλοιποι κόμβοι αντιστοιχούν στους πελάτες που πρέπει να εξυπηρετηθούν. Το κόστος του ταξιδιού που διασχίζει ένα όχημα κατά μήκος ενός τόξου (i, j) του γραφήματος, συσχετίζεται με ένα μη αρνητικό κόστος $c_{i,j}$. Όταν τα τόξα του δικτύου είναι μη προσανατολισμένα, δηλαδή $(i, j) = (j, i)$, το πρόβλημα ονομάζεται συμμετρικό περιορισμένης χωρητικότητας πρόβλημα δρομολόγησης οχημάτων, ενώ αν είναι προσανατολισμένα υποδεικνύοντας κατευθύνσεις, ονομάζεται μη συμμετρικό [9].

Στο πρόβλημα δρομολόγησης οχημάτων με περιορισμένη χωρητικότητα τίθεται ως στόχος η εύρεση K αριθμών κύκλων όπου $K = 1, \dots, n$ ο αριθμός των οχημάτων, με το ελάχιστο δυνατό κόστος. Το κόστος αυτό είναι το άθροισμα των αποστάσεων ή άλλων εξόδων για όλες τις διαδρομές. Τα βασικά χαρακτηριστικά αυτού του προβλήματος είναι τα εξής:

1. Κάθε διαδρομή πρέπει να ξεκινά και να τελειώνει στην αποθήκη.
2. Κάθε πελάτης πρέπει να εξυπηρετείται από μία μόνο διαδρομή.
3. Η συνολική ζήτηση των πελατών δεν πρέπει να ξεπερνά τη συνολική χωρητικότητα του οχήματος για κάθε διαδρομή.

Έστω ότι:

Έχουμε n πελάτες, όπου $i = 1$ είναι η αποθήκη και $j = 2, 3, \dots, n$ οι πελάτες που πρέπει να εξυπηρετηθούν. Έχουμε m στόλο οχημάτων διαθέσιμα με μέγιστη χωρητικότητα Q_k το κάθε ένα από αυτά με $k = 1, 2, \dots, m$. Με q_i συμβολίζεται η ζήτηση του πελάτη i .

Οι μεταβλητές απόφασης του προβλήματος είναι οι εξής:

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

Εξ. 2.6

$$y_{i,k} = \begin{cases} 1, & \text{εάν το όχημα } k \text{ εξυπηρετεί τον πελάτη } i \\ 0, & \text{αλλιώς} \end{cases}$$

Εξ. 2.7

Το βασικό πρόβλημα δρομολόγησης οχημάτων μπορεί να καθοριστεί ως εξής:

$$\min \sum_i \sum_j c_{i,j} \sum_k x_{i,j,k} \quad \text{Εξ. 2.8}$$

$$\sum_k y_{i,k} = \begin{cases} 1, & i=2,\dots,n \\ m, & i=1 \end{cases} \quad \text{Εξ. 2.9}$$

$$\sum_i q_i y_{ij} \leq Q_k, \quad k=1,\dots,m \quad \text{Εξ. 2.10}$$

$$\sum_j x_{i,j,k} = \sum_k x_{j,i,k} = y_{i,k}, \quad i=1,\dots,n \quad \text{Εξ. 2.11}$$

$$\sum_{i,j \in S} x_{i,j,k} \leq |S| - 1, \quad \text{για όλα τα } S \subseteq \{2,\dots,n\} \quad \text{Εξ. 2.12}$$

$$y_{i,k} \in \{0,1\} \quad \text{Εξ. 2.13}$$

$$x_{i,j,k} \in \{0,1\} \quad \text{Εξ. 2.14}$$

Η Εξ. 2.9 δείχνει ότι σε κάθε πελάτη ανατίθεται ένα μόνο όχημα το οποίο θα τον εξυπηρετήσει, εκτός από την αποθήκη που ξεκινάνε όλα τα οχήματα από εκεί. Η Εξ. 2.10 αναπαριστά το περιορισμό χωρητικότητας του οχήματος k . Η Εξ. 2.11 δείχνει ότι ένα όχημα το οποίο έχει επισκεφτεί ένα πελάτη πρέπει να φύγει και από αυτόν, ενώ τέλος οι εξισώσεις Εξ. 2.13 και Εξ. 2.14 αναφέρουν ότι οι μεταβλητές απόφασης του προβλήματος είναι δυαδικοί αριθμοί.

2.2.3 ΤΟ ΠΡΟΒΛΗΜΑ ΔΡΟΜΟΛΟΓΗΣΗΣ ΣΕ ΔΕΔΟΜΕΝΑ ΧΡΟΝΙΚΑ ΠΕΡΙΘΩΡΙΑ (VRPTW)

Έχοντας ήδη εφαρμόσει περιορισμούς που συσχετίζονται με τη χωρητικότητα των οχημάτων, η προσθήκη του χρονικού περιορισμού εξυπηρέτησης των πελατών και του μέγιστου χρόνου διαδρομής για τα οχήματα στο πρόβλημα, δημιουργεί μια πιο σύνθετη και ρεαλιστική παραλλαγή, γνωστή ως πρόβλημα δρομολόγησης οχημάτων με χρονικά περιθώρια (Vehicle Routing Problem with Time Windows - VRPTW). Συγκεκριμένα πρέπει να ληφθεί υπόψη το γεγονός ότι η εξυπηρέτηση του κάθε πελάτη οφείλει να πραγματοποιείται εντός κατάλληλων χρονικών πλαισίων, γνωστά ως χρονικά παράθυρα (time windows). Στο πρόβλημα αυτό, σε σχέση με το κλασσικό VRP, μπορούμε να έχουμε δεδομένα όπως:

- Ο χρόνος φόρτωσης
- Ο χρόνος αναχώρησης του οχήματος από την αποθήκη
- Ο χρόνος μετακίνησης από πελάτη σε πελάτη
- Ο χρόνος εκφόρτωσης
- Ο χρόνος εξυπηρέτησής τους καθώς και
- Τα καθορισμένα χρονικά πλαίσια που οφείλει να εξυπηρετηθεί ο κάθε πελάτης.

Υπάρχουν δύο κύρια είδη χρονικών παραθύρων που χρησιμοποιούνται: τα σκληρά χρονικά παράθυρα και τα χαλαρά χρονικά παράθυρα. Συγκεκριμένα, τα σκληρά χρονικά παράθυρα ορίζουν αυστηρά χρονικά όρια εντός των οποίων πρέπει να εξυπηρετηθεί ένας πελάτης που σημαίνει ότι η άφιξη του οχήματος νωρίτερα ή αργότερα από το προκαθορισμένο χρονικό πλαίσιο δεν είναι αποδεκτή. Από την άλλη τα χαλαρά χρονικά παράθυρα επιτρέπουν μεγαλύτερη ευελιξία, καθώς επιτρέπουν την εξυπηρέτηση πελατών εκτός των προκαθορισμένων χρονικών ορίων, αλλά με ένα επιπλέον κόστος [10].

Αυτά τα προβλήματα στοχεύουν στην εύρεση του συνόλου των διαδρομών που εκτελούνται, με τη κάθε διαδρομή να αρχίζει και τελειώνει στην αποθήκη, ενώ παράλληλα πρέπει να τηρούνται αυστηρά οι περιορισμοί χωρητικότητας και οι παραδόσεις να πραγματοποιούνται εντός συγκεκριμένων χρονικών παραθύρων.

2.2.4 ΤΟ ΠΡΟΒΛΗΜΑ ΠΑΡΑΛΑΒΗΣ ΚΑΙ ΠΑΡΑΔΟΣΗΣ (VRPPD)

Στο πρόβλημα παραλαβής και παράδοσης, τα οχήματα πρέπει όχι μόνο να παραδώσουν προϊόντα σε πελάτες αλλά και να παραλάβουν προϊόντα από κάποιες τοποθεσίες κατά τη διάρκεια της διαδρομής τους.

Κάθε πελάτης συνδέεται με δύο διαφορετικές ποσότητες προϊόντων:

- την ποσότητα που χρειάζεται να διανεμηθεί σε αυτόν και
- την ποσότητα που πρέπει να παραλάβει.

Αυτές οι ποσότητες αντιπροσωπεύουν τη ζήτηση του εκάστοτε πελάτη και πρέπει να εκτελούνται εντός του χωρικού και χρονικού πλαισίου που ορίζει κάθε ένας, προκειμένου να εξασφαλιστεί η ομαλή λειτουργία της διαδικασίας χωρίς υπερβολικές καθυστερήσεις ή πρόωρες παραλαβές. Ένας σημαντικός περιορισμός στο πρόβλημα δρομολόγησης οχημάτων με διανομή και παραλαβή προϊόντων είναι ότι η διανομή των προϊόντων σε κάθε πελάτη πρέπει να γίνεται πριν από την παραλαβή τους. Αυτό σημαίνει ότι η συνολική φόρτωση του οχήματος κατά την άφιξή του σε κάθε πελάτη i προκύπτει από την αρχική του φόρτωση στην αποθήκη, μείον το άθροισμα των προϊόντων που έχει παραδώσει σε προηγούμενους πελάτες πριν τον πελάτη i , συν το άθροισμα των προϊόντων που έχει παραλάβει από προηγούμενους πελάτες πριν τον πελάτη i .

$$Q_i = Q_0 - \sum D_j + \sum P_k \quad \text{Εξ. 2.15}$$

Ο κύριος στόχος του προβλήματος είναι η εύρεση της βέλτιστης διαδρομής για κάθε όχημα, η ελαχιστοποίηση του συνολικού κόστους και του χρόνου που απαιτούνται ώστε να ικανοποιηθούν όλες οι απαιτήσεις διανομής και παραλαβής, λαμβάνοντας υπόψη τις χωρητικότητες και τους περιορισμούς χρόνου των οχημάτων και των πελατών.

2.2.5 ΥΠΑΡΞΗ ΠΟΛΛΑΠΛΩΝ ΑΠΟΘΗΚΩΝ (MD-VRP)

Η κατηγορία των προβλημάτων όπου περιλαμβάνονται παραπάνω από μια αποθήκες, αποτελεί μια επέκταση του κλασικού προβλήματος δρομολόγησης οχημάτων που περιλαμβάνει πολλαπλές αποθήκες από όπου ξεκινούν και τερματίζουν τα οχήματα. Αυτό το πρόβλημα δημιουργήθηκε από επιχειρήσεις οι οποίες δύναται να χρησιμοποιούν περισσότερα από ένα κέντρα διανομής.

Στο Multi-Depot VRP, οι πελάτες εξυπηρετούνται από πολλά οχήματα, όπου το καθένα βρίσκεται σε μία από τις πολλές αποθήκες που είναι εγκατεστημένες σε διαφορετικά μέρη.

Υπάρχουν δυο βασικοί τρόποι επίλυσης του συγκεκριμένου προβλήματος:

1. Πρώτων, κάθε αποθήκη μπορεί και λειτουργεί ως ανεξάρτητο VRP με δικά της οχήματα και πελάτες που εξυπηρετεί. Έτσι, έχουμε ουσιαστικά πολλαπλά ανεξάρτητα προβλήματα VRP.
2. Κάθε όχημα ξεκινάει τη διαδρομή του από μία αποθήκη, τερματίζει σε μία διαφορετική αποθήκη και ενδέχεται να περάσει μέσω μιας τρίτης αποθήκης για εφοδιασμό ή για κάποια άλλη δραστηριότητα.

Επιπλέον, το MDVRP είναι NP-hard που σημαίνει ότι δεν είναι διαθέσιμος ένας αποτελεσματικός αλγόριθμος για την επίλυση του προβλήματος στη βέλτιστη δυνατή. Επομένως, η επίλυση του προβλήματος με έναν ακριβή αλγόριθμο είναι χρονοβόρα και υπολογιστικά δυσεπίλυτη και για την αντιμετώπιση αυτής της κατηγορίας προβλημάτων χρησιμοποιούνται ευρετικοί και μεθευρετικοί αλγόριθμοι [11], τους οποίους θα αναλύσουμε στο επόμενο κεφάλαιο.

2.2.6 ΤΟ ΠΡΟΒΛΗΜΑ ΔΡΟΜΟΛΟΓΗΣΗΣ ΟΧΗΜΑΤΩΝ ΜΕ BACKHAULS ΚΑΙ LINEHAULS

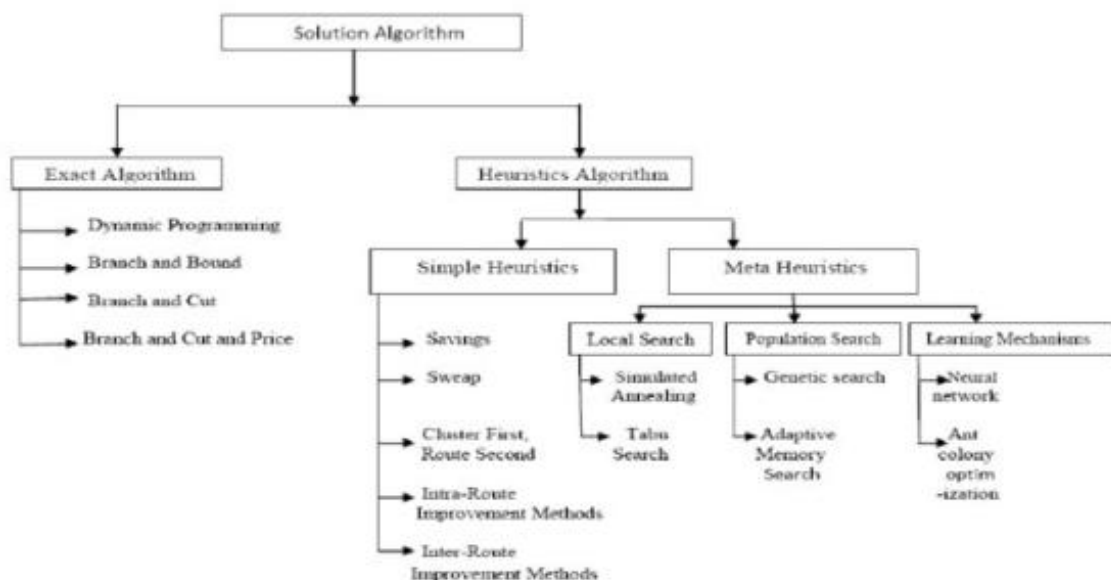
Σε αυτήν την κατηγορία προβλημάτων, οι πελάτες διακρίνονται σε δύο υποσύνολα. Το πρώτο υποσύνολο περιλαμβάνει τους πελάτες που απαιτούν τη διανομή προϊόντων (linehaul customers), ενώ το δεύτερο υποσύνολο περιλαμβάνει τους πελάτες που απαιτούν την παραλαβή προϊόντων από αυτούς (backhaul customers). Στο δεύτερο υποσύνολο ανήκουν επίσης οι πελάτες που επιθυμούν την επιστροφή προϊόντων.

Πέρα από τους κλασικούς περιορισμούς των προβλημάτων δρομολόγησης οχημάτων, υπάρχει ένας επιπλέον σημαντικός περιορισμός μεταξύ των δύο τύπων πελατών. Συγκεκριμένα, οι backhaul customers πρέπει να εξυπηρετούνται μετά τους linehaul customers. Αυτό οφείλεται στο γεγονός ότι τα οχήματα φορτώνονται προς τα πίσω και η αναδιάταξη των φορτίων στα σημεία παράδοσης δεν είναι εφικτή. Όλες οι παραδόσεις πρέπει να ολοκληρώνονται σε κάθε διαδρομή πριν από τις παραλαβές και οι ποσότητες που παραδίδονται και παραλαμβάνονται είναι προκαθορισμένες και γνωστές εκ των προτέρων. Ο

στόλος των οχημάτων είναι ομοιογενής, με κάθε όχημα να έχει σταθερή χωρητικότητα σε βάρος ή όγκο. Έτσι, μια εφικτή λύση στο πρόβλημα περιλαμβάνει ένα σύνολο διαδρομών όπου οι παραδόσεις για κάθε διαδρομή ολοκληρώνονται πριν από τις παραλαβές και η χωρητικότητα του οχήματος δεν υπερβαίνεται ούτε από τα σημεία linehaul ούτε από τα σημεία backhaul που έχουν εκχωρηθεί στη διαδρομή [12].

3 ΚΕΦΑΛΑΙΟ: ΑΛΓΟΡΙΘΜΟΙ ΕΠΙΛΥΣΗΣ ΠΡΟΒΛΗΜΑΤΩΝ ΣΥΝΔΙΑΣΤΙΚΗΣ ΒΕΛΤΙΣΤΟΠΟΙΗΣΗΣ

Τα προβλήματα συνδυαστικής βελτιστοποίησης μπορούν να λυθούν και να βελτιστοποιηθούν χρησιμοποιώντας δύο διαφορετικές προσεγγίσεις όπως θα παρουσιαστούν στη συνέχεια αυτής της ενότητας. Η πρώτη προσέγγιση εστιάζει στη χρήση πλήρων ή όπως αλλιώς καλούνται, ακριβών αλγορίθμων. Αυτοί οι αλγόριθμοι είναι ιδανικοί για προβλήματα με μικρότερες και λιγότερο περίπλοκες δομές, καθώς μπορούν να εξετάσουν και να εφαρμόσουν κάθε πιθανή επιλογή με ακρίβεια. Η δεύτερη προσέγγιση χρησιμοποιείται για μεγαλύτερα και πιο σύνθετα προβλήματα και συχνά οδηγεί σε σχεδόν βέλτιστες λύσεις. Χαρακτηρίζεται από τη χρήση κατά προσέγγιση αλγορίθμων, οι οποίοι είναι πιο γρήγοροι αλλά μπορεί να μην εγγυώνται την απόλυτη βέλτιστη λύση. Αυτή η δεύτερη προσέγγιση μπορεί να διακριθεί περαιτέρω σε ευρετικές μεθόδους, οι οποίες χρησιμοποιούν εμπειρικά κριτήρια για την επιλογή λύσεων και σε μετα-ευρετικές μεθόδους, οι οποίες επικεντρώνονται στη βελτίωση των λύσεων που παρήχθησαν από ευρετικούς αλγορίθμους.



Εικόνα 3.1: Αλγόριθμοι επίλυσης για παραλλαγές στο Vehicle Routing Problem [13].

3.1 ΑΚΡΙΒΕΙΣ ΑΛΓΟΡΙΘΜΟΙ

Οι ακριβείς ή πλήρεις αλγόριθμοι είναι εγγυημένο ότι βρίσκουν για κάθε παράδειγμα πεπερασμένου μεγέθους ενός προβλήματος συνδυαστικής βελτιστοποίησης μια βέλτιστη λύση σε περιορισμένο χρόνο. Αν και αυτοί οι αλγόριθμοι βρίσκουν τη βέλτιστη λύση, καθώς όμως χρειάζονται μεγάλο υπολογιστικό χρόνο στις περισσότερες περιπτώσεις που έχουμε προβλήματα μεγάλου αριθμού πελατών, οδηγούν σε μη πρακτικό υπολογιστικό φόρτο για πραγματικές εφαρμογές και καθιστούν τη χρήση τους αδύνατη. Η οικογένεια των ακριβών μεθόδων είναι αρκετά μεγάλη, αλλά οι πιο κοινές ακριβείς/πλήρες μέθοδοι για προβλήματα προγραμματισμού είναι οι αλγόριθμοι διακλάδωσης και δεσμεύσεως, ο προγραμματισμός μεικτού ακέραιου αριθμού και οι μέθοδοι αποσύνθεσης [14].

Αρχικά, προτάθηκαν ακριβείς αλγόριθμοι λόγω της υψηλής ποιότητας των λύσεων που παράγονται, αλλά στην περίπτωση προβλημάτων μεγάλης κλίμακας (περισσότεροι από 100 πελάτες), ο υπολογιστικός χρόνος που απαιτείται για έναν ακριβή αλγόριθμο για την εύρεση της βέλτιστης λύσης αυξάνεται τόσο πολύ που καταλήγει να μην είναι πρακτικό. Κατά συνέπεια, αναπτύχθηκαν ευρετικοί και μεταευρετικοί αλγόριθμοι, καθώς και οι δύο προσφέρουν καλύτερη ισορροπία μεταξύ ποιότητας λύσης και υπολογιστικού χρόνου.

3.2 ΕΥΡΕΤΙΚΟΙ ΑΛΓΟΡΙΘΜΟΙ

Η χρήση πλήρων/ακριβών μεθόδων για την επίλυση σύνθετων προβλημάτων συνδυαστικής βελτιστοποίησης οδηγεί συχνά σε μη ρεαλιστικούς υπολογιστικούς χρόνους. Αυτό το φαινόμενο έχει οδηγήσει τη πλειοψηφία των ερευνητών στην εύρεση μεθόδων επίλυσης για τέτοια προβλήματα. Οι ευρετικοί (heuristic) αλγόριθμοι μειώνουν σημαντικά τον χρόνο επίλυσης προβλημάτων και παρέχουν λύσεις υψηλής ποιότητας που είναι πολύ κοντά στη βέλτιστη λύση, αλλά δεν επιτυγχάνουν απόλυτη βελτιστοποίηση όπως οι ακριβείς αλγόριθμοι που αναλύθηκαν παραπάνω. Αυτό συμβαίνει καθώς ο εξεταζόμενος αλγόριθμος εκτελεί μια περιορισμένη αναζήτηση στο χώρο των πιθανών λύσεων, δεν μπορεί να

επιτευχθεί απόλυτη βελτιστοποίηση. Στις μεθόδους προσέγγισης, η εγγύηση της εύρεσης της βέλτιστης λύσης θυσιάζεται προκειμένου να επιτευχθεί μια σχεδόν βέλτιστη λύση σε εύλογο και πρακτικό υπολογιστικό χρόνο.

Οι ευρετικοί αλγόριθμοι μπορούν να χωριστούν σε διάφορες κατηγορίες, ανάλογα με την εφαρμογή τους και το πρόβλημα που προσπαθούν να επιλύσουν. Ορισμένοι από τους πιο γνωστούς είναι οι εξής:

1. Αλγόριθμοι Απληστίας (Greedy Algorithms): Οι αλγόριθμοι απληστίας λαμβάνουν την τοπικά βέλτιστη επιλογή σε κάθε βήμα, προσδοκώντας ότι η συνολική σειρά αποφάσεων θα οδηγήσει σε μια καλή γενική λύση για το πρόβλημα. Παράδειγμα περιλαμβάνουν ο αλγόριθμος επιλογής της καλύτερης προσέγγισης (best-first search) και ο αλγόριθμος του πλησιέστερου γείτονα (nearest neighbor).
2. Προσεγγιστικοί Αλγόριθμοι (Approximation Algorithms): Οι προσεγγιστικοί αλγόριθμοι παρέχουν μια γρήγορη λύση για προβλήματα που είναι δύσκολο να επιλυθούν με ακριβείς μεθόδους. Αυτοί οι αλγόριθμοι προσεγγίζουν τη βέλτιστη λύση με μια προσεγγιστική ακρίβεια, συχνά εφαρμόζοντας τεχνικές όπως γραμμικός προγραμματισμός ή προσεγγίζοντας βέλτιστες λύσεις με αποδεκτά σφάλματα.
3. Αλγόριθμοι Τοπικής Αναζήτησης (Local Search Algorithms): Οι αλγόριθμοι τοπικής αναζήτησης ερευνούν τον χώρο κατάστασης με στόχο να βρουν μια καλή λύση ή να πραγματοποιηθούν μερικές επαναλήψεις, ξεκινώντας από μια αρχική κατάσταση και μεταβαίνοντας σε καλύτερες καταστάσεις κατά το δυνατόν.

3.3 ΜΕΘΕΥΡΕΤΙΚΟΙ ΑΛΓΟΡΙΘΜΟΙ

Τις τελευταίες δεκαετίες, μια νέα οικογένεια προσεγγιστικών αλγορίθμων έχει δημιουργηθεί και έχει κυριαρχήσει στην έρευνα επίλυσης προβλημάτων συνδυαστικής βελτιστοποίησης. Αυτός ο νέος τύπος αλγορίθμου συνδυάζει διαδικασίες τοπικής αναζήτησης και υψηλότερου επιπέδου στρατηγικές για να δημιουργηθεί μια διαδικασία που είναι ικανή να ξεφύγει από κάποιο τοπικό ελάχιστο. Ο στόχος της νέας προσέγγισης είναι να

ερευνήσει αποτελεσματικά τον χώρο αναζήτησης με τη γνώση των λογικών και των αποτελεσμάτων κινήσεων, διευκολύνοντας έτσι τη διαφυγή από τις τοπικές βέλτιστες λύσεις. Αυτές οι μέθοδοι σήμερα καλούνται μετα-ευρετικές ή μεθευρετικές (meta-heuristic).

Οι μετα-ευρετικές μέθοδοι έχουν ένα πλεονέκτημα έναντι των απλούστερων ευρετικών από την άποψη της ευρωστίας της λύσης. Ωστόσο, είναι συνήθως πιο δύσκολο να εφαρμοστούν και να συντονιστούν, καθώς χρειάζονται ειδικές πληροφορίες σχετικά με το πρόβλημα που πρέπει να λυθεί για να εμφανιστούν καλά αποτελέσματα. Οι μεθευρετικοί αλγόριθμοι, όπως και οι ευρετικοί που αναφέρθηκαν προηγουμένως, παρέχουν λύσεις που δεν είναι ακριβείς, αλλά πολύ κοντά στη βέλτιστη λύση ενώ οι μεταευρετικοί χρησιμοποιούν ευρετικούς για να λύσουν τις υποδιεργασίες τους.

Οι μετα-ευρετικές μέθοδοι ανήκουν στο ευρύτερο πεδίο των ευρετικών μεθόδων και χρησιμοποιούνται ευρέως για την επίλυση προβλημάτων συνδυαστικής βελτιστοποίησης, εξαιτίας των περιορισμών και των προκλήσεων που προκύπτουν από την εφαρμογή ακριβών αλγορίθμων.

Ακολουθούν κάποια χαρακτηριστικά των μεθευρετικών:

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

Οι μεθευρετικοί αλγόριθμοι που χρησιμοποιούνται συχνά για την επίλυση προβλημάτων δρομολόγησης οχημάτων περιλαμβάνουν τους ακόλουθους:

- Simulated Annealing (SA):

Η προσομοίωση ανόπτησης είναι μια από τις απλούστερες και πιο γνωστές μετα-ευρετικές μεθόδους για την αντιμετώπιση δύσκολων προβλημάτων βελτιστοποίησης "μαύρου κουτιού", όπου η αντικειμενική συνάρτηση δεν δίνεται ρητά και μπορεί να αξιολογηθεί μόνο με ορισμένες ακριβές προσομοιώσεις σε υπολογιστή [15]. Περιλαμβάνει τη δημιουργία ενός αριθμητικού μοντέλου για την αναπαραγωγή δεδομένων. Προσαρμόζει επαναληπτικά το μοντέλο έως ότου ταιριάζει με τα δεδομένα, αποδέχοντας ή απορρίπτοντας αλλαγές με βάση μια κατανομή πιθανοτήτων

εμπνευσμένη από τη διαδικασία της ανόπτησης. Η μέθοδος αυτή είναι ιδιαίτερα χρήσιμη για δύσκολα προβλήματα που αφορούν δυναμικά δεδομένα, μαλακά δεδομένα μεγάλης κλίμακας, στατιστικές πολλαπλών σημείων, τοποθέτηση αντικειμένων ή ειδική συνέχεια ακραίων τιμών [16].

Στο TSP, ξεκινάμε με την απόκτηση μιας αρχικής λύσης, η οποία μπορεί να είναι οποιαδήποτε έγκυρη διαδρομή για τον πωλητή. Στη συνέχεια, επιλέγονται τυχαία δύο πόλεις εντός της διαδρομής και ανταλλάσσονται οι θέσεις τους. Η SA χρησιμοποιείται για τον υπολογισμό της πιθανότητας αποδοχής ή απόρριψης της νέας λύσης. Η διαδικασία αυτή επαναλαμβάνεται για έναν καθορισμένο αριθμό επαναλήψεων, μειώνοντας σταδιακά την αντικειμενική συνάρτηση. Καθ' όλη τη διάρκεια των επαναλήψεων, διατηρείται ένα αρχείο της καλύτερης συνολικής λύσης που βρίσκεται [17].

- Tabu Search (TS):

Το Tabu Search είναι μια μετα-ευρετική μέθοδος που καθοδηγεί μια τοπική διαδικασία αναζήτησης για να εξερευνήσει το χώρο λύσεων πέρα από την τοπική βελτιστοποίηση. Ένα από τα κύρια στοιχεία της Tabu Search είναι η χρήση προσαρμοστικής μνήμης, η οποία δημιουργεί μια πιο ευέλικτη συμπεριφορά αναζήτησης. Επομένως, οι στρατηγικές που βασίζονται στη μνήμη αποτελούν το σήμα κατατεθέν των προσεγγίσεων αναζήτησης tabu, οι οποίες βασίζονται στην αναζήτηση "αρχών ολοκλήρωσης", με τις οποίες συνδυάζονται κατάλληλα εναλλακτικές μορφές μνήμης με αποτελεσματικές στρατηγικές για την εκμετάλλευσή τους. Το Tabu list χρησιμοποιεί μια δομή μνήμης για να αποφύγει την επανάληψη και να ερευνήσει νέες περιοχές της λύσης, λειτουργώντας ουσιαστικά ως μηχανισμός αποθήκευσης προηγούμενων καταστάσεων ή κινήσεων που θεωρούνται "ταμπού" για κάποιο χρονικό διάστημα, αποτρέποντας την αναζήτηση από το να επιστρέψει σε αυτές.

Μια νέα διαπίστωση είναι ότι τέτοιες αρχές είναι μερικές φορές αρκετά ισχυρές ώστε να αποδίδουν αποτελεσματική συμπεριφορά επίλυσης προβλημάτων από μόνες τους, με αμελητέα εξάρτηση από τη μνήμη. Σε ένα ευρύ φάσμα προβληματικών ρυθμίσεων, ωστόσο, η στρατηγική χρήση της μνήμης μπορεί να επιφέρει δραματικές διαφορές στην ικανότητα επίλυσης προβλημάτων. Αμιγείς και υβριδικές προσεγγίσεις αναζήτησης Tabu

έχουν σημειώσει νέα ρεκόρ στην εύρεση καλύτερων λύσεων σε προβλήματα σχεδιασμού και προγραμματισμού παραγωγής, κατανομής πόρων, σχεδιασμού δικτύων, δρομολόγησης, χρηματοοικονομικής ανάλυσης, τηλεπικοινωνιών, σχεδιασμού χαρτοφυλακίου, διαχείρισης εφοδιαστικής αλυσίδας, μοντελοποίησης βάσει πρακτόρων, σχεδιασμού επιχειρηματικών διαδικασιών, πρόβλεψης, μηχανικής μάθησης, εξόρυξης δεδομένων, βιοϋπολογισμού, μοριακού σχεδιασμού, διαχείρισης δασών και σχεδιασμού πόρων, μεταξύ πολλών άλλων τομέων [18].

- Genetic Algorithms (GA):

Οι Γενετικοί Αλγόριθμοι είναι τεχνικές βελτιστοποίησης που μιμούνται τις φυσικές διαδικασίες της εξέλιξης και της επιλογής. Διατηρούν έναν πληθυσμό λύσεων, όπου κάθε λύση αναπαρίσταται ως χρωμόσωμα. Μέσω διαδικασιών όπως η αναπαραγωγή και η μετάλλαξη, παράγονται απόγονοι που κληρονομούν χαρακτηριστικά από τους γονείς τους. Η καταλληλότητα κάθε λύσης, συνήθως συνδεδεμένη με την απόδοση της λύσης στο πρόβλημα, καθορίζει την πιθανότητα επιβίωσης και αναπαραγωγής της. Στο πλαίσιο του προβλήματος δρομολόγησης οχημάτων, οι GA χρησιμοποιούν αναπαραστάσεις που καθορίζουν ποιο όχημα εξυπηρετεί κάθε πελάτη, και εξελίσσονται μέσω της βελτίωσης της καταλληλότητας των λύσεων. Η καταλληλότητα (fitness) κάθε λύσης συνδέεται με την τιμή της αντικειμενικής συνάρτησης, στην προκειμένη περίπτωση τη συνολική απόσταση που διανύθηκε και το επίπεδο οποιασδήποτε παραβίασης περιορισμών [19].

- Ant Colony Optimization (ACO):

Η βελτιστοποίηση αποικίας μυρμηγκιών είναι μια μεταερευνητική μέθοδος βασισμένη σε πληθυσμό που μπορεί να χρησιμοποιηθεί για την εύρεση προσεγγιστικών λύσεων σε δύσκολα προβλήματα βελτιστοποίησης. Μοντελοποιεί τη συμπεριφορά των μυρμηγκιών που αφήνουν φερομόνες για την εύρεση των συντομότερων διαδρομών [20]. Η βελτιστοποίηση αποικίας μυρμηγκιών αντλεί έμπνευση από τη φυσική συμπεριφορά των μυρμηγκιών στην εύρεση τροφής, όπως αποδείχθηκε πειραματικά από τους Beckers, Deneubourg και Goss (1992). Εισήχθη για πρώτη φορά για την επίλυση του προβλήματος του πλανόδιου πωλητή. Έκτοτε έχουν προταθεί πολλές υλοποιήσεις της ACO για διάφορα προβλήματα συνδυαστικής βελτιστοποίησης. Τα πραγματικά μυρμηγκία, όταν αναζητούν

τροφή, καταθέτουν φερομόνες στο έδαφος και η συλλογική χρήση αυτών των ιχνών τα βοηθά να εντοπίσουν και να ακολουθήσουν την συντομότερη διαδρομή μεταξύ της φωλιάς τους και μιας πηγής τροφής.

4 ΚΕΦΑΛΑΙΟ: ΜΕΛΕΤΗ ΠΕΡΙΠΤΩΣΗΣ

Στόχος της παρούσας διπλωματικής εργασίας ήταν η επίλυση του προβλήματος της δρομολόγησης οχημάτων με περιορισμένη χωρητικότητα. Συγκεκριμένα αφορά την μελέτη περίπτωσης για μια Εμπορική Εταιρία Διανομών με έδρα τα Χανιά. Μέσα από αρχείο πωλήσεων που καταγράφεται από ERP λογισμικό της εταιρίας, λαμβάνονται δεδομένα που αφορούν την τιμολόγηση που έχει γίνει σε ένα συγκεκριμένο εύρος ημερομηνιών.

Το πρόβλημα που μελετάται, αφορά τον καθορισμό των βέλτιστων διαδρομών που θα ακολουθηθούν σε έναν στόλο οχημάτων που ξεκινούν από μια κεντρική αποθήκη (depot) και εξυπηρετούν όλους τους πελάτες, με σκοπό την ελαχιστοποίηση του συνολικού κόστους. Το συγκεκριμένο πρόβλημα όπως θα οριστεί, αφορά μόνο την παράδοση προϊόντων από την αποθήκη προς τους πελάτες.

4.1 ΠΛΗΡΟΦΟΡΙΕΣ ΕΤΑΙΡΙΑΣ 3PL

Η εταιρία εξυπηρετεί συνεχώς όλο το πελατολόγιο της, το οποίο κατά βάση περιορίζεται στους Νομούς Χανίων και Ρεθύμνου. Η εταιρία εκτελεί απευθείας, τριγωνικές και τετραγωνικές πωλήσεις και πωλήσεις δια λογαριασμό τρίτων, καλύπτοντας όλες τις ετήσιες εμπορικές συμφωνίες που έχουν με τους διάφορους ομίλους αγορών. Το πελατολόγιο περιλαμβάνει περίπου 720 πελάτες. Αυτοί οι πελάτες καλύπτουν ένα ευρύ φάσμα καταστημάτων τροφίμων και συναφών υπηρεσιών, όπως:

- Χονδρεμπόριο
- Σούπερ μάρκετ
- Μίνι μάρκετ
- Περίπτερα
- Εποχιακά σημεία πώλησης
- Ξενοδοχεία

- Εστιατόρια
- Πλοία
- Δημόσιες υπηρεσίες

Το πελατολόγιο το οποίο καλείται να εξυπηρετήσει απαιτεί εξειδικευμένες λύσεις δρομολόγησης για να εξασφαλιστεί η έγκαιρη και αποτελεσματική παράδοση των προϊόντων σε κάθε πελάτη. Η εταιρεία διαθέτει έναν ιδιόκτητο στόλο 17 φορτηγών διαφορετικών τύπων και μεγεθών για τη διακίνηση των προϊόντων της.

Η εταιρεία διαθέτει δύο οχήματα για την εξυπηρέτηση των πελατών της εταιρία που προσφέρει σνακ, καθώς και δύο επιπλέον οχήματα για την εξυπηρέτηση της εταιρίας που έχει αναθέσει τη διανομή για τα αρτοσκευάσματα. Θεωρούμε ότι τα οχήματα του στόλου, για τις συγκεκριμένες εταιρίες που θα μελετηθούν δεν έχουν την ίδια πεπερασμένη χωρητικότητα. Συγκεκριμένα 2 από τα τέσσερα φορτηγά έχουν μέγιστη χωρητικότητα 15 παλέτες ενώ τα άλλα δυο μπορούν να χωρέσουν μέχρι και 6 παλέτες. Υπάρχει ένα επιπλέον φορτηγό το οποίο χρησιμεύει σε περίπτωση που είτε χρειαστεί να καλύψει περισσότερη ζήτηση, είτε για να αντικαταστήσει κάποιο άλλο. Στο πρόβλημά μας, τα φορτηγά της εταιρίας εξυπηρετούν το πελατολόγιο που αφορά το νομό Χανίων.

Η διαδικασία της δρομολόγησης στην εταιρία πραγματοποιείται εμπειρικά. Χωρίζοντας το νομό σε περιοχές όπως κέντρο, Ακρωτήρι, Σούδα, Τουριστική περιοχή, Αποκόρωνας, Καστέλι, Σφακιά και Παλαιόχωρα, στον κάθε οδηγό ανατίθεται ένα από αυτά τα δρομολόγια. Οι οδηγοί επιλέγουν τον τρόπο εκτέλεσης του δρομολογίου και εξυπηρέτησης των πελατών σύμφωνα με τις προσωπικές τους προτιμήσεις και εμπειρίες, πράγμα το οποίο φυσικά δεν φέρει τα βέλτιστα αποτελέσματα είτε από άποψη της άμεσης εξυπηρέτησης πελατών είτε λόγω της μη σωστής διαχείρισης του κόστους, όπως ο χρόνος, τα καύσιμα ή άλλοι πόροι.

Καθώς λοιπόν δεν διατίθεται στην εταιρία λογισμικό δρομολόγησης, θα πραγματοποιηθεί επίλυση του προβλήματος. Στόχος είναι η ελαχιστοποίηση της συνολικά διανυθείσας απόστασης και η εύρεση των βέλτιστων διαδρομών που καλείται να καλύψει το κάθε όχημα.

4.2 ΔΙΑΧΕΙΡΙΣΗ ΔΕΔΟΜΕΝΩΝ

Το πρώτο βήμα που πραγματοποιήθηκε στη τρέχουσα εργασία ήταν η διαχείριση δεδομένων από ένα αρχείο Excel το οποίο εξάγεται από εμπορικό λογισμικό της υπό μελέτη εταιρίας. Αυτή η διαδικασία αποτελεί την αφετηρία της ανάλυσής, καθώς επιτρέπει την σωστή αξιοποίηση και ανάλυση των δεδομένων, καθώς πρέπει να εφαρμοστούν κατάλληλες αλλαγές. Συγκεκριμένα λαμβάνεται αρχείο το οποίο περιέχει τις στήλες

- DistrCustCode: Κωδικός πελάτη στο εμπορικό λογισμικό
- CustomerName: Επωνυμία πελάτη
- DeliveryPlace: Τόπος αποστολής. Χρησιμοποιείται καθώς υπάρχουν περιπτώσεις πελατών που έχουν παραπάνω από ένα υποκαταστήματα
- InvoiceType: Τύπος τιμολόγησης. Επιλέγονται μόνο τιμολόγια – δελτία αποστολής, καθώς στη συγκεκριμένη στήλη βρίσκονται και Πιστωτικά Τιμολόγια Επιστροφών, Δελτία Ποσοτικών Παραλαβών και άλλα εμπορικά παραστατικά που έχουν εκδοθεί εντός της τρέχουσας ημέρας
- DistributorSKU: Κωδικός προϊόντος
- ProductName: Ονομασία προϊόντος
- QTY: Ποσότητα που έχει τιμολογηθεί για ένα συγκεκριμένο προϊόν
- Creation_Time: Ημερομηνία δημιουργίας του παραστατικού.

Δεδομένου ότι δεν υπάρχει προκαθορισμένος αριθμός τεμαχίων για το σύνολο όλων των προϊόντων που μπορούν να τοποθετηθούν σε μια παλέτα και λαμβάνοντας υπόψιν ότι, στην υπό εξέταση μελέτη, θα πραγματοποιηθεί δρομολόγηση σε προϊόντα που αφορούν σνακ και αρτοσκευάσματα τα οποία καταλαμβάνουν διαφορετική χωρητικότητα ο κάθε κωδικός σε μια παλέτα, πρέπει να βρεθεί το ποσοστό του χώρου που καταλαμβάνει το κάθε προϊόν αλλά και το σύνολο των προϊόντων που αντιστοιχούν στη ζήτηση του κάθε πελάτη. Έτσι, στο τρέχον αρχείο προστίθενται 2 νέες στήλες:

- PC/PAL: το σύνολο των τεμαχίων του προϊόντος με μοναδικό ProductName, που χωράνε σε μια παλέτα (δίδεται από τις εταιρίες σνακ και αρτοσκευασμάτων)

- QTY/PAL: το ποσοστό του χώρου που καταλαμβάνουν τα τεμάχια του ProductName που έχουν τιμολογηθεί για τον κάθε πελάτη σε μια παλέτα. Υπολογίζεται διαιρώντας τα κελία της στήλης QTY με τα PC/PAL.

Το CVRP δημιουργεί μια διάσταση χρησιμοποιώντας τη μέθοδο `AddDimensionWithVehicleCapacity`, η οποία λαμβάνει μια σειρά από χωρητικότητες, με μία καταχώρηση για κάθε όχημα. Αξίζει λοιπόν να σημειωθεί ότι, καθώς οι ποσότητες QTY/PAL είναι δεκαδικοί αριθμοί, πρέπει να μετατραπούν σε ακέραιες τιμές. Επειδή οι ποσότητες που υπολογίζονται είναι πολύ μικρές (της τάξης 0,0001), πολλαπλασιάζουμε και στρογγυλοποιούμε τη QTY/PAL με έναν μεγάλο αριθμό, π.χ. 10.000. Ο ίδιος πολλαπλασιασμός θα γίνει και στις χωρητικότητες των οχημάτων.

Καθώς υπάρχουν προϊόντα, συγκεκριμένα πακέτα, εντός του τρέχοντος αρχείου που τιμολογούνται με ποσότητα 1, αλλά στην πραγματικότητα η ποσότητα αυτή είναι διαφορετική, πρέπει να γίνει η αντίστοιχη ενημέρωση των σωστών ποσοτήτων. Επειδή όμως υπάρχει η πιθανότητα να έχουν τιμολογηθεί πάνω από ένα δωροπακέτα μέσα στην ημέρα, πολλαπλασιάζουμε την ποσότητα τιμολόγησης με τη σωστή ποσότητα των προϊόντων. Η αλλαγή της ποσότητας στα δωροπακέτα είναι αναγκαία για την επακριβή αντικατάσταση των δεδομένων προκειμένου να ληφθούν καλύτερα αποτελέσματα. Η διόρθωση αυτή γίνεται με χρήση του ακόλουθου κώδικα, όπως φαίνεται στην Εικόνα 4.1:

```
prb = [
    (data['ProductName'] == 'ΣΟΚ ΠΑΥ ΥΓΕΙΑ 8.4KG 84TMX ΕΠΑ ΣΤΝΤ 4312616'),
    (data['ProductName'] == '7DAYS MINI CROIS PROMO 30TMX PRB_10000066'),
    (data['ProductName'] == 'CHIPS & TSIPERS STAND 36TMX PRB_10000060'),
    (data['ProductName'] == 'ΣΟΚ LAC ΤΑΜΠ LIDL 9.446KG 104T 2020'),
    (data['ProductName'] == 'LACTA LIGHT PROMOBX PRB_10000045'),
    (data['ProductName'] == 'PAVLIDES LIGHT PROMOBX PRB_10000043'),
    (data['ProductName'] == 'OREO PROMO STAND 120TMX PRB_10000065'),
    (data['ProductName'] == 'MOLTO ΚΡΣΝ PROMO 54TMX PRB_10000064')
]

#update σε ποσότητα που θέλω
quantities = [84, 30, 36, 104, 27, 14, 120, 54]

#αντιστοιχισιη του καθε ProductName με το factor του
#η συναρτησιη zip συνδιαζει στοιχεια απο δυο πινακες
for product, quantity in zip(prb, quantities):
    data.loc[product, 'QTY'] *= quantity
```

Εικόνα 4.1: Αλλαγή ειδικών προϊόντων που ανήκουν στη λίστα ProductName σε συγκεκριμένες ποσότητες.

Έπειτα γίνεται αποθήκευση των αλλαγών σε καινούριο αρχείο, το οποίο στη συνέχεια θα επεξεργαστεί ώστε να λάβουμε ένα συνολικό άθροισμα ανά πελάτη, ανά υποκατάστημα και ανά ημέρα (Εικόνα 4.2).

```
data = pd.read_excel(file_path, sheet_name='Sheet1')
sum_data = data.groupby(['CustomerName', 'DeliveryPlace', 'Creation_Time'])['QTY/PAL'].sum().reset_index()
sum_data.rename(columns={'QTY/PAL': 'Total_QTY/PAL'}, inplace=True)
```

Εικόνα 4.2: Υπολογισμός συνολικής ζήτησης ανά πελάτη, ανά υποκατάστημα ανά ημέρα.

Επόμενο βήμα ήταν η δημιουργία ενός αρχείου το οποίο περιέχει μόνο δεδομένα για πελάτες των οποίων μπορούν να βρεθούν οι συντεταγμένες τους. Μέσω της συνάρτησης `.merge`, η οποία συγκρίνει τα δεδομένα από το αρχείο συντεταγμένων με το αρχείο που δημιουργήθηκε στο προηγούμενο βήμα, επιστρέφεται ένα καινούριο αρχείο το οποίο περιλαμβάνει μόνο όσους πελάτες έχουν κοινά στοιχεία μεταξύ των δυο αρχείων. Το νέο αυτό αρχείο θα περιλαμβάνει τις στήλες:

1. CustomerName
2. DeliveryPlace
3. Creation_Time
4. Total_QTY/PAL
5. Latitude
6. Longitude

```
coordinates_file = 'C:/Users/lpero/OneDrive/Υπολογιστής/sellout/CUST.xlsx'
df_customers = pd.read_excel(sum_file_path, sheet_name='Sheet1')
df_reference = pd.read_excel(coordinates_file, sheet_name='Sheet1')

df_merged = pd.merge(df_customers, df_reference, on=['CustomerName', 'DeliveryPlace'], how='inner')

#τι στοιχεία θα περιλαμβάνει το τελικο αρχείο
final_df = df_merged[['CustomerName', 'DeliveryPlace', 'Creation_Time', 'Total_QTY/PAL', 'Latitude', 'Longitude']]
```

Εικόνα 4.3: Δημιουργία νέου αρχείου βάση πελατών που υπάρχουν διαθέσιμες οι συντεταγμένες τους.

Για να γίνει πιο εύκολη η σύγκριση των αποτελεσμάτων στον κύριο κώδικα μεταξύ των διαδρομών που εκτελεί κάθε όχημα, εισάγουμε μια νέα στήλη η οποία θα δίνει σε κάθε πελάτη, για κάθε υποκατάστημα που κατέχει, ένα μοναδικό CustomerID στον καθένα. Η δημιουργία αυτή της στήλης, φαίνεται στο ακόλουθο μέρος του κώδικα της Εικόνας 4.4.

```
# Create a combined column for CustomerName and DeliveryPlace
df['Customer_Delivery'] = df['CustomerName'] + ' - ' + df['DeliveryPlace']

#Δημιουργία νέας στήλης με CustomerID για κάθε συνδυασμό των CustomerName και DeliveryPlace
df['CustomerID'] = df['Customer_Delivery'].astype('category').cat.codes + 1

#Drop οτι δεν χρειάζεται να κρατήσει
df = df.drop(columns=['Customer_Delivery'])

df.insert(0, 'CustomerID', df.pop('CustomerID'))
```

Εικόνα 4.4: Εισαγωγή νέας στήλης, όπου ο πελάτης χαρακτηρίζεται από μοναδικό CustomerID.

Τέλος, γίνεται αποθήκευση του τελικού αρχείου Excel με όνομα output_file , το οποίο περιλαμβάνει τις στήλες

- CustomerID
- CustomerName
- DeliveryPlace
- Creation_Time
- Total_QTY/PAL
- Latitude
- Longitude

Το παρόν αρχείο είναι αυτό που θα ενσωματωθεί στον κύριο κώδικα προκειμένου να επιλυθεί το τρέχον πρόβλημα που αφορά τον καθορισμό των βέλτιστων διαδρομών που θα ακολουθήσουν τα οχήματα.

4.3 ΜΟΝΤΕΛΟΠΟΙΗΣΗ ΠΡΟΒΛΗΜΑΤΟΣ

Μεταβλητές απόφασης:

$x_{ijk} = 1$, αν το φορτηγό k εξυπηρετήσει τον πελάτη j μετά τον i , Δυαδική,

$x_{ijk} = 0$, αν όχι

Οι περιορισμοί που διέπουν το πρόβλημα είναι οι εξής:

1^{ος} περιορισμός: Κάθε πελάτης πρέπει να εξυπηρετηθεί ακριβώς μια φορά και από ένα μόνο φορτηγό

$$\sum_{j=0}^N \sum_{k=1}^K x_{ijk} = 1, \forall i \in \{1, \dots, N\}$$

Εξ. 4.1

$$\sum_{i=0}^N \sum_{k=1}^K x_{ijk} = 1, \forall j \in \{1, \dots, N\}$$

Εξ. 4.2

Ο περιορισμός δηλώνει ουσιαστικά ότι για κάθε πελάτη i , το άθροισμα των μεταβλητών x_{ijk} για όλους τους πιθανούς προορισμούς j (συμπεριλαμβανομένης της αποθήκης) και για όλα τα φορτηγά k πρέπει να είναι ίσο με 1.

2ος περιορισμός: Δεν μπορεί να υπάρχει διαδρομή όταν $i = j$. Συνεπώς για κάθε φορτηγό δεν μπορεί να υπάρχει διαδρομή από τον πελάτη i ξανά πίσω σε αυτόν, καθώς θα πρέπει υποχρεωτικά να αναχωρεί από αυτόν.

$$\sum_{j=0}^N x_{ijk} = \sum_{j=0}^N x_{jik}, \forall i \in \{1, \dots, N\}, \forall k$$

Εξ. 4.3

3ος περιορισμός: Κάθε φορτηγό εκτελεί το πολύ ένα δρομολόγιο ξεκινώντας και καταλήγοντας στην αποθήκη.

$$\sum_{j=1}^N x_{0jk} + \sum_{i=1}^N x_{i0k} \leq 1, \forall k$$

Εξ. 4.4

4ος περιορισμός: Η χωρητικότητα του οχήματος k μπορεί να καλύψει το σύνολο των ζητήσεων των πελατών που έχουν ανατεθεί στο δρομολόγιο του. Συνεπώς, το σύνολο των παραδόσεων που γίνονται σε κάθε διαδρομή δεν πρέπει να υπερβαίνει τη χωρητικότητα του οχήματος.

$$\sum_{i=0}^N \sum_{j=0}^N x_{ijk} * d_i \leq Q_k, \forall k$$

Εξ. 4.5

5ος περιορισμός: Ο κώδικας έχει οριστεί να εκτελείται για ένα συγκεκριμένο μέγιστο χρονικό διάστημα κάποιων δευτερολέπτων (`search_parameters.time_limit.FromSeconds()`).

6ος περιορισμός: Κάθε όχημα πρέπει να μεταφέρει ένα ελάχιστο φορτίο. Αυτός ο περιορισμός είναι απαραίτητος προκειμένου να εξαλειφθούν διαδρομές που δεν χρησιμοποιείται κάποιο από τα τέσσερα διαθέσιμα οχήματα.

$$\sum_{i=0}^N \sum_{j=0}^N x_{ijk} * d_i \geq Min_{Load_k}, \forall k$$

Εξ. 4.6

Όπου Min_{Load_k} το ελάχιστο φορτίο που πρέπει να μεταφέρει το φορτηγό k.

7ος περιορισμός: Κάθε όχημα πρέπει να εξυπηρετήσει έναν ελάχιστο αριθμό πελατών. Και αυτός ο περιορισμός, όπως και ο προηγούμενος, είναι απαραίτητος προκειμένου να εξαλειφθούν διαδρομές που δεν χρησιμοποιείται κάποιο από τα τέσσερα διαθέσιμα οχήματα.

$$\sum_{i=0}^N \sum_{j=0}^N x_{ijk} \geq Min_{Cust_k}, \forall k$$

Εξ. 4.7

Όπου Min_{Cust_k} είναι ο ελάχιστος αριθμός πελατών που πρέπει να εξυπηρετήσει το φορτηγό k.

8ος περιορισμός: Υπάρχουν πελάτες που ανήκουν στην κατηγορία "big market". Σε αυτούς ανήκουν πελάτες όπως χονδρέμποροι και Super-Market όπου απαιτούνται μεγάλες παραδόσεις και εξυπηρετούνται μονάχα από τα πρώτα δυο οχήματα.

Αν S_i είναι μια σταθερά που δείχνει αν ο πελάτης i ανήκει στην κατηγορία "big market" (με $S_i = 1$ αν ανήκει και $S_i = 0$ αν δεν ανήκει), τότε ο περιορισμός μπορεί να διατυπωθεί ως εξής:

$$\sum_{i=0}^N \sum_{j=0}^N x_{ijk} * S_i = 0, \forall k > 2, \forall i \exists S_i = 1$$

Εξ. 4.8

9ος περιορισμός: Για κάθε όχημα έχει οριστεί ένας μέγιστος αριθμός πελατών που δύναται να εξυπηρετήσει. Για τα πρώτα δυο οχήματα, τα οχήματα δηλαδή που έχουν οριστεί να εξυπηρετούν πελάτες της μεγάλης αγοράς, το όριο είναι 20 πελάτες, ενώ για τα άλλα ανέρχεται σε 40 πελάτες.

Αν Max_{Cust_k} είναι ο μέγιστος αριθμός πελατών που μπορεί να εξυπηρετήσει το όχημα k και είναι:

$$Max_{Cust_k} = 20, \text{ για } k = 1 \text{ και } k = 2$$

$$Max_{Cust_k} = 40 \text{ για } k > 2$$

τότε ο περιορισμός μπορεί να διατυπωθεί ως εξής:

$$\sum_{i=0}^N \sum_{j=0}^N x_{ijk} \leq Max_{Cust_k}, \forall k$$

Εξ. 4.9

Ένας ακόμη περιορισμός που θα μπορούσε να υπάρχει είναι ότι η συνολική διανειθήσα απόσταση που εκτελεί ένα όχημα σε κάθε διαδρομή δεν πρέπει να ξεπερνάει ένα προκαθορισμένο μέγιστο μήκος διαδρομής, λαμβάνοντας υπόψη και τον χρόνο εξυπηρέτησης κάθε πελάτη. Στο συγκεκριμένο πρόβλημα, δεν θα ληφθεί υπόψιν σχετικός περιορισμός ως προς το μέγιστο μήκος διαδρομής.

4.4 ΕΠΙΛΥΣΗ ΠΡΟΒΛΗΜΑΤΟΣ

Προηγουμένως καθορίστηκε η συνολική ποσότητα προϊόντων που τιμολογείται καθημερινά για κάθε πελάτη. Ωστόσο, δεδομένου ότι η χωρητικότητα των φορτηγών μετράτε

σε παλέτες, ήταν απαραίτητο να γίνει η αντίστοιχη μετατροπή, ώστε τα δεδομένα της ζήτησης να είναι ακριβή και σωστά.

Έτσι, και εφόσον έχει πραγματοποιηθεί η αντίστοιχη μετατροπή που αναφέρεται παραπάνω, έχουμε πλέον τα κατάλληλα δεδομένα που απαιτούνται για την επίλυση του προβλήματος:

- Συντεταγμένες αποθήκης και πελατών, οι οποίες θα χρησιμοποιηθούν για την κατασκευή του πίνακα αποστάσεων που θα δημιουργηθεί στη συνέχεια.
- Τη ζήτηση των πελατών σε ημερήσια βάση όπως αυτή δημιουργήθηκε προηγουμένως.
- Τον αριθμό του στόλου οχημάτων καθώς και την διαθέσιμη χωρητικότητά τους ανά όχημα.
- Τους κατάλληλους περιορισμούς που διέπουν το πρόβλημα.

Η επίλυση του προβλήματος έγινε αρχικά μέσω βιβλιοθηκών του ελεύθερου λογισμικού OR-Tools και έπειτα με τη χρήση του αλγορίθμου της πλησιέστερης εισαγωγής τα οποία θα αναλυθούν στη συνέχεια.

5 ΚΕΦΑΛΑΙΟ: ΕΠΙΛΥΣΗ VRP ΜΕ ΧΡΗΣΗ GOOGLE OR-TOOLS

Το OR-Tools αποτελεί μια σουίτα λογισμικού ανοιχτού κώδικα για προβλήματα συνδυαστικής βελτιστοποίησης όπου επιδιώκεται η εύρεση της καλύτερης λύσης σε ένα πρόβλημα από ένα πολύ μεγάλο πλήθος πιθανών λύσεων. Τα εργαλεία αυτά έχουν την ικανότητα επίλυσης των πιο δύσκολων προβλημάτων όπως αυτά που αφορούν την δρομολόγηση οχημάτων, τις ροές, τον ακέραιο και γραμμικό προγραμματισμό και τον προγραμματισμό περιορισμών.

Το OR-Tools περιλαμβάνει λύσεις επίλυσης για:

- Constraint Programming

Περιλαμβάνει ένα σύνολο τεχνικών για την εύρεση εφικτών λύσεων σε ένα πρόβλημα που εκφράζεται με τη μορφή περιορισμών, ενώ χρησιμοποιείται για την εύρεση λύσεων σε προβλήματα προγραμματισμού και συνδυαστικής βελτιστοποίησης. Βασίζεται κυρίως σε βασικές αρχές της επιστήμης των υπολογιστών, όπως ο λογικός προγραμματισμός και η θεωρία γραφημάτων [21].

- Γραμμικός και Μικτός Ακέραιος Προγραμματισμός

Ο γραμμικός βελτιστοποιητής Glop (Google linear optimization solver) βρίσκει τη βέλτιστη τιμή μιας γραμμικής αντικειμενικής συνάρτησης, δεδομένου ενός συνόλου γραμμικών ανισοτήτων ως περιορισμών. Το Glop και το λογισμικό προγραμματισμού μεικτού ακέραιου αριθμού SCIP είναι επίσης διαθέσιμα μέσω της Υπηρεσίας βελτιστοποίησης σεναρίων Google Apps [22].

- Δρομολόγηση Οχημάτων

Μια εξειδικευμένη βιβλιοθήκη για τον εντοπισμό των καλύτερων διαδρομών οχημάτων με δεδομένους περιορισμούς.

- Αλγόριθμοι γραφημάτων

Αλγόριθμοι που στοχεύουν στην εύρεση συντομότερων διαδρομών σε γραφήματα, ροές ελάχιστου κόστους, μέγιστες ροές και γραμμικές εκχωρήσεις αθροίσματος.

Σε πολλούς από τους solver του, το OR-Tools δέχεται μόνο ακέραιους αριθμούς. Για αυτό και στην τρέχουσα μελέτη τα δεδομένα που θα χρησιμοποιηθούν θα υποστούν τις κατάλληλες μετατροπές.

Στο Or-Tools η στρατηγική πρώτης λύσης είναι η μέθοδος που χρησιμοποιεί ο solver για να βρει μια αρχική λύση.

Τα προϊόντα της Πλατφόρμας Χαρτών Google προστατεύονται από μη εξουσιοδοτημένη χρήση περιορίζοντας τις API calls σε εκείνες που παρέχουν κατάλληλα διαπιστευτήρια ελέγχου ταυτότητας. Αυτά τα διαπιστευτήρια έχουν τη μορφή κλειδιού API - μια μοναδική αλφαριθμητική συμβολοσειρά που συσχετίζει τον λογαριασμό χρέωσης Google με το έργο και με το συγκεκριμένο API ή SDK.

5.1 ΕΠΙΛΟΓΕΣ ΓΙΑ ΤΗΝ ΕΠΙΛΥΣΗ ΔΡΟΜΟΛΟΓΗΣΗΣ

Το OR Tools είναι ένα εργαλείο ανοιχτού κώδικα, που αναπτύχθηκε από την Google, για την επίλυση προβλημάτων βελτιστοποίησης στη δρομολόγηση οχημάτων, τις ροές δικτύου, τον ακέραιο και γραμμικό προγραμματισμό και τον προγραμματισμό περιορισμών, όπως περιεγράφηκε προηγουμένως.

Η επίλυση που δίνεται από το OR-Tools για προβλήματα δρομολόγησης οχημάτων περιλαμβάνει 12 διαφορετικές στρατηγικές βελτιστοποίησης, όπως CHEAPEST_ARC, BEST_INSERTION, GLOBAL_CHEAPEST_ARC κ.λπ. καθεμία από τις οποίες μπορεί να ενσωματωθεί σε 6 διαφορετικά ευρετικά, όπως:

- AUTOMATIC: Επιτρέπει στον solver να επιλέξει τον μεθευρικό αλγόριθμο.
- GREEDY_DESCENT: Αποδέχεται τη βελτίωση (μείωση κόστους) των γειτόνων τοπικής αναζήτησης έως ότου επιτευχθεί ένα τοπικό ελάχιστο.

- GUIDED_LOCAL_SEARCH: Χρησιμοποιεί καθοδηγούμενη τοπική αναζήτηση για να ξεφύγει από τα τοπικά ελάχιστα. Αυτή είναι γενικά η πιο αποδοτική μεταερευνητική για τη δρομολόγηση οχημάτων.
- SIMULATED_ANNEALING: Χρησιμοποιεί προσομοιωμένη ανόπτηση για να ξεφύγει από τα τοπικά ελάχιστα.
- TABU_SEARCH: Χρησιμοποιεί αναζήτηση tabu για να ξεφύγει από τα τοπικά ελάχιστα.
- GENERIC_TABU_SEARCH: Χρησιμοποιεί αναζήτηση tabu στην αντικειμενική τιμή της λύσης για να ξεφύγει από τα τοπικά ελάχιστα.

Η στρατηγική πρώτης λύσης είναι η μέθοδος που χρησιμοποιεί ο επιλυτής για να βρει μια αρχική λύση. Πιο συγκεκριμένα, χρησιμοποιεί τις ακόλουθες στρατηγικές:

- AUTOMATIC: Ο solver αποφασίζει αυτόματα ποια στρατηγική θα χρησιμοποιήσει ανάλογα με το μοντέλο του προβλήματος.
- PATH_CHEAPEST_ARC: Ξεκινώντας από έναν κόμβο "έναρξης" διαδρομής, συνδέεται με τον κόμβο που παράγει το φθηνότερο τμήμα διαδρομής και στη συνέχεια, επεκτείνει τη διαδρομή επαναλαμβάνοντας τον τελευταίο κόμβο που προστέθηκε στη διαδρομή.
- PATH_MOST_CONSTRAINED_ARC: Αυτή είναι παρόμοια με τη στρατηγική PATH_CHEAPEST_ARC, αλλά σε αυτήν θα ευνοήσει το πιο περιορισμένο τόξο πρώτα χρησιμοποιώντας μια συνάρτηση επιλογής που βασίζεται στη σύγκριση.
- SAVINGS : Αυτή η στρατηγική βασίζεται στον αλγόριθμο που δίνεται από τους Clarke, G. & Wright.
- ΧΡΙΣΤΟΦΙΔΗΣ: Αυτή η στρατηγική λειτουργεί με βάση τον αλγόριθμο που έδωσε ο Νίκος Χριστοφίδης στον οποίο επεκτείνει τη διαδρομή για το μοντέλο δρομολόγησης οχημάτων έως ότου δεν μπορούν να εισαχθούν περαιτέρω κόμβοι.
- ALL_UNPERFORMED: Σε αυτήν τη στρατηγική, όλοι οι κόμβοι γίνονται ανενεργοί και η λύση βρίσκεται μόνο όταν υπάρχουν προαιρετικοί κόμβοι.
- BEST_INSERTION: Σε αυτή τη στρατηγική μια λύση δημιουργείται επαναληπτικά με την εισαγωγή του φθηνότερου κόμβου, στη φθηνότερη θέση του όπου η τιμή της συνάρτησης εισαγωγής αντιστοιχεί στο συνολικό εκτιμώμενο κόστος. Αυτή η στρατηγική λειτουργεί μόνο με προαιρετικούς κόμβους.

- **PARALLEL_CHEAPEST_INSERTION:** Είναι παρόμοια με τη στρατηγική **BEST_INSERTION**. Η μόνη διαφορά είναι η συνάρτηση εισαγωγής που αντιστοιχεί στη συνάρτηση κόστους τόξου.
- **LOCAL_CHEAPEST_INSERTION:** Σε αυτή τη στρατηγική, δημιουργείται μια λύση εισάγοντας κάθε κόμβο επαναληπτικά στη φθηνότερη θέση του, όπου η τιμή της συνάρτησης εισαγωγής αντιστοιχεί στη συνάρτηση κόστους τόξου. Είναι ταχύτερη από την **PARALLEL_CHEAPEST_INSERTION** και διαφέρει από αυτήν ως προς τον κόμβο που επιλέγεται για εισαγωγή καθώς εδώ οι κόμβοι εξετάζονται με τη σειρά δημιουργίας τους.
- **GLOBAL_CHEAPEST_ARC:** Σε αυτή τη στρατηγική δύο κόμβοι συνδέονται επαναληπτικά με αποτέλεσμα να παράγουν το φθηνότερο τμήμα διαδρομής.
- **LOCAL_CHEAPEST_ARC:** Σε αυτή τη στρατηγική επιλέγεται ο πρώτος κόμβος με μη δεσμευμένο διάδοχο και συνδέεται επαναληπτικά με κόμβους που παράγουν το φθηνότερο τμήμα διαδρομής.
- **FIRST_UNBOUND_MIN_VALUE:** Σε αυτήν τη στρατηγική, ο πρώτος διαθέσιμος κόμβος επιλέγεται και συνδέεται επαναληπτικά με τον πρώτο μη δεσμευμένο διάδοχο

Ο κώδικας που χρησιμοποιείται στη παρούσα εργασία συγκεκριμένα ορίζει την στρατηγική πρώτης λύσης σε **PATH_CHEAPEST_ARC**, η οποία δημιουργεί μια αρχική διαδρομή για τον λύτη προσθέτοντας επανειλημμένα άκρα με το μικρότερο βάρος που δεν οδηγούν σε έναν κόμβο που έχει επισκεφτεί προηγουμένως (εκτός από την αποθήκη).

Έπειτα εφαρμόζει και τον μεθευρετικό αλγόριθμο **Guided Local Search**, που είναι μια τεχνική που βελτιώνει την αρχική λύση μέσω τοπικής αναζήτησης. Προσπαθεί να βελτιώσει την αρχική διαδρομή αποφεύγοντας τοπικά βέλτιστα σημεία, ώστε να μειωθεί περαιτέρω η συνολική απόσταση ή το κόστος της διαδρομής.

5.1.1 PATH CHEAPEST ARC

Η διαδικασία **PATH CHEAPEST ARC** είναι μια μέθοδος για την επίλυση προβλημάτων δρομολόγησης που στοχεύει στη δημιουργία μιας αρχικής λύσης με βάση τις ελάχιστες αποστάσεις.

Ακολουθεί μια περιγραφή της διαδικασίας:

Ξεκινώντας από ένα σημείο εκκίνησης (depot) επιλέγεται η επόμενη τοποθεσία με βάση το ελάχιστο κόστος απόστασης. Για κάθε επόμενο βήμα, υπολογίζονται οι αποστάσεις από τον τρέχοντα πελάτη σε όλους τους μη επισκεπτόμενους πελάτες που βρίσκονται από αυτόν. Επιλέγεται ο πελάτης που βρίσκεται σε μικρότερη απόσταση (δηλαδή το "φθηνότερο" arc) και τον επισκέπτεται.

Αυτή η διαδικασία επαναλαμβάνεται μέχρι να επισκεφθούν όλοι οι πελάτες ή να μην υπάρχει διαθέσιμη κίνηση, δηλαδή όλοι οι πελάτες έχουν εξυπηρετηθεί ή η χωρητικότητα έχει φτάσει το μέγιστο επιτρεπόμενο όριο. Όταν δεν υπάρχουν άλλοι διαθέσιμοι πελάτες ή η χωρητικότητα έχει εξαντληθεί, το όχημα που εκτελεί τη τρέχουσα διαδρομή επιστρέφει στο depot. Οι διαδρομές που δημιουργούνται είναι οι αρχικές λύσεις για το πρόβλημα δρομολόγησης.

5.1.2 GUIDED LOCAL SEARCH

Η μέθοδος αυτή προτάθηκε από τους Voudouris et al.(2010)[23]. Ως μετα-ευρετική μέθοδος, η Guided Local Search (GLS) βρίσκεται πάνω από τους αλγορίθμους τοπικής αναζήτησης. Για την εφαρμογή του GLS, ορίζεται ένα σύνολο χαρακτηριστικών για τις υποψήφιες λύσεις. Όταν η τοπική αναζήτηση παγιδεύεται σε τοπικά βέλτιστα, επιλέγονται και τιμωρούνται ορισμένα χαρακτηριστικά. Η τοπική αναζήτηση, αναζητά το χώρο λύσεων χρησιμοποιώντας τη συνάρτηση στόχου, ενισχυμένη με τις συσσωρευμένες ποινές.

Η Guided Local Search χρησιμοποιεί μια διαδικασία τοπικής αναζήτησης που βρίσκει (άπληστα) ένα τοπικό ελάχιστο της επαυξημένης αντικειμενικής. Κάθε φορά που επιτυγχάνεται ένα τοπικό ελάχιστο, αυτός ο επαυξημένος στόχος ενημερώνεται, επιτρέποντας στην αναζήτηση να ξεφεύγει από τα τοπικά ελάχιστα.

Παρατίθεται ακολούθως ο ψευδοκώδικας που αναπαριστά τον τρόπο εύρεσης της λύσης μέσω χρήσης του αλγορίθμου.

Algorithm 3 Guided Local Search

```

1: Select features  $F_i, i = 1, \dots, N$ 
2: Initialize penalties  $p_i$  and costs  $c_i$ 
3: Generate initial solution  $x$ 
4: while stopping criterion is not met do
5:   Find a local minimum  $x^*$  of the augmented objective function using
     local search starting from  $x$ 
6:   Compute the utility  $u_i$  of each feature  $F_i$  of  $x^*$ 
7:    $j \leftarrow \arg \max_i u_i$ 
8:    $p_j \leftarrow p_j + 1$ 
9:    $x \leftarrow x^*$ 
10: end while
11: return best solution found

```

Εικόνα 5.1: Πρότυπο ψευδοκώδικα για τον αλγόριθμο Guided Local Search [24].

Σύμφωνα με τους Boussaid, Lepagnot et.al (2013) [25] ο GLS ορίζει ένα σύνολο χαρακτηριστικών $F_i, i = 1, \dots, N$ και μια συνάρτηση $I_i(x)$ που δείχνει αν μια λύση x έχει το χαρακτηριστικό i . Στα χαρακτηριστικά δίνονται κόστη c_i (τα οποία δεν αλλάζουν κατά τη διάρκεια της αναζήτησης) και ποινές p_i , οι οποίες αρχικοποιούνται στο μηδέν και αυξάνονται κατά τη διάρκεια της αναζήτησης ανάλογα με τις λύσεις που επισκέπτονται. Ο επαυξημένος στόχος ενημερώνεται μόνο με την αλλαγή των ποινών p_i και το χαρακτηριστικό που θα τιμωρηθεί σε κάθε επανάληψη επιλέγεται με έναν υπολογισμό χρησιμότητας. Η χρησιμότητα της τιμωρίας του χαρακτηριστικού i σε ένα τοπικό βέλτιστο x^* είναι $u_i(x^*) = I_i(x^*) \frac{c_i}{1+p_i}$. Ο σκοπός αυτού είναι να βρεθούν τα χαρακτηριστικά που έχουν τη μεγαλύτερη επίπτωση όσον αφορά την αποφυγή ανακύκλωσης της αναζήτησης.

Ο επαυξημένος στόχος ορίζεται ως $g(x) = f(x) + \lambda \sum_i^N p_i I_i(x)$, όπου $f(x)$ είναι η αμετάβλητη αντικειμενική συνάρτηση και λ να είναι μια παράμετρος που ρυθμίζει τη σχετική ισχύ των ποινών. Ένα μεγάλο λ επιφέρει μεγάλες ποινές και διαφοροποιεί την αναζήτηση, ενώ ένα μικρό λ εντείνει την αναζήτηση ($\lambda = 0$ αντιστοιχεί στον αμετάβλητο αντικειμενικό στόχο). Αυτή η διαδικασία τιμωρίας χαρακτηριστικών εστιάζει την αναζήτηση σε πολλά υποσχόμενες περιοχές [24].

5.2 ΥΠΟΛΟΓΙΣΜΟΣ ΑΠΟΣΤΑΣΕΩΝ

Το πρώτο βήμα είναι η δημιουργία ενός πίνακα $n \times n$, όπου κάθε στοιχείο του πίνακα $[i, j]$ αντιπροσωπεύει την απόσταση μεταξύ του σημείου i και του σημείου j . Η απόσταση μεταξύ δύο τοποθεσιών μπορεί να υπολογιστεί με απλή Ευκλείδεια μέθοδο ή με πιο σύνθετους τρόπους, όπως η χρήση API δρομολόγησης που χρησιμοποιεί το Google Maps για να παρέχει πραγματικές αποστάσεις βάσει των κυκλοφοριακών συνθηκών. Τα δεδομένα για τη δημιουργία αυτού του πίνακα αντλούνται από ένα αρχείο Excel μέσω ενός εμπορικού προγράμματος ERP που χρησιμοποιεί η εταιρεία, το οποίο παράγεται με βάση τις καθημερινές πωλήσεις προϊόντων αρτοσκευασμάτων και σνακ που πραγματοποιούν οι συνεργαζόμενες εταιρείες με την 3PL. Ο πίνακας αποστάσεων που θα δημιουργηθεί πρέπει να περιλαμβάνει στοιχεία ακέραιων αριθμών καθώς ο Solver του Google OR-Tools και συγκεκριμένα ο Solver που χρησιμοποιείται για τα προβλήματα δρομολόγησης οχημάτων όπως το VRP, υποστηρίζει μόνο ακέραιες τιμές (π.χ. τύπου `int64_t`).

Όσον αφορά τα δεδομένα του προβλήματος, έχουμε τον αριθμό του στόλου οχημάτων που μελετάται να ανέρχεται σε 4 οχήματα, όπου 2 εξυπηρετούν μεγάλο όγκο σε προϊόντα και διανέμουν σε μεγάλα καταστήματα τύπου Super Market, χονδρέμπορους κ.α. και θεωρούνται ομοιόμορφα με μέγιστη χωρητικότητα 15 παλέτες και 2 ακόμη να εξυπηρετούν τους υπόλοιπους πελάτες με μέγιστη χωρητικότητα 6 παλέτες. Τα οχήματα αυτά ξεκινάνε πάντα από την αποθήκη που βρίσκεται στη θέση 0.

Αξίζει να σημειωθεί ότι, σε κάθε χρήση του κώδικα, η σειρά των πελατών που μελετάμε στον πίνακα αποστάσεων είναι αυθαίρετη και σχετίζεται με τα φίλτρα που έχουν επιλεγεί από το αρχικό αρχείο Excel μαζί με το αρχείο Excel που έχουμε που αφορά τις συντεταγμένες των πελατών.

Στο υπό μελέτη πρόβλημα χρησιμοποιείται συνάρτηση για τον υπολογισμό των αποστάσεων μεταξύ των σημείων και συμπληρώνεται ο σχετικός πίνακας. Η λύση που μπορούμε να πάρουμε, είναι μέσω της χρήσης του Google Maps Distance Matrix API καθώς είναι ο τρόπος που η ίδια η Google χρησιμοποιεί στο Google Maps. Ωστόσο, το API key που δίνεται δωρεάν μπορεί, βάση είτε των διευθύνσεων είτε των Plus Code των πελατών, να

δώσει έναν πίνακα διάστασης μόνο 26x26. Το οποίο συνεπάγεται ότι μπορεί να δώσει αποτέλεσμα για δεδομένα μόνο από 25 πελάτες. Στην συγκεκριμένη εργασία, καθώς το πλήθος των πελατών που μελετώνται είναι πάνω από 25, υπολογίστηκε ο πίνακας αποστάσεων βάση των γεωγραφικών συντεταγμένων των πελατών.

Συγκεκριμένα, λαμβάνοντας τις γεωγραφικές συντεταγμένες των πελάτων που βρίσκονται στο σχετικό αρχείο Excel, υπολογίστηκε η Haversine distance. Κατά αυτή την διαδικασία [26] εφαρμόζεται, όπως φαίνεται παρακάτω, μια φόρμουλα για να υπολογιστεί η απόσταση μεγάλου κύκλου μεταξύ δύο σημείων, δηλαδή η μικρότερη απόσταση πάνω από την επιφάνεια της γης.

$$a = \sin^2\left(\frac{\Delta\varphi}{2}\right) + \cos\varphi_1 \times \cos\varphi_2 \times \sin^2\left(\frac{\Delta\lambda}{2}\right) \quad 5.1$$

$$c = 2 \times \operatorname{atan2}(\sqrt{a}, \sqrt{1-a}) \quad \text{Εξ.5.2}$$

$$d = R \times c \quad \text{Εξ.5.3}$$

όπου φ είναι το γεωγραφικό πλάτος, λ είναι το γεωγραφικό μήκος και R είναι η ακτίνα της γης (μέση ακτίνα = 6.371 km). Σημειώνεται επίσης ότι καθώς οι γωνίες πρέπει να είναι σε ακτίνια γίνεται σχετική μετατροπή αυτών πολλαπλασιάζοντας με τη σταθερά π και διαιρώντας με 180.

Προκειμένου να διασφαλιστεί η ακρίβεια των αποτελεσμάτων, σε ένα τυχαίο παράδειγμα όπου εξετάζονται λιγότεροι από 25 πελάτες, πραγματοποιήθηκε σύγκριση του πίνακα αποστάσεων που παίρνουμε κάνοντας χρήση των πραγματικών διευθύνσεων μέσω χρήσης του API (Εικόνα 5.2), σε σχέση με τον πίνακα αποστάσεων που δημιουργείται με βάση τις γεωγραφικές συντεταγμένες των πελατών (Εικόνα 5.3).

Προκειμένου να γίνει έλεγχος της απόκλισης που έχουμε μεταξύ των δεδομένων στους δυο πίνακες, υπολογίστηκε το ποσοστό της απόλυτης διαφοράς τους. Αυτός ο υπολογισμός έγινε με χρήση του τύπου:

$$\text{Ποσοστιαία διαφορά} = \frac{|Data \text{ from Distance Matrix1} - Data \text{ from Distance Matrix2}|}{Data \text{ from Distance Matrix1}}$$

Εξ.5.4

0	21088	22125	20530	22215	26701	21054	33197	33443	33421	33286	29297	42866	41262	43132	43352
20650	0	6604	6858	6694	4198	6599	10694	10939	10918	10783	10265	20362	18759	20629	20849
21859	6604	0	329	90	6666	678	13162	13407	13386	13251	7172	22830	21226	23097	23317
20470	6858	329	0	387	6995	524	13491	13736	13715	13580	7469	23159	21556	23426	23646
21949	6694	90	387	0	6756	736	13252	13498	13476	13342	7082	22921	21317	23187	23407
26662	6507	6637	6967	6728	0	7051	6496	6742	6720	6586	5920	16165	14561	16431	16651
22315	6599	678	524	736	6833	0	13329	13575	13553	13419	7817	22998	21394	23264	23484
33219	11228	13195	13524	13285	6558	13609	0	246	224	89	4954	9820	8216	10087	10306
33465	11473	13440	13770	13531	6803	13854	246	0	71	156	5200	9667	8063	9933	10153
33443	11452	13419	13748	13509	6781	13833	224	71	0	134	5178	9596	7992	9863	10083
33308	11317	13284	13614	13375	6647	13698	89	156	134	0	5044	9731	8127	9997	10217
29076	10241	7217	7514	7207	10267	7863	4954	5200	5178	5044	0	15529	10484	15795	16015
42799	21149	22774	23104	22865	16137	23188	9921	9768	9697	9832	14875	0	3558	295	552
39606	17956	19582	19911	19672	12945	19996	6729	6575	6505	6639	11683	3616	0	3883	4103
43094	21444	23070	23399	23160	16432	23483	10216	10063	9992	10127	15171	295	3853	0	227
44637	22987	24613	24942	24703	17975	25027	11759	11606	11536	11670	16714	1094	4764	227	0

Εικόνα 5.2: Πίνακας αποστάσεων υπολογισμένους βάση πραγματικών διαδρομών με χρήση API key.

0	15500	18563	18304	18619	18531	18181	24508	24692	24685	24575	22417	33329	30030	33596	33788
15500	0	4855	4794	4878	3294	4462	9101	9302	9292	9174	6971	17831	14532	18100	18293
18563	4855	0	305	62	5520	421	8552	8795	8771	8638	5262	15664	12737	15891	16054
18304	4794	305	0	366	5646	359	8833	9076	9053	8920	5555	15967	13042	16194	16357
18619	4878	62	366	0	5505	464	8499	8742	8719	8586	5206	15602	12676	15829	15993
18531	3294	5520	5646	5505	0	5299	5980	6161	6154	6045	4622	14960	11595	15239	15441
18181	4462	421	359	464	5299	0	8638	8880	8858	8725	5401	15931	12964	16163	16329
24508	9101	8552	8833	8499	5980	8638	0	243	219	86	3381	9104	5712	9392	9601
24692	9302	8795	9076	8742	6161	8880	243	0	30	157	3622	8977	5585	9266	9476
24685	9292	8771	9053	8719	6154	8858	219	30	0	133	3595	8974	5582	9264	9474
24575	9174	8638	8920	8586	6045	8725	86	157	133	0	3466	9056	5664	9345	9554
22417	6971	5262	5555	5206	4622	5401	3381	3622	3595	3466	0	10953	7768	11211	11397
33329	17831	15664	15967	15602	14960	15931	9104	8977	8974	9056	10953	0	3393	296	512
30030	14532	12737	13042	12676	11595	12964	5712	5585	5582	5664	7768	3393	0	3682	3892
33596	18100	15891	16194	15829	15239	16163	9392	9266	9264	9345	11211	296	3682	0	216
33788	18293	16054	16357	15993	15441	16329	9601	9476	9474	9554	11397	512	3892	216	0

Εικόνα 5.3: Πίνακας αποστάσεων υπολογισμένους με χρήση γεωγραφικών συντεταγμένων.

Ο Πίνακας 5.4 δείχνει τον υπολογισμό αυτόν που πρόκυψε για κάθε στοιχείο του πίνακα:

0	0,264985	0,160994	0,108427	0,161873	0,305981	0,136459	0,261741	0,261669	0,261393	0,261702	0,234836	0,222484	0,272212	0,221089	0,220613
0,249395	0	0,264839	0,300962	0,271288	0,215341	0,323837	0,148962	0,149648	0,148928	0,149216	0,320896	0,1243	0,225332	0,122594	0,122596
0,150785	0,264839	0	0,072948	0,311111	0,171917	0,379056	0,350251	0,343999	0,344763	0,348125	0,266313	0,313885	0,399934	0,311989	0,311489
0,105813	0,300962	0,072948	0	0,054264	0,192852	0,314885	0,345267	0,339255	0,33992	0,343152	0,256259	0,310549	0,394971	0,308717	0,308255
0,151715	0,271288	0,311111	0,054264	0	0,185169	0,369565	0,358663	0,352348	0,352998	0,356468	0,264897	0,319314	0,405357	0,317333	0,316743
0,304966	0,493776	0,168299	0,189608	0,181778	0	0,248475	0,079433	0,086176	0,084226	0,082144	0,219257	0,074544	0,203695	0,072546	0,072668
0,185257	0,323837	0,379056	0,314885	0,369565	0,224499	0	0,351939	0,345856	0,346418	0,349803	0,30907	0,307288	0,394036	0,305236	0,304676
0,262229	0,189437	0,351876	0,346865	0,360256	0,088137	0,365273	0	0,012195	0,022321	0,033708	0,317521	0,072912	0,304771	0,068901	0,068407
0,262154	0,189227	0,34561	0,340886	0,353928	0,09437	0,35903	0,012195	0	0,577465	0,00641	0,303462	0,071377	0,30733	0,06715	0,06668
0,261878	0,188613	0,346375	0,341504	0,354578	0,092464	0,359647	0,022321	0,577465	0	0,007463	0,305716	0,064819	0,301552	0,060732	0,060399
0,262189	0,189361	0,349744	0,344792	0,358056	0,090567	0,363046	0,033708	0,00641	0,007463	0	0,312847	0,069366	0,303064	0,06522	0,064892
0,22902	0,319305	0,270888	0,260713	0,277647	0,54982	0,313112	0,317521	0,303462	0,305716	0,312847	0	0,294674	0,259061	0,290218	0,288355
0,221267	0,156887	0,312198	0,308908	0,317647	0,072938	0,312964	0,082351	0,080979	0,074559	0,078926	0,263664	0	0,046374	0,00339	0,072464
0,241782	0,190688	0,349556	0,344985	0,355632	0,104287	0,35167	0,151137	0,15057	0,141891	0,146859	0,335102	0,06167	0	0,051764	0,051426
0,220402	0,155941	0,311183	0,307919	0,316537	0,072602	0,311715	0,080658	0,079201	0,072858	0,077219	0,261024	0,00339	0,044381	0	0,048458
0,243049	0,204202	0,347743	0,344199	0,352589	0,140974	0,347545	0,183519	0,183526	0,178745	0,18132	0,318117	0,531993	0,183039	0,048458	0

Εικόνα 5.4: Πίνακας που περιλαμβάνει τη διαφορά μεταξύ των αποστάσεων που υπολογίστηκαν βάση των πραγματικών διαδρομών και με χρήση γεωγραφικών συντεταγμένων.

Τέλος για την εύρεση της ποσοστιαίας διαφοράς ανάμεσα στις δυο μεθόδους υπολογισμού αυτή προκύπτει αθροίζοντας όλα τα στοιχεία του πίνακα και διαιρώντας τα με n^2 όπου n ο αριθμός των δεδομένων εισόδου που έχουμε (συνολικός αριθμός πελατών +1 για την αποθήκη). Έτσι η ποσοστιαία διαφορά που βρέθηκε είναι 0,215438.

Η διαφορά ανάμεσα στις αποστάσεις που υπολογίζονται με χρήση API και αυτές που υπολογίζονται βάσει γεωγραφικών συντεταγμένων είναι αρκετά υψηλή (21%). Η μεγάλη

αυτή απόκλιση οφείλεται στο γεγονός ότι το API παρέχει ακριβείς αποστάσεις βασισμένες στην πραγματική διαδρομή που ακολουθεί ένα όχημα, λαμβάνοντας υπόψη διάφορους παράγοντες όπως το οδικό δίκτυο, οι συνθήκες κυκλοφορίας, οι μονοδρομήσεις και οι περιορισμοί ταχύτητας. Αντίθετα, ο υπολογισμός με χρήση γεωγραφικών συντεταγμένων βασίζεται στην ευθεία απόσταση μεταξύ των σημείων, χωρίς να λαμβάνει υπόψη τις πραγματικές συνθήκες του οδικού δικτύου, γεγονός που οδηγεί σε λιγότερο ακριβείς εκτιμήσεις των αποστάσεων.

Η ίδια μεθοδολογία πραγματοποιήθηκε ακόμη 3 φορές χρησιμοποιώντας διαφορετικούς πελάτες για να βρούμε μια μέση τιμή. Οπότε οι τιμές που βρέθηκαν για την ποσοστιαία διαφορά ήταν

1. 0,259200472
2. 0,265839066
3. 0,221613015

Ένας μέσος αυτών λοιπόν βρέθηκε να είναι 0,240522557.

Επομένως, η μεγάλη απόκλιση που παρατηρείτε είναι λογική, δεδομένου ότι οι δύο μέθοδοι υπολογισμού έχουν διαφορετικές βάσεις. Το API παρέχει ακριβή δεδομένα σύμφωνα με την πραγματική διαδρομή, ενώ οι γεωγραφικές συντεταγμένες προσφέρουν μία πιο απλή μέτρηση.

Στον τελικό πίνακα που θα χρησιμοποιηθεί για την επίλυση του προβλήματος, θα πολλαπλασιαστεί αυτό το ποσοστό προκειμένου να ληφθεί υπόψη η απόκλιση και να τροποποιηθούν τα αποτελέσματα με βάση τις ακριβείς αποστάσεις που παρέχει το API. Αυτή η προσέγγιση θα βοηθήσει ώστε να υπάρξει καλύτερη ισορροπία μεταξύ της απόδοσης και της ακρίβειας των αποτελεσμάτων.

5.3 ΚΩΔΙΚΑΣ

5.3.1 ΔΗΜΙΟΥΡΓΙΑ ΔΕΔΟΜΕΝΩΝ

Η συνάρτηση `create_data_model`, όπως φαίνεται στην εικόνα 5.5, κατασκευάζει ένα μοντέλο δεδομένων για το πρόβλημα δρομολόγησης με βάση τα δεδομένα των πελατών από μια καθορισμένη ημερομηνία που εισάγεται από τον χρήστη. Ξεκινά ορίζοντας σταθερές συντεταγμένες για την αποθήκη και διαβάζει ένα αρχείο Excel που περιέχει τις πληροφορίες των πελατών. Η συνάρτηση φιλτράρει αυτά τα δεδομένα για να περιλαμβάνει μόνο καταχωρήσεις που ταιριάζουν με την καθορισμένη ημερομηνία, τα ID των πελατών, τις συντεταγμένες και τις ζητήσεις τους. Φυσικά αν ο χρήστης έχει εισάγει ημερομηνία που δεν βρίσκεται στο σχετικό αρχείο, τότε εμφανίζεται σχετικό μήνυμα στην οθόνη.

```
def create_data_model(date):
    data = {}

    depot_lat = ---
    depot_lon = ---

    file_path = '---'
    output_file = pd.read_excel(file_path)

    output_file['Creation_Time'] = pd.to_datetime(output_file['Creation_Time']).dt.date
    output_file_s = output_file[output_file['Creation_Time'] == date]

    if output_file_s.empty:
        print(f"Couldn't find data for the specified date: {date}")
        return None

    output_file_s = output_file_s[output_file_s['Total_QTY/PAL'] > 0]

    depot_data = {'CustomerID': 'Depot', 'Latitude': depot_lat, 'Longitude': depot_lon, 'Total_QTY/PAL': 0}
    output_file_s = pd.concat([pd.DataFrame([depot_data]), output_file_s], ignore_index=True)

    customer_ids = output_file_s['CustomerID'].tolist()
    demands = output_file_s['Total_QTY/PAL'].tolist()

    latitudes = output_file_s['Latitude'].tolist()
    longitudes = output_file_s['Longitude'].tolist()

    distance_matrix = calculate_distance_matrix(output_file_s)
```

Εικόνα 5.5: Δημιουργία δεδομένων μέσω της συνάρτησης `create_data_model`.

Έπειτα φιλτράρονται οι σειρές του DataFrame που χρησιμοποιείται, κρατώντας μόνο τις γραμμές όπου η στήλη 'Total_QTY/PAL' έχει τιμή μεγαλύτερη από 0. Αυτό πραγματοποιείται ώστε όταν θα γίνει η εκτέλεση του κώδικα, να διασφαλίζεται ότι στη συνέχεια της επεξεργασίας θα ληφθούν υπόψη μόνο οι πελάτες που έχουν θετική ζήτηση ώστε να μην εισάγονται στις διαδρομές πελάτες οι οποίοι δεν έχουν ζήτηση.

Στη συνέχεια, εξάγονται οι στήλες 'CustomerID', 'Latitude', 'Longitude', 'Total_QTY/PAL' από τα δεδομένα και μετατρέπονται σε λίστες, μέσω της συνάρτησης .tolist(), ώστε στη συνέχεια να εισαχθούν οι πληροφορίες της αποθήκης σε αυτές τις λίστες. Συγκεκριμένα, μέσω της συνάρτησης .insert εισάγονται οι συντεταγμένες της αποθήκης στην αρχή της λίστας, εισάγεται ζήτηση ίση με μηδέν για την αποθήκη και προστίθεται το αναγνωριστικό 'Depot' στην αρχή της λίστας των αναγνωριστικών πελατών (customer IDs).

Στη συνέχεια, γίνεται κλήση της συνάρτησης calculate_distance_matrix, που υπολογίζει και επιστρέφει τον πίνακα αποστάσεων για τα δεδομένα του DataFrame output_file_s. Η συνάρτηση υπολογίζει έναν συμμετρικό πίνακα αποστάσεων χρησιμοποιώντας τον τύπο Haversine, προσαρμόζοντας τις αποστάσεις με έναν παράγοντα, όπως περιεγράφηκε προηγουμένως.

```
def calculate_distance_matrix(dataframe):  
    num_customers = len(dataframe)  
    distance_matrix = [[0] * num_customers for _ in range(num_customers)]  
  
    for i in range(num_customers):  
        lat1 = dataframe.iloc[i]['Latitude']  
        lon1 = dataframe.iloc[i]['Longitude']  
        for j in range(num_customers):  
            if i != j:  
                lat2 = dataframe.iloc[j]['Latitude']  
                lon2 = dataframe.iloc[j]['Longitude']  
                distance_matrix[i][j] = haversine_distance(lat1, lon1, lat2, lon2)  
                # Προσαρμογή της απόστασης με τον συντελεστή FACTOR  
                distance_matrix[i][j] = round(distance_matrix[i][j] * (1 + FACTOR))  
    return distance_matrix
```

Εικόνα 5.6: Υπολογισμός του πίνακα αποστάσεων, μέσω της συνάρτησης calculate_distance_matrix.

Το λεξικό δεδομένων συμπληρώνεται με αυτόν τον πίνακα απόστασης, τις απαιτήσεις των πελατών, τα ID τους, τις χωρητικότητες των οχημάτων, τον αριθμό των οχημάτων και τον δείκτη της αποθήκης. Ορίζεται ένα σύνολο πελατών για μεγάλες αγορές (Supermarket,

χονδρέμπορους), υποδεικνύοντας πελάτες που μπορούν να εξυπηρετηθούν μόνο από συγκεκριμένα οχήματα, οι οποίοι χαρακτηρίζονται από συγκεκριμένα ID.

```
distance_matrix = calculate_distance_matrix(output_file_s)

data["distance_matrix"] = distance_matrix
data["demands"] = demands
data["customer_ids"] = customer_ids
data["vehicle_capacities"] = [150000, 150000, 60000, 60000]
data["num_vehicles"] = 4
data["depot"] = 0 # το depot είναι το πρώτο στοιχείο

# ID πελατών που μπορούν να εξυπηρετηθούν μόνο από τα οχήματα 1 και 2
data["big_markets"] = {6, 17, 27, 28, 32, 71, 72, 73, 74, 75, 76, 132, 156, 161, 170, 233,
                       234, 235, 236, 237, 238, 239, 240, 241, 242, 243, 244, 245, 246, 247,
                       248, 249, 250, 251, 252, 253, 254, 255, 256, 257, 258, 282, 292, 293, 294}

return data
```

Εικόνα 5.7: Υπολογισμός πίνακα απόστασης και δημιουργία λεξικού δεδομένων.

Οι χωρητικότητες των οχημάτων ορίζονται σε 150.000 και 60.000, οι οποίες αντιστοιχούν σε 15 και 6 παλέτες αντίστοιχα, για κάθε κατηγορία οχήματος. Ο συνολικός αριθμός των διαθέσιμων οχημάτων ανέρχεται σε 4, ενώ το σημείο εκκίνησης έχει οριστεί στο 0, το οποίο αντιστοιχεί στην αποθήκη.

Τέλος, η συνάρτηση επιστρέφει το συμπληρωμένο λεξικό δεδομένων όπως φαίνεται από την εικόνα 5.7.

5.3.2 ΔΗΜΙΟΥΡΓΙΑ ΜΟΝΤΕΛΟΥ

Στο κύριο τμήμα του αλγορίθμου, όπως φαίνεται στο παρακάτω τμήμα κώδικα της Εικόνας 5.8, δημιουργείται ένας δείκτης (index manager) και το μοντέλο δρομολόγησης (routing). Η μέθοδος manager.IndexToNode μετατρέπει τους εσωτερικούς δείκτες του solver σε αριθμούς για τις τοποθεσίες. Οι αριθμοί θέσης αντιστοιχούν στους δείκτες για τον πίνακα απόστασης.

```
def main():  
  
    user_date = input("Please give a date you wish to run the VRP (Year-Month-Day): ")  
    date = pd.to_datetime(user_date).date()  
  
    data = create_data_model(date)  
  
    if not data:  
        return  
  
    file_path = '---'  
    save_distance_matrix_to_file(data["distance_matrix"], file_path)  
  
    # Create the routing index manager  
    manager = pywrapcp.RoutingIndexManager(len(data["distance_matrix"]), data["num_vehicles"], data["depot"])  
  
    # Create Routing Model  
    routing = pywrapcp.RoutingModel(manager)
```

Εικόνα 5.8: Κύριο τμήμα κώδικα, εισαγωγή ημ/νίας από τον χρήστη και δημιουργία μοντέλου δρομολόγησης.

Ο solver της δρομολόγησης, χρησιμοποιεί εσωτερικούς δείκτες για τις μεταβλητές του προβλήματος, όπως τον πίνακα αποστάσεων, τον αριθμό των διαθέσιμων οχημάτων, το σημείο εκκίνησης κ.α.. Ο σκοπός της κλάσης `RoutingIndexManager` είναι να απλοποιήσει τη χρήση των μεταβλητών indexes (node indexes) και να επιτρέπει στους χρήστες να χρησιμοποιούν τις μεθόδους αυτής της κλάσης για να διαχειρίζονται και να χαρτογραφούν τα ευρετήρια των κόμβων σε προβλήματα δρομολόγησης. Η μοντελοποίηση και η επίλυση των προβλημάτων δρομολόγησης πραγματοποιείται μέσω της κλάσης `RoutingModel`.

Πολλές μεταβλητές προστίθενται κατά τη δημιουργία ενός μοντέλου και την προσθήκη περιορισμών. Κάθε θέση (κόμβος) έχει έναν κόμβο `Index` και ένα `variable_Index`. Για την επίλυση χρησιμοποιούνται μόνο τα `variableIndex's`, ενώ τα `Index's` του κόμβου χρησιμοποιούνται για να γνωρίζουμε ποιες τοποθεσίες έχουν επιλεγεί.

Τα δεδομένα εισόδου στο `RoutingIndexManager` είναι:

- Ο αριθμός των γραμμών του πίνακα αποστάσεων, που είναι ο αριθμός των τοποθεσιών (συμπεριλαμβανομένης της αποθήκης).
- Ο αριθμός των οχημάτων που υπάρχουν στο πρόβλημα.
- Το depot που δείχνει τον κόμβο που αντιστοιχεί στην αποθήκη.

Ενώ επιπλέον στο πρόβλημα προστίθενται δυο νέες ποσότητες, που αφορούν τη ζήτηση των πελατών και τη χωρητικότητα των οχημάτων.

- Ζήτηση: Κάθε πελάτης έχει μια ζήτηση που αντιστοιχεί στην ποσότητα, βάρος ή όγκο των προϊόντων που πρόκειται να παραλάβει.
- Χωρητικότητα: Κάθε όχημα έχει μια χωρητικότητα, τη μέγιστη ποσότητα που μπορεί να χωρέσει το όχημα. Καθώς ένα όχημα ταξιδεύει κατά μήκος της διαδρομής του, η συνολική ποσότητα των αντικειμένων που μεταφέρει δεν μπορεί ποτέ να υπερβεί τη χωρητικότητά του.

5.3.3 ΔΗΜΙΟΥΡΓΙΑ ΣΥΝΑΡΤΗΣΗΣ ΕΠΙΣΤΡΟΦΗΣ

Για να χρησιμοποιηθεί το εργαλείο επίλυσης δρομολόγησης, πρέπει να δημιουργηθεί μια επιστροφή κλήσης απόστασης (ή διέλευσης), μια συνάρτηση δηλαδή που λαμβάνει οποιοδήποτε ζεύγος τοποθεσιών και επιστρέφει την απόσταση μεταξύ τους. Η χρήση του πίνακα αποστάσεων είναι ο ευκολότερος τρόπος για να επιτευχθεί αυτό. Η ακόλουθη συνάρτηση, όπως φαίνεται στην Εικόνα 5.9, δημιουργεί την επανάκληση και την καταχωρεί στο πρόγραμμα επίλυσης ως `transit_callback_index` έχοντας σαν παραμέτρους δύο δείκτες τους `from_index` και `to_index`.

```
# Create and register a transit callback
def distance_callback(from_index, to_index):
    """Returns the distance between the two nodes."""
    from_node = manager.IndexToNode(from_index)
    to_node = manager.IndexToNode(to_index)
    return data["distance_matrix"][from_node][to_node]

transit_callback_index = routing.RegisterTransitCallback(distance_callback)
```

Εικόνα 5.9: Δημιουργία συνάρτησης επιστροφής.

5.3.4 ΔΗΜΙΟΥΡΓΙΑ ΚΟΣΤΟΥΣ ΤΑΞΙΔΙΟΥ

Για να βρει το μοντέλο μια λύση στο πρόβλημα, θα πρέπει να ανατρέχει συνεχώς σε δεδομένα που θα έχουν την απόσταση μεταξύ δύο κόμβων. Ο αξιολογητής κόστους τόξου

(CostEvaluator) δίνει στον solver τον τρόπο υπολογισμού του κόστους της διαδρομής μεταξύ δύο οποιωνδήποτε τοποθεσιών. Δηλαδή, ο δείκτης πρέπει να χρησιμοποιηθεί ως το βάρος της διαδρομής (ή του τόξου) μεταξύ δύο κόμβων. Το ακόλουθο κομμάτι κώδικα ορίζει τον αξιολογητή κόστους τόξου (Εικόνα 5.10).

```
# Define cost of each arc  
routing.SetArcCostEvaluatorOfAllVehicles(transit_callback_index)
```

Εικόνα 5.10: Συνάρτηση αξιολόγησης κόστους τόξου.

Στο τρέχον πρόβλημα, καθώς και στο κλασσικό TSP, ο εκτιμητής κόστους είναι το `transit_callback_index`. Αυτός ο εκτιμητής αποτελεί μια εσωτερική αναφορά του λύτη (solver) στη συνάρτηση απόστασης που επιστρέφει την απόσταση μεταξύ δύο τοποθεσιών. Αυτό σημαίνει ότι το κόστος ταξιδιού μεταξύ οποιωνδήποτε δύο τοποθεσιών είναι ακριβώς η απόσταση μεταξύ τους. Ωστόσο, γενικά το κόστος μπορεί να περιλαμβάνει και άλλους παράγοντες.

Μπορούν επίσης να οριστούν πολλοί αξιολογητές κόστους τόξου που εξαρτώνται από το όχημα που ταξιδεύει μεταξύ τοποθεσιών, χρησιμοποιώντας τη μέθοδο `routing.SetArcCostEvaluatorOfVehicle()`. Για παράδειγμα, εάν τα οχήματα έχουν διαφορετικές ταχύτητες, θα μπορούσε να οριστεί το κόστος μετακίνησης μεταξύ τοποθεσιών ως την απόσταση διαιρούμενη με την ταχύτητα του οχήματος, δηλαδή τον χρόνο ταξιδιού.

5.3.5 ΟΡΙΣΜΟΣ ΔΙΑΣΤΑΣΗΣ ΑΠΟΣΤΑΣΗΣ

Για την επίλυση του VRP, πρέπει να δημιουργηθεί μια διάσταση απόστασης, η οποία υπολογίζει τη σωρευτική απόσταση που διανύει κάθε όχημα κατά μήκος της διαδρομής του. Στη συνέχεια, ορίζεται ένα κόστος ανάλογο με το μέγιστο των συνολικών αποστάσεων σε κάθε διαδρομή. Τα προγράμματα δρομολόγησης χρησιμοποιούν διαστάσεις για να παρακολουθούν τις ποσότητες που συσσωρεύονται στη διαδρομή ενός οχήματος.

Ο κώδικας δημιουργεί τη διάσταση της απόστασης, χρησιμοποιώντας τη μέθοδο `AddDimension` του `solver`. Το όρισμα `transit_callback_index` είναι το ευρετήριο για το `distance_callback`.

Η μέθοδος `SetGlobalSpanCostCoefficient` ορίζει έναν μεγάλο συντελεστή (100) για το συνολικό εύρος των διαδρομών, που σε αυτό το παράδειγμα είναι το μέγιστο των αποστάσεων των διαδρομών. Αυτό καθιστά ένα εύρος κυρίαρχο παράγοντα στη συνάρτηση στόχου, επομένως το πρόγραμμα ελαχιστοποιεί το μήκος της μεγαλύτερης διαδρομής.

5.3.6 ΔΗΜΙΟΥΡΓΙΑ ΣΥΝΑΡΤΗΣΗΣ ΕΠΙΣΤΡΟΦΗΣ ΖΗΤΗΣΗΣ ΚΑΙ ΠΕΡΙΟΡΙΣΜΟΥ ΧΩΡΗΤΙΚΟΤΗΤΑΣ

Εκτός από την επιστροφή κλήσης απόστασης, ο `solver` απαιτεί επίσης μια συνάρτηση επανάκλησης ζήτησης, η οποία επιστρέφει τη ζήτηση σε κάθε τοποθεσία και μια διάσταση για τους περιορισμούς χωρητικότητας (Εικόνα 5.11).

```
def demand_callback(from_index):
    """Returns the demand of the node."""
    from_node = manager.IndexToNode(from_index)
    return data["demands"][from_node]

demand_callback_index = routing.RegisterUnaryTransitCallback(demand_callback)
routing.AddDimensionWithVehicleCapacity(
    demand_callback_index,
    0, # null capacity slack
    data["vehicle_capacities"], # vehicle maximum capacities
    True, # start cumul to zero
    "Capacity",
)
```

Εικόνα 5.11: Συνάρτηση επανάκλησης ζήτησης.

Η συνάρτηση επιστροφής ζήτησης εξαρτάται μόνο από την τοποθεσία (`from_node`) του σημείου παράδοσης. Δεδομένου ότι οι περιορισμοί χωρητικότητας περιλαμβάνουν το βάρος του φορτίου που μεταφέρει ένα όχημα, πρέπει να δημιουργήσουμε μια διάσταση για τις χωρητικότητες. Σε αυτήν την περίπτωση, χρησιμοποιούμε τη μέθοδο `AddDimensionWithVehicleCapacity`, η οποία παίρνει ένα διάνυσμα χωρητικότητας το οποίο

χειρίζεται τη γενικότερη περίπτωση στην οποία διαφορετικά οχήματα έχουν διαφορετική χωρητικότητα.

Συγκεκριμένα στη μέθοδο `AddDimension`, συναντάμε τις εξής παραμέτρους:

- `callback_index`: είναι ο δείκτης που αντιστοιχεί στην συνάρτηση `demand_callback`. Ο δείκτης αυτός δημιουργείτε από τη μέθοδο `RegisterUnaryTransitCallback`.
- `slack_max`: Το `capacity slack` καθορίζει την επιτρεπόμενη απόκλιση από τη μέγιστη χωρητικότητα του οχήματος κατά τον υπολογισμό του δρομολογίου. Με το `capacity slack` στο 0, δεν επιτρέπεται καμία απόκλιση από τη μέγιστη χωρητικότητα.
- `capacity`: Μεταβλητή χωρητικότητας ανά διαδρομή. Αυτός ο πίνακας περιέχει μια τιμή χωρητικότητας για κάθε όχημα στο πρόβλημα δρομολόγησης οχημάτων.
- `fix_start_cumulative_to_zero`: Boolean τιμή. Εάν είναι αληθές, η αθροιστική τιμή της ποσότητας ξεκινά από το 0. Στις περισσότερες περιπτώσεις, αυτό θα πρέπει να οριστεί σε True. Ωστόσο, για το VRPTW ή προβλήματα με περιορισμούς πόρων, ορισμένα οχήματα ενδέχεται να χρειαστεί να ξεκινήσουν μετά την ώρα 0 λόγω περιορισμών του χρονικού παραθύρου, επομένως θα πρέπει να ορίσετε το `fix_start_cumulative_to_zero` σε False για αυτά τα προβλήματα.
- `dimension_name`: Συμβολοσειρά για το όνομα της διάστασης, (`capacity` στο τρέχον πρόβλημα) η οποία μπορεί να χρησιμοποιηθεί για την πρόσβαση στις μεταβλητές σε άλλα σημεία του προγράμματος.

Επιπρόσθετα, προκειμένου και τα 4 οχήματα που υπάρχουν διαθέσιμα να χρησιμοποιούνται, στο πρόβλημα προστίθενται δυο ακόμη περιορισμοί.

1. Ο πρώτος περιορισμός που προστίθεται είναι αυτός που αφορά τον περιορισμό ελάχιστου φορτίου (Εικόνα 5.12), δηλαδή ότι κάθε όχημα πρέπει να φέρει τουλάχιστον ένα ελάχιστο φορτίο (10000 σε αυτήν την περίπτωση).

```
min_vehicle_load = 10000
capacity_dimension = routing.GetDimensionOrDie("Capacity")
for vehicle_id in range(data["num_vehicles"]):
    capacity_dimension.CumulVar(routing.End(vehicle_id)).SetMin(min_vehicle_load)
```

Εικόνα 5.12: Περιορισμός ελάχιστου φορτίου.

2. Περιορισμός για τον ελάχιστο αριθμό πελατών (Εικόνα 5.13) που οφείλει να εξυπηρετήσει κάθε όχημα. Σε αυτό το σημείο προστίθεται ο περιορισμός για τον ελάχιστο αριθμό πελατών που μπορεί να εξυπηρετήσει κάθε όχημα (`min_customer_count`) και εξασφαλίζει ότι κάθε όχημα εξυπηρετεί τουλάχιστον 5 πελάτες.

```
min_customer_count = 5
customer_count_dimension = routing.GetDimensionOrDie("CustomerCount")
for vehicle_id in range(data["num_vehicles"]):
    customer_count_dimension.CumulVar(routing.End(vehicle_id)).SetMin(min_customer_count)
```

Εικόνα 5.13: Περιορισμός ελαχίστου αριθμού πελατών προς εξυπηρέτηση.

5.3.7 ΟΡΙΣΜΟΣ ΠΑΡΑΜΕΤΡΩΝ ΑΝΑΖΗΤΗΣΗΣ

Το ακόλουθο κομμάτι κώδικα που παρέχεται (Εικόνα 5.14) αφορά τη ρύθμιση των παραμέτρων αναζήτησης του αλγορίθμου επίλυσης του προβλήματος δρομολόγησης με τη χρήση της κλάσης `DefaultRoutingSearchParameters`. Αυτές οι παράμετροι καθορίζουν τη στρατηγική επίλυσης και τους περιορισμούς που θα χρησιμοποιηθούν κατά την αναζήτηση της λύσης.

Ο κώδικας πρώτα βρίσκει μια αρχική λύση χρησιμοποιώντας την στρατηγική `PATH_CHEAPEST_ARC` και στη συνέχεια προσπαθεί να βελτιώσει αυτή τη λύση με την `GUIDED_LOCAL_SEARCH`. Αυτή είναι η στρατηγική που χρησιμοποιείται για τη βελτίωση της λύσης αφού έχει βρεθεί η αρχική λύση μέσω της προηγούμενης στρατηγικής. Ο παρακάτω κώδικας συνεπώς ορίζει τις προεπιλεγμένες παραμέτρους αναζήτησης και μια ευρετική μέθοδο για την εύρεση της πρώτης λύσης:

```
# Setting first solution heuristic
search_parameters = pywrapcp.DefaultRoutingSearchParameters()
search_parameters.first_solution_strategy = routing_enums_pb2.FirstSolutionStrategy.PATH_CHEAPEST_ARC
search_parameters.local_search_metaheuristic = routing_enums_pb2.LocalSearchMetaheuristic.GUIDED_LOCAL_SEARCH
search_parameters.time_limit.FromSeconds(1)
```

Εικόνα 5.14: Παράμετροι αναζήτησης αλγορίθμου επίλυσης.

Ο κώδικας ορίζει την πρώτη στρατηγική λύσης σε PATH_CHEAPEST_ARC, η οποία δημιουργεί μια αρχική διαδρομή για τον λύτη προσθέτοντας επανειλημμένα άκρα με το μικρότερο βάρος που δεν οδηγούν σε έναν κόμβο που έχει επισκεφτεί προηγουμένως (εκτός από την αποθήκη). Το PATH_CHEAPEST_ARC είναι ένας ευρετικός (heuristic) αλγόριθμος που χτίζει μια διαδρομή με βάση το κόστος (ή την απόσταση) για κάθε τόξο που προστίθεται στη διαδρομή. Ο αλγόριθμος επιλέγει πάντα το τόξο με το χαμηλότερο κόστος για να επεκτείνει την τρέχουσα διαδρομή.

5.3.8 ΟΡΙΣΜΟΣ ΣΥΝΑΡΤΗΣΗΣ ΕΚΤΥΠΩΣΗΣ

Η συνάρτηση που ακολουθεί στην Εικόνα 5.15 επιστρέφει τη διαδρομή κάθε οχήματος και το συνολικό όγκο που μεταφέρει το όχημα σε κάθε πελάτη που εξυπηρετεί στη διαδρομή του.

```
def print_solution(data, manager, routing, solution):
    """Prints solution on console."""
    print(f"Objective: {solution.ObjectiveValue()}")
    total_distance = 0
    total_load = 0
    total_customers = 0
    for vehicle in range(data["num_vehicles"]):
        index = routing.Start(vehicle)
        plan_output = f"Route for vehicle {vehicle+1}:\n"
        route_distance = 0
        route_load = 0
        route_customers = 0
        while not routing.IsEnd(index):
            node_index = manager.IndexToNode(index)
            route_load += data["demands"][node_index]
            customer_id = data["customer_ids"][node_index]
            if node_index != 0:
                route_customers += 1
            if node_index == 0:
                plan_output += f" {node_index} (CustomerID: {customer_id}) Deliver(0) -> "
            else:
                plan_output += f" {node_index} (CustomerID: {customer_id}) Deliver({data['demands'][node_index]}) -> "
            previous_index = index
            index = solution.Value(routing.NextVar(index))
            route_distance += routing.GetArcCostForVehicle(previous_index, index, vehicle)
        plan_output += f" {manager.IndexToNode(index)} (CustomerID: Depot) Load({route_load})\n"
        plan_output += f"Distance of the route: {route_distance}m\n"
        plan_output += f"Load of the route: {route_load}\n"
        plan_output += f"Customers served: {route_customers}\n"
        print(plan_output)
        total_distance += route_distance
        total_load += route_load
        total_customers += route_customers
    print(f"Total distance of all routes: {total_distance}m")
    print(f"Total load of all routes: {total_load}")
    print(f"Total customers served: {total_customers}")
```

Εικόνα 5.15: Συνάρτηση εκτύπωσης αποτελεσμάτων αλγορίθμου.

6 ΚΕΦΑΛΑΙΟ: ΕΠΙΛΥΣΗ VRP ΜΕ ΧΡΗΣΗ ΤΟΥ ΑΛΓΟΡΙΘΜΟΥ ΠΛΗΣΙΕΣΤΕΡΗΣ ΕΙΣΑΓΩΓΗΣ

Ο αλγόριθμος πλησιέστερης εισαγωγής αποτελεί έναν ευρετικό αλγόριθμο που έχει αποδειχτεί να παρέχει μια επιτυχημένη τεχνική που μπορεί να βρει εφαρμογή σε έναν αριθμό συνδυαστικών προβλημάτων βελτιστοποίησης, όπως το πρόβλημα δρομολόγησης οχημάτων. Ο αλγόριθμος πλησιέστερης εισαγωγής είναι μια παραλλαγή της ευρετικής του πλησιέστερου γείτονα (nearest neighbor), η οποία επιδιώκει να κατασκευάσει σταδιακά τη λύση. Η κύρια ιδέα είναι να ξεκινήσει από ένα αρχικό σύνολο κορυφών και να εισάγει επαναληπτικά νέες κορυφές με βάση τη μικρότερη απόσταση από την τρέχουσα δρομολόγηση. Ο αλγόριθμος συνεχίζει με αυτό τον τρόπο, προσθέτοντας πάντα την πιο κοντινή κορυφή, έως ότου όλες οι κορυφές συμπεριληφθούν στη διαδρομή.

Στο τρέχων κεφάλαιο γίνεται μελέτη της δυνατότητας εύρεσης λύσης στο πρόβλημα δρομολόγησης οχημάτων που μπορεί να προσφέρει ο αλγόριθμος αυτός και τίθεται μια πρακτική εφαρμογή της μεθόδου που χρησιμοποιείται για την επίλυση αυτού του προβλήματος.

Ο λόγος που κατασκευάστηκε ο συγκεκριμένος αλγόριθμος στην τρέχουσα εργασία ήταν για να προσομοιώσει τη φυσική συμπεριφορά των οδηγών κατά τη δημιουργία των δρομολογίων τους. Ο αλγόριθμος αυτός επιλέγει κάθε φορά τον πλησιέστερο πελάτη για να τον ενσωματώσει στο δρομολόγιο, κάτι που αντανακλά τον τρόπο με τον οποίο οι οδηγοί συνήθως πραγματοποιούν την εκτέλεση του δρομολογίου τους με σκοπό να ελαχιστοποιήσουν την απόσταση ή τον χρόνο μετακίνησης.

6.1 ΑΛΓΟΡΙΘΜΟΣ ΠΛΗΣΙΕΣΤΕΡΟΥ ΓΕΙΤΟΝΑ

Η μέθοδος πλησιέστερου γείτονα, η οποία εφαρμόζεται για την επίλυση του προβλήματος του πλανόδιου πωλητή, είναι μια διαδικασία κατά την οποία ο πωλητής

ξεκινάει από κάποια πόλη, στη συνέχεια επισκέπτεται την πόλη που είναι πλησιέστερη σε αυτήν που ξεκίνησε και επαναλαμβάνει τη διαδικασία μέχρι να επισκεφτεί όλες τις πόλεις από μια φορά και τότε επιστρέφει στην αρχική.

Η διαδικασία της μεθόδου αυτής, όπως περιγράφεται από τους (Rosenkrantz, Stearns, Lewis, 1974) [29] είναι η ακόλουθη:

Βήμα 1. Ξεκινάμε με οποιονδήποτε κόμβο ως ξεκινάμε του μονοπατιού.

Βήμα 2. Βρίσκουμε τον κόμβο που είναι πλησιέστερα στον τελευταίο κόμβο που προστέθηκε στη διαδρομή. Προσθέτουμε αυτό τον κόμβο στο μονοπάτι.

Βήμα 3. Επαναλαμβάνουμε το Βήμα 2 έως ότου όλες οι κορυφές να βρίσκονται στο μονοπάτι.

6.2 ΑΛΓΟΡΙΘΜΟΣ ΠΛΗΣΙΕΣΤΕΡΗΣ ΕΙΣΑΓΩΓΗΣ

Σε αυτόν τον αλγόριθμο, μια διαδρομή κατασκευάζεται, όπως περιγράφεται από τους (Rosenkrantz, Stearns, Lewis, 1974) [29] ως εξής:

Βήμα 1. Ξεκινάμε με ένα υπογράφημα που αποτελείται μόνο από τον κόμβο i .

Βήμα 2. Βρίσκουμε τον κόμβο r έτσι ώστε το κόστος c_{ir} να είναι ελάχιστο και σχηματίζουμε την υπο-περιήγηση $i-r-i$.

Βήμα 3. (Βήμα επιλογής) Με δεδομένη μια υπο-περιήγηση, βρίσκουμε τον κόμβο r που δεν ανήκει στην υπο-περιήγηση που βρίσκεται πλησιέστερα σε οποιονδήποτε κόμβο j της διαδρομής, δηλαδή με ελάχιστο c_{rj} .

Βήμα 4. (Βήμα εισαγωγής) Βρίσκουμε το τόξο (i, j) στην υποπεριήγηση που ελαχιστοποιεί το $c_{ir} + c_{rj} - c_{ij}$. Εισάγουμε το r μεταξύ i και j .

Βήμα 5. Εάν όλοι οι κόμβοι έχουν προστεθεί στην περιήγηση, η διαδικασία ολοκληρώνεται. Εάν υπάρχουν ακόμα κόμβοι εκτός της υπο-περιήγησης, επιστρέφουμε στο Βήμα 3 για να επιλέξουμε τον επόμενο κόμβο που θα προστεθεί.

Ο αλγόριθμος μπορεί να ακολουθήσει τα βήματα που περιεγράφηκαν προηγουμένως. Αυτή η ευρετική μέθοδος βασίζεται στην προσθήκη ενός κόμβου κάθε φορά στην τρέχουσα διαδρομή, λαμβάνοντας υπόψη την πλησιέστερη απόσταση, ενώ παράλληλα σέβεται περιορισμούς όπως η χωρητικότητα των οχημάτων. Ας αναλύσουμε τον αλγόριθμο περαιτέρω:

Η αρχικοποίηση των δεδομένων είναι το πρώτο βήμα όπου, έχουμε μια αποθήκη, έναν στόλο οχημάτων με μέγιστη χωρητικότητα και ένα σύνολο πελατών που χρειάζεται να εξυπηρετηθούν με συγκεκριμένες απαιτήσεις (ζήτηση προϊόντων). Ξεκινάμε με κάθε όχημα άδειο και χωρίς πελάτες στη διαδρομή του. Ορίζουμε ότι η αρχική διαδρομή για κάθε όχημα ξεκινάει από την αποθήκη και πρέπει να επιστρέψει σε αυτήν.

Επόμενο βήμα είναι η επιλογή του πρώτου πελάτη που θα εισαχθεί στη διαδρομή του πρώτου οχήματος. Για κάθε όχημα, ξεκινάμε επιλέγοντας τον πλησιέστερο πελάτη από την αποθήκη όπου θα είναι και ο πελάτης που θα εισαχθεί πρώτος στη διαδρομή του οχήματος. Μετά την επιλογή του πρώτου πελάτη, το επόμενο βήμα είναι να επιλέξουμε τον επόμενο πελάτη που δεν έχει εξυπηρετηθεί ακόμα και είναι ο πλησιέστερος σε οποιονδήποτε κόμβο της τρέχουσας διαδρομής (με βάση την απόσταση ή το κόστος).

Τέταρτο βήμα της διαδικασίας είναι, πριν την εισαγωγή του νέου πελάτη στη διαδρομή, να ελέγξουμε αν η ζήτηση αυτού του πελάτη μπορεί να εξυπηρετηθεί από το όχημα χωρίς να υπερβεί τη μέγιστη χωρητικότητα του οχήματος και εφόσον ισχύουν και οι υπόλοιποι περιορισμοί που τίθενται στο πρόβλημα δρομολόγησης. Αν η χωρητικότητα του οχήματος επαρκεί, προχωρούμε στην εισαγωγή του νέου πελάτη στην κατάλληλη θέση στη διαδρομή του οχήματος. Αν η χωρητικότητα του οχήματος δεν επαρκεί, ολοκληρώνουμε τη διαδρομή του οχήματος, επιστρέφοντας στην αποθήκη και ξεκινάμε νέα διαδρομή με άλλο όχημα ή ακόμη και με το ίδιο όχημα αν υπάρχει δυνατότητα.

Φυσικά πρέπει να βρεθεί η καταλληλότερη θέση στην οποία θα τοποθετηθεί ο πελάτης στη διαδρομή του οχήματος. Ο αλγόριθμος πρέπει να βρει την κατάλληλη θέση εισαγωγής του νέου πελάτη στη διαδρομή, όπου η αύξηση του συνολικού κόστους είναι η ελάχιστη. Για να γίνει αυτό, υπολογίζουμε το κόστος εισαγωγής του νέου πελάτη r μεταξύ δύο κόμβων i και j που βρίσκονται ήδη στη διαδρομή. Το κόστος υπολογίζεται ως:

$$c_{ir} + c_{rj} - c_{ij}$$

Εξ. 6.1

Ο πελάτης r εισάγεται στη θέση που αυτός ο υπολογισμός δίνει την ελάχιστη αύξηση στο κόστος.

Επόμενο βήμα της διαδικασίας είναι να επαναλαμβάνουμε τα Βήματα 3-5 μέχρι να προστεθούν όλοι οι πελάτες στη διαδρομή. Αν ένα όχημα γεμίσει πριν εξυπηρετήσει όλους τους πελάτες, επιστρέφει στην αποθήκη και ένα άλλο όχημα ξεκινά νέα διαδρομή για τους υπόλοιπους πελάτες.

Η διαδικασία τερματίζεται όταν όλοι οι πελάτες έχουν εισαχθεί σε κάποια διαδρομή και έχουν εξυπηρετηθεί.

Στο τέλος, έχουμε ένα σύνολο διαδρομών για τα οχήματα, όπου κάθε διαδρομή ξεκινά και καταλήγει στην αποθήκη και όλοι οι πελάτες έχουν εξυπηρετηθεί χωρίς να υπερβεί κανένα όχημα τη χωρητικότητά του ενώ παράλληλα τηρούνται όλοι οι περιορισμοί του προβλήματος.

Μια αναπαράσταση του ψευδοκώδικα που περιγράφει τον αλγόριθμο πλησιέστερης εισαγωγής όπως εφαρμόστηκε στον κώδικα όπως θα αναλυθεί παρακάτω, παρουσιάζεται στην Εικ. 6.1.

Procedure Nearest Insertion Heuristic

```
Select an arbitrary node  $j$  as the starting point
Set the initial tour  $T = \{j\}$ 
Set  $W = \{1, 2, \dots, n\} \setminus \{j\}$  // Remaining nodes to visit
While  $W \neq \emptyset$  do
  For each node  $l \in W$  do
    Find the best position to insert  $l$  into  $T$ 
    Calculate the cost of the insertion
    Let  $k$  be the position that minimizes the cost
    Insert node  $l$  into  $T$  at position  $k$ 
  Remove  $l$  from  $W$ 
End for
End While
Connect the last node in  $T$  to the starting node  $j$  to form a Hamiltonian cycle
```

Εικόνα 6.1: Ψευδοκώδικας για την περιγραφή του αλγορίθμου πλησιέστερης εισαγωγής.

6.3 ΚΩΔΙΚΑΣ

6.3.1 ΔΗΜΙΟΥΡΓΙΑ ΔΕΔΟΜΕΝΩΝ

Εδώ η διαδικασία που ακολουθείτε είναι παρόμοια με τη κατασκευή των δεδομένων όπως πραγματοποιήθηκε προηγουμένως για την επίλυση μέσω OR-Tools. Η διαφορά με πριν είναι ότι ο αλγόριθμος έχει σχεδιαστεί έτσι ώστε να επιλύει δύο ξεχωριστά VRP, ένα για τα μεγάλα οχήματα και ένα για τα μικρά οχήματα. Οπότε αρχικά ορίζονται παράμετροι για το πρόβλημα όπου έχουμε:

- `num_big_vehicles = 2` και `num_small_vehicles = 2`
- `big_vehicles_capacity = 150000` και `small_vehicles_capacity = 60000`

Η συνάρτηση `create_data_model`, όπως φαίνεται στην Εικόνα 6.2, κατασκευάζει ένα μοντέλο δεδομένων για το πρόβλημα δρομολόγησης με βάση τα δεδομένα των πελατών από μια καθορισμένη ημερομηνία που εισάγεται από τον χρήστη. Ξεκινά ορίζοντας σταθερές συντεταγμένες για την αποθήκη και διαβάζει ένα αρχείο Excel που περιέχει τις πληροφορίες των πελατών. Η συνάρτηση φιλτράρει αυτά τα δεδομένα για να περιλαμβάνει μόνο καταχωρήσεις που ταιριάζουν με την καθορισμένη ημερομηνία, τα ID των πελατών, τις συντεταγμένες και τις ζητήσεις τους. Φυσικά αν ο χρήστης έχει εισάγει ημερομηνία που δεν βρίσκεται στο σχετικό αρχείο, τότε εμφανίζεται σχετικό μήνυμα στην οθόνη.

Στη συνέχεια, εξάγει τις στήλες 'CustomerID', 'Latitude', 'Longitude', 'Total_QTY/PAL' από τα δεδομένα και τις μετατρέπει σε λίστες, μέσω της συνάρτησης `.tolist()`, ώστε στη συνέχεια να εισάγει τις πληροφορίες της αποθήκης σε αυτές τις λίστες. Συγκεκριμένα, μέσω της συνάρτησης `.insert` εισάγει τις συντεταγμένες της αποθήκης στην αρχή της λίστας, εισάγει ζήτηση ίση με μηδέν για την αποθήκη και προσθέτει το αναγνωριστικό 'Depot' στην αρχή της λίστας των αναγνωριστικών πελατών (customer IDs). Έπειτα πραγματοποιείται διαχωρισμός των πελατών όπου ορίζονται οι πελάτες σε δύο κατηγορίες:

- Μεγάλες Αγορές: Πελάτες με ID που περιλαμβάνονται στη λίστα `big_markets_IDs`.
- Μικρές Αγορές: Όλοι οι υπόλοιποι πελάτες.

```

def create_data_model(date):
    data = {}

    depot_lat = ---
    depot_lon = ---

    file_path = '---'
    output_file = pd.read_excel(file_path)

    output_file['Creation_Time'] = pd.to_datetime(output_file['Creation_Time']).dt.date
    output_file_s = output_file[output_file['Creation_Time'] == date]

    if output_file_s.empty:
        print(f"Couldn't find data for the specified date: {date}")
        return None

    output_file_s = output_file_s[output_file_s['Total_QTY/PAL'] > 0]

    depot_data = pd.DataFrame([{'CustomerID': 'Depot', 'Latitude': depot_lat, 'Longitude': depot_lon, 'Total_QTY/PAL': 0}])
    output_file_s = pd.concat([depot_data, output_file_s], ignore_index=True)

    big_markets_dataframe = output_file_s[output_file_s['CustomerID'].isin(big_markets_IDs)]
    small_markets_dataframe = output_file_s[~output_file_s['CustomerID'].isin(big_markets_IDs)]

    big_markets_dataframe = pd.concat([depot_data, big_markets_dataframe], ignore_index=True)
    small_markets_dataframe = pd.concat([depot_data, small_markets_dataframe], ignore_index=True)

    big_markets_distance_matrix = calculate_distance_matrix(big_markets_dataframe)
    small_markets_distance_matrix = calculate_distance_matrix(small_markets_dataframe)

    big_markets_demands = big_markets_dataframe['Total_QTY/PAL'].to_list()
    small_markets_demands = small_markets_dataframe['Total_QTY/PAL'].to_list()

```

Εικόνα 6.2: Εισαγωγή δεδομένων προβλήματος, μέσω της συνάρτησης `create_data_model`.

Στη συνέχεια, η συνάρτηση υπολογίζει έναν συμμετρικό πίνακα αποστάσεων χρησιμοποιώντας τον τύπο Haversine, προσαρμόζοντας τις αποστάσεις με έναν παράγοντα, όπως περιεγράφηκε προηγουμένως. Συγκεντρώνει όλα τα δεδομένα που υπολογίστηκαν σε μια δομή λεξικού, χωριστά για τις μεγάλες και μικρές αγορές. Το λεξικό δεδομένων συμπληρώνεται με αυτόν τον πίνακα απόστασης, τις απαιτήσεις των πελατών, τα ID τους, τις χωρητικότητες των οχημάτων, τον αριθμό των οχημάτων και τον δείκτη της αποθήκης. Ορίζεται ένα σύνολο ID πελατών για μεγάλες αγορές (Supermarket, χονδρέμπορους), υποδεικνύοντας πελάτες που μπορούν να εξυπηρετηθούν μόνο από συγκεκριμένα οχήματα.

Οι χωρητικότητες των οχημάτων ορίζονται σε 150.000 και 60.000, οι οποίες αντιστοιχούν σε 15 και 6 παλέτες αντίστοιχα, για κάθε κατηγορία οχήματος. Ο συνολικός αριθμός των διαθέσιμων οχημάτων ανέρχεται σε 4, ενώ το σημείο εκκίνησης έχει οριστεί στο 0, το οποίο αντιστοιχεί στην αποθήκη. Τέλος, η συνάρτηση επιστρέφει το συμπληρωμένο λεξικό δεδομένων όπως φαίνεται από την Εικόνα 6.3.

```
data["big_markets"] = {  
    "distance_matrix": big_markets_distance_matrix,  
    "demands": big_markets_demands,  
    "customer_ids": big_markets_dataframe['CustomerID'].tolist(),  
    "vehicles_capacity": big_vehicles_capacity,  
    "num_vehicles": 2  
}  
  
data["small_markets"] = {  
    "distance_matrix": small_markets_distance_matrix,  
    "demands": small_markets_demands,  
    "customer_ids": small_markets_dataframe['CustomerID'].tolist(),  
    "vehicles_capacity": small_vehicles_capacity,  
    "num_vehicles": 2  
}  
  
return data
```

Εικόνα 6.3: Δημιουργία λεξικού δεδομένων.

6.3.2 ΔΗΜΙΟΥΡΓΙΑ ΣΥΝΑΡΤΗΣΗΣ NEAREST INSERTION

Ας αναλύσουμε τη συνάρτηση `nearest_insertion`, η οποία υλοποιεί τον αντίστοιχο αλγόριθμο για την επίλυση του προβλήματος δρομολόγησης. Ο αλγόριθμος αυτός προσεγγίζει την βέλτιστη διαδρομή για την εξυπηρέτηση πελατών με δεδομένα που περιλαμβάνουν τις αποστάσεις μεταξύ των πελατών και τις απαιτήσεις τους.

Η συνάρτηση δέχεται ως ορίσματα:

- `num_customers`: Αντιστοιχεί στον αριθμό των πελατών.
- `distance_matrix`: Ο πίνακας αποστάσεων μεταξύ πελατών και αποθήκης.
- `demands`: Η ζήτηση κάθε πελάτη.
- `vehicle_capacity`: Η μέγιστη χωρητικότητα του οχήματος.
- `max_customers_per_vehicle`: Ο μέγιστος αριθμός πελατών που μπορεί να εξυπηρετήσει ένα όχημα.
- `num_vehicles`: Ο αριθμός των διαθέσιμων οχημάτων.

Εντός της συνάρτησης, όπως φαίνεται και στην Εικ. 6.4 περιλαμβάνεται μια εσωτερική συνάρτηση `find_best_position`, η οποία υπολογίζει την καλύτερη θέση όπου πρόκειται να γίνει εισαγωγή ενός νέου πελάτη σε μια υπάρχουσα διαδρομή (sub-tour), όπου για κάθε ζεύγος γειτονικών πελατών στο sub-tour υπολογίζεται το κόστος εισαγωγής και επιστρέφεται η καλύτερη θέση που αντιστοιχεί σε αυτή με το μικρότερο κόστος.

```
def find_best_position(sub_tour, closest_customer):
    best_insertion_cost = float('inf')
    best_position = None

    for i in range(len(sub_tour) - 1):
        j = i + 1
        insertion_cost = (distance_matrix[sub_tour[i]][closest_customer] +
                          distance_matrix[closest_customer][sub_tour[j]] -
                          distance_matrix[sub_tour[i]][sub_tour[j]])

        if insertion_cost < best_insertion_cost:
            best_insertion_cost = insertion_cost
            best_position = j

    return best_position
```

Εικόνα 6.4: Δημιουργία συνάρτησης εύρεσης καλύτερης θέσης εισαγωγής πελατών εντός της υπο-διαδρομής.

Στη συνέχεια ορίζεται ένα σύνολο με τους πελάτες που δεν έχουν εξυπηρετηθεί ακόμη, ξεκινώντας από τον πελάτη με ID 1, καθώς ο πελάτης με ID 0 αντιστοιχεί στην αποθήκη, έως τον αριθμό των πελατών.

Προχωρώντας στο κύριο τμήμα του αλγορίθμου, εφαρμόζονται επαναληπτικοί βρόγχοι για τη δημιουργία των διαδρομών, ξεκινώντας τη διαδικασία όσο υπάρχουν μη εξυπηρετημένοι πελάτες και δεν έχουν χρησιμοποιηθεί όλα τα διαθέσιμα οχήματα. Δημιουργούμε μια νέα υπο-διαδρομή ξεκινώντας από το depot και αρχικοποιούμε τη χωρητικότητα και τον μετρητή εξυπηρετημένων πελατών.

Επόμενο βήμα είναι η επιλογή του πλησιέστερου πελάτη από την αποθήκη, όπου ο πελάτης με τη μικρότερη απόσταση είναι αυτός που θα μπει πρώτος στη διαδρομή. Έπειτα εξετάζουμε τους μη επισκεπτόμενους πελάτες, αναζητώντας τους πλησιέστερους για να τους προσθέσουμε στην υποδιαδρομή, φυσικά βάση των περιορισμών που έχουν καθοριστεί.

Προϋποθέσεις που τίθενται:

- Ελέγχεται αν η συνολική χωρητικότητα του οχήματος (*current_capacity*), εφόσον προστεθεί η ζήτηση του πελάτη που εξετάζεται (*demands[customer]*), υπερβαίνει τη συνολική χωρητικότητα του οχήματος (*vehicle_capacity*)..
- Ελέγχεται αν η συνολική ζήτηση που απομένει (*total_demand - (παρούσα χωρητικότητα + ζήτηση πελάτη)*) είναι μικρότερη από τη ελάχιστη ζήτηση που απαιτείται για κάθε όχημα (*minimum_demand_for_each_vehicle*).
- Τέλος, ελέγχεται αν η προσθήκη του πελάτη θα ξεπεράσει τον αριθμό των πελατών που είναι αναγκαίοι για να εξυπηρετηθούν, δηλαδή αν ο αριθμός των εξυπηρετημένων πελατών (*served_customers_counter + 1*) είναι μεγαλύτερος από το σύνολο των πελατών που μπορούν να εξυπηρετηθούν αφαιρώντας τους ελάχιστους που απαιτούνται ανά όχημα.

```

while unvisited_customers and vehicle_count < num_vehicles:
    sub_tour = [0]
    current_capacity = 0
    served_customers_counter = 0

    if unvisited_customers:
        nearest_customer = min(unvisited_customers, key=lambda customer: distance_matrix[0][customer])
        sub_tour.extend([nearest_customer, 0])
        current_capacity += demands[nearest_customer]
        served_customers_counter += 1
        unvisited_customers.remove(nearest_customer)

    while unvisited_customers and served_customers_counter < max_customers_per_vehicle:
        closest_customer = None
        closest_distance = float('inf')

        for customer in unvisited_customers:
            if (current_capacity + demands[customer] > vehicle_capacity or \
                total_demand - (current_capacity + demands[customer]) < minimum_demand_for_each_vehicle or \
                served_customers_counter + 1 > customers_to_serve - minimum_customers_for_each_vehicle):
                continue

            min_distance = min(distance_matrix[already_served_customer][customer] for already_served_customer in sub_tour[:-1])
            if min_distance < closest_distance:
                closest_distance = min_distance
                closest_customer = customer

        if closest_customer is None:
            break

        best_position = find_best_position(sub_tour, closest_customer)
        sub_tour.insert(best_position, closest_customer)
        current_capacity += demands[closest_customer]
        served_customers_counter += 1
        unvisited_customers.remove(closest_customer)

    solution.append((sub_tour, current_capacity))
    vehicle_count += 1 # Αυξάνουμε τον αριθμό των οχημάτων

return solution

```

Εικόνα 6.5: Δημιουργία διαδικασίας επαναλήψεων εκτέλεσης αλγορίθμου εύρεσης πλησιέστερης εισαγωγής πελατών εντός της υπο-διαδρομής.

Αν κάποιος από αυτούς τους ελέγχους δεν ικανοποιείται, δεν μπορεί να προστεθεί ο πελάτης στη διαδρομή και η εκτέλεση συνεχίζεται με τον επόμενο πελάτη. Στη συνέχεια, εάν έχει βρεθεί ο κατάλληλος πελάτης, εισαγάγετε στην καλύτερη θέση στην υποδιαδρομή του συγκεκριμένου οχήματος.

Η διαδικασία ολοκληρώνεται όταν δεν υπάρχει κάποιος πελάτης ο οποίος δεν έχει εξυπηρετηθεί και όλα τα οχήματα εκτελούν από ένα δρομολόγιο.

7 ΚΕΦΑΛΑΙΟ: ΑΠΟΤΕΛΕΣΜΑΤΑ & ΣΥΜΠΕΡΑΣΜΑΤΑ

Στόχος του κεφαλαίου αυτού είναι η αξιολόγηση της ακρίβειας και της αποτελεσματικότητας των αλγόριθμων δρομολόγησης οχημάτων που υλοποιήθηκαν. Η σύγκριση θα βοηθήσει στον προσδιορισμό της αποτελεσματικότητας που προσφέρει η χρήση των βιβλιοθηκών του OR-TOOLS καθώς και στην αναγνώριση πιθανών περιοχών βελτίωσης στον αλγόριθμο και τη δυνατότητα βελτιστοποίησης των δρομολογίων της εταιρείας. Θα πραγματοποιηθεί μια ερμηνεία βάσει των αποτελεσμάτων που παράγονται από τους κώδικες που αναπτύχθηκαν στα προηγούμενα κεφάλαια και θα εξεταστούν τα αποτελέσματα που δίδονται από τους δυο κώδικες που κατασκευάστηκαν για συγκεκριμένες ημερομηνίες, όπως αυτές ορίζονται από τον χρήστη και σύγκριση των αποτελεσμάτων.

Για τη διαδικασία αυτή, θα πραγματοποιηθεί εκτέλεση του κώδικα για συγκεκριμένες ημερομηνίες που καθορίζονται από τον χρήστη. Κάθε εκτέλεση του κώδικα θα παράγει ένα σύνολο δρομολογίων για τα οχήματα της εταιρείας, λαμβάνοντας υπόψη τους περιορισμούς που έχουν οριστεί, όπως χωρητικότητα οχημάτων, αριθμός πελατών και ειδικοί περιορισμοί.

Η ανάλυση θα επικεντρωθεί σε συγκεκριμένα ερωτήματα που θέτονται όπως:

Ποια είναι η συνολική απόσταση που διανύουν τα οχήματα σύμφωνα με τον κάθε αλγόριθμο;

Πόσοι πελάτες εξυπηρετούνται από κάθε όχημα;

Πώς κατανέμεται το φορτίο στα οχήματα

Με ποια σειρά εξυπηρετούνται οι πελάτες;

Τέλος, θα παρουσιαστούν προτάσεις για μελλοντική έρευνα, πιθανές βελτιώσεις στον αλγόριθμο και στη διαδικασία λήψης αποφάσεων της εταιρείας, με στόχο τη βελτίωση της δρομολόγησης οχημάτων αλλά και του κώδικα που εκτελέστηκε. Αυτή η ερμηνεία και η σύγκριση θα δώσει χρήσιμα συμπεράσματα για το πώς η υλοποίησή του κώδικα μπορεί να χρησιμοποιηθεί στην πράξη και ποιες είναι οι δυνατότητες βελτίωσης των δρομολογίων της εταιρείας.

7.1 ΑΠΟΤΕΛΕΣΜΑΤΑ ΕΡΕΥΝΑΣ

Όσον αφορά τα αποτελέσματα που λάβαμε από τη χρήση των δυο αλγορίθμων που εκτελέστηκαν, αυτά παρουσιάζονται στον πίνακα 7.1, όπου η διαδικασία εκτελέστηκε για διαφορετικές ημερομηνίες και λάβαμε από τον κάθε αλγόριθμο τη μέγιστη απόσταση που καλούνται να διανύσουν τα οχήματα. Οι αποστάσεις που παρουσιάζονται στον πίνακα είναι σε μέτρα, ενώ στην τελευταία στήλη του πίνακα αναγράφεται η διαφορά που παρουσιάστηκε μεταξύ των δυο αλγορίθμων στην εκτέλεσή τους.

Πίνακας 7.1: Σύγκριση αποτελεσμάτων μεταξύ αλγορίθμων που εκτελέστηκαν με Or-TOOLS και Nearest Insertion.

DATE	ΧΡΗΣΗ OR-TOOLS	NEAREST INSERTION ALGORITHM	ΔΙΑΦΟΡΑ
1/5	304342	394463	23%
2/5	286191	333772	14%
3/5	268387	377570	29%
4/5	356032	469642	24%
8/5	288301	383275	25%
9/5	171924	212247	19%
10/5	292769	328456	11%
11/5	324123	448990	28%
13/5	247177	266094	7%
14/5	310617	436005	29%
15/5	285203	350456	19%
16/5	232236	286919	19%
17/5	300575	387508	22%
18/5	294962	399856	26%
20/5	217013	238743	9%
21/5	271935	305525	11%
22/5	295831	395975	25%
23/5	235214	278609	16%
24/5	309695	416907	26%
25/5	335552	447482	25%
27/5	217218	244194	11%
28/5	272416	314387	13%
29/5	288757	378874	24%
30/5	224522	293030	23%
31/5	276925	371503	25%

1/6	347279	411677	16%
3/6	158680	175941	10%
4/6	268262	317323	15%
5/6	289330	392697	26%

Οι αποστάσεις που προκύπτουν από τον κώδικα με χρήση βιβλιοθηκών του OR-TOOLS, όπως φαίνονται και από τον σχετικό πίνακα, είναι γενικά χαμηλότερες σε σύγκριση με αυτές του αλγορίθμου Nearest Insertion. Αυτό υποδεικνύει ότι ο πρώτος είναι πιο αποδοτικός στο να ελαχιστοποιήσει τη συνολική διαδρομή που πρέπει να διανύσουν τα οχήματα. Αυτό φυσικά οφείλεται στο γεγονός ότι κατά την εκτέλεση αυτού του κώδικα, πέρα από την εύρεση αρχικής λύσης μέσω της στρατηγικής Path Cheapest Arc, χρησιμοποιείται επίσης μεθευρετικός αλγόριθμος τοπικής αναζήτησης, συγκεκριμένα ο Guided Local Search. Αντιθέτως, ο Nearest Insertion, αν και είναι απλός και γρήγορος, μπορεί να μην είναι σε θέση να καλύψει τις πιο βέλτιστες διαδρομές λόγω της φύσης του, καθώς εγκλωβίζεται συχνά σε τοπικά ελάχιστα, με αποτέλεσμα να μην εκμεταλλεύεται πλήρως τις ευκαιρίες για βελτίωση των διαδρομών.

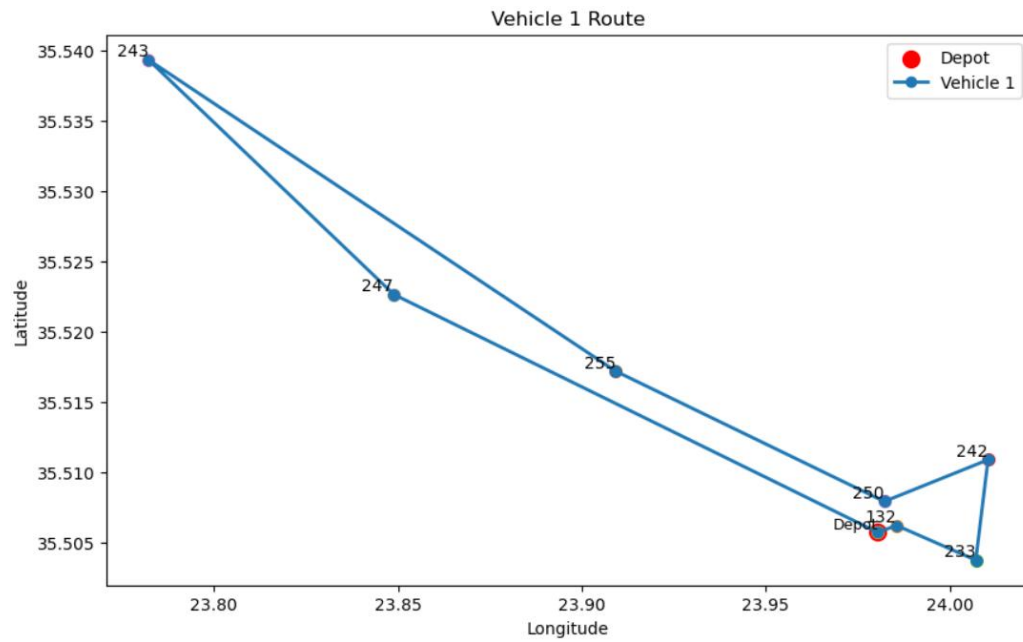
Η ανάλυση των ποσοστών διαφοράς δείχνει ότι ο κώδικας μέσω OR-TOOLS προσφέρει σταθερά καλύτερα αποτελέσματα, ιδίως σε πιο περίπλοκες διαδρομές. Είναι επίσης ενδιαφέρον ότι σε περιπτώσεις μικρότερων αποστάσεων, οι διαφορές που κυμαίνονται γύρω από το 10% θεωρούνται θετικές, καθώς υποδεικνύουν ότι ο Nearest Insertion μπορεί να προσφέρει ικανοποιητικές λύσεις σε λιγότερο περίπλοκες καταστάσεις.

Αυτή η ανάλυση αναδεικνύει τη σημασία της επιλογής του κατάλληλου αλγορίθμου, ανάλογα με την πολυπλοκότητα της κάθε διαδρομής. Για πιο απαιτητικές περιπτώσεις, η “επένδυση” στο OR-TOOLS μπορεί να προσφέρει σημαντικά πλεονεκτήματα, ενώ για πιο απλές καταστάσεις, ο Nearest Insertion παρέχει μια γρήγορη και σχετικά καλή λύση. Η καλή απόδοση του τελευταίου σε μικρότερες αποστάσεις υποδεικνύει ότι αποτελεί μια εναλλακτική επιλογή που μπορεί να χρησιμοποιηθεί σε λιγότερο απαιτητικές συνθήκες, χωρίς να επηρεάζεται σημαντικά η ποιότητα.

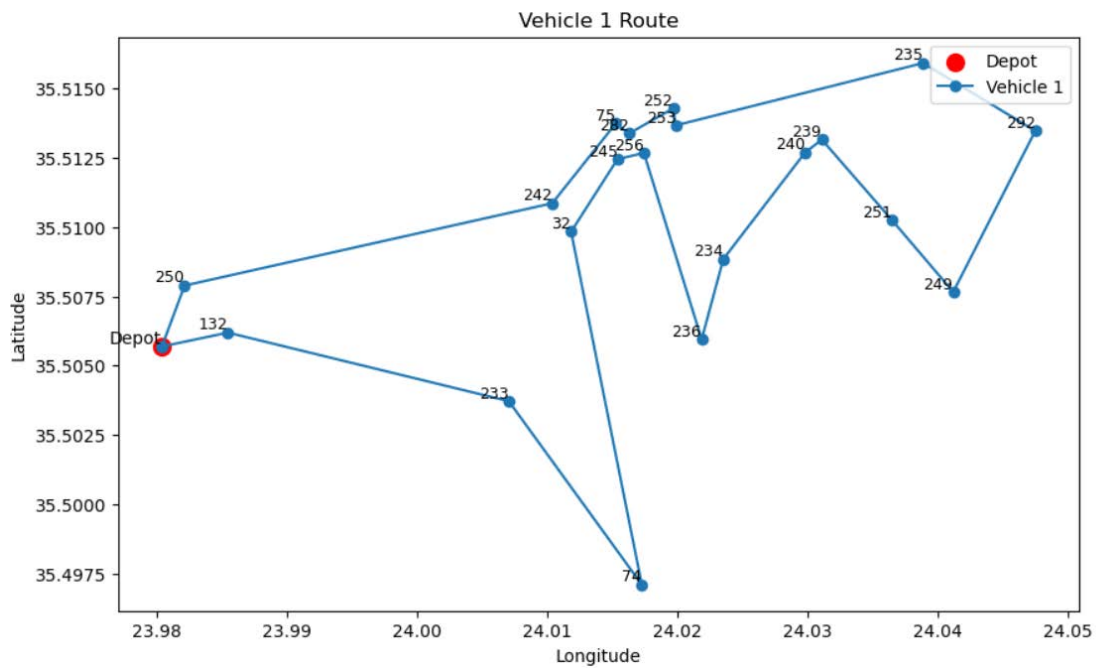
Ακολουθώς παρουσιάζονται τα γραφήματα που παρήχθησαν για μια ενδεικτική ημερομηνία, με plot μέσα στους κώδικες και με χρήση του Badger Maps:

Πίνακας 7.2: Αποτελέσματα δρομολόγησης οχήματος 1 από τους κώδικες.

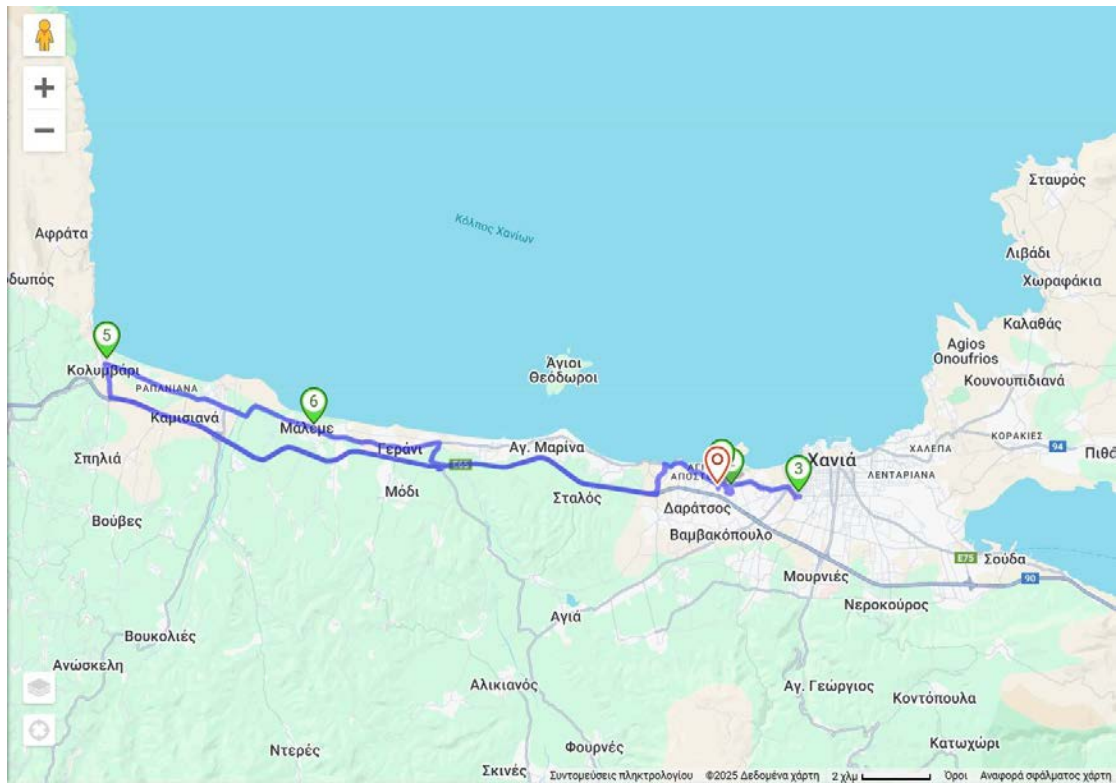
Route for vehicle 1:	
OR-TOOLS	NEAREST INSERTION
0 (CustomerID: Depot) Deliver(0) ->	0 (CustomerID: Depot) Load(118207) ->
33 (CustomerID: 132) Deliver(11251) ->	20 (CustomerID: 250) Deliver(7382) ->
56 (CustomerID: 233) Deliver(7117) ->	15 (CustomerID: 242) Deliver(1226) ->
63 (CustomerID: 242) Deliver(1226) ->	5 (CustomerID: 75) Deliver(26971) ->
68 (CustomerID: 250) Deliver(7382) ->	26 (CustomerID: 282) Deliver(714) ->
72 (CustomerID: 255) Deliver(7067) ->	22 (CustomerID: 252) Deliver(1800) ->
64 (CustomerID: 243) Deliver(4914) ->	23 (CustomerID: 253) Deliver(5248) ->
66 (CustomerID: 247) Deliver(9044) ->	10 (CustomerID: 235) Deliver(3871) ->
0 (CustomerID: Depot) Load(48001)	27 (CustomerID: 292) Deliver(2595) ->
	19 (CustomerID: 249) Deliver(3381) ->
	21 (CustomerID: 251) Deliver(6174) ->
	13 (CustomerID: 239) Deliver(3894) ->
	14 (CustomerID: 240) Deliver(3805) ->
	9 (CustomerID: 234) Deliver(4138) ->
	11 (CustomerID: 236) Deliver(5055) ->
	25 (CustomerID: 256) Deliver(2511) ->
	17 (CustomerID: 245) Deliver(208) ->
	1 (CustomerID: 32) Deliver(2571) ->
	4 (CustomerID: 74) Deliver(18295) ->
	8 (CustomerID: 233) Deliver(7117) ->
	6 (CustomerID: 132) Deliver(11251) ->
	0 (CustomerID: Depot) Load(118207)



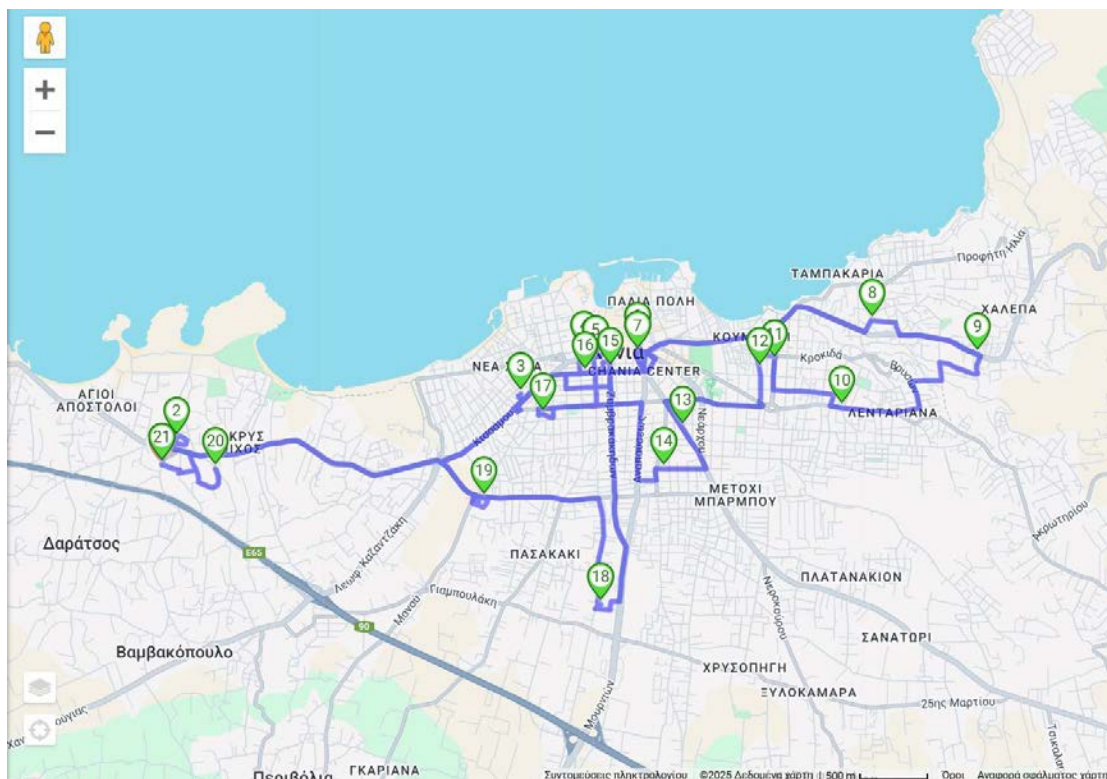
Εικόνα 7.1: Διάγραμμα απεικόνισης δρομολόγησης οχήματος 1 με χρήση OR-TOOLS.



Εικόνα 7.2: Διάγραμμα απεικόνισης δρομολόγησης οχήματος 1 με χρήση Nearest Insertion.



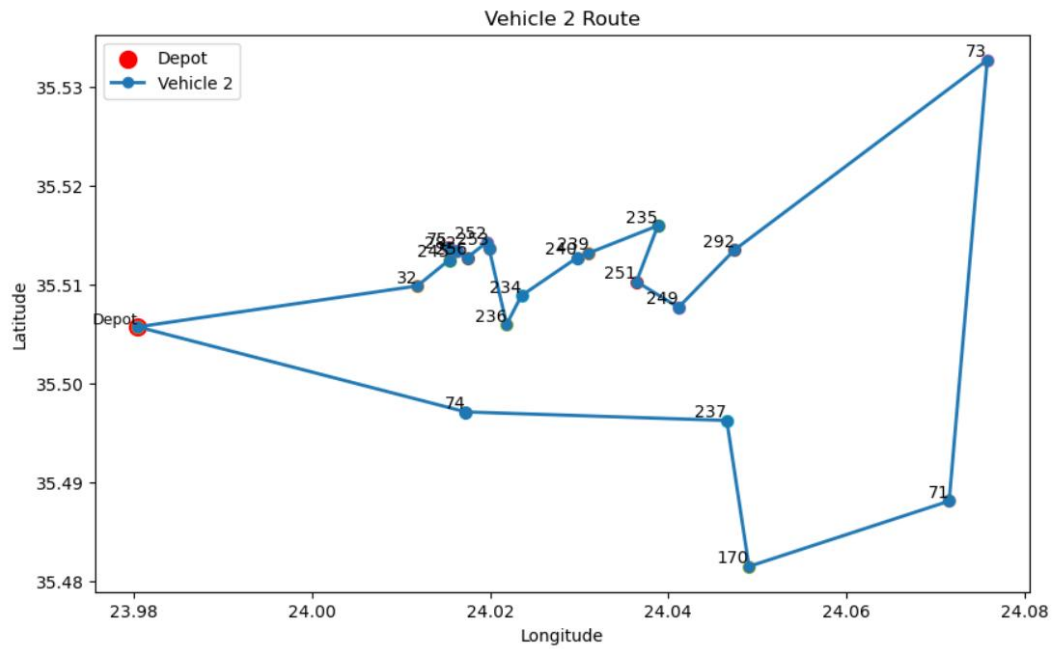
Εικόνα 7.3: Διάγραμμα απεικόνισης δρομολόγησης οχήματος 1 με χρήση OR-TOOLS μέσω Badger Maps.



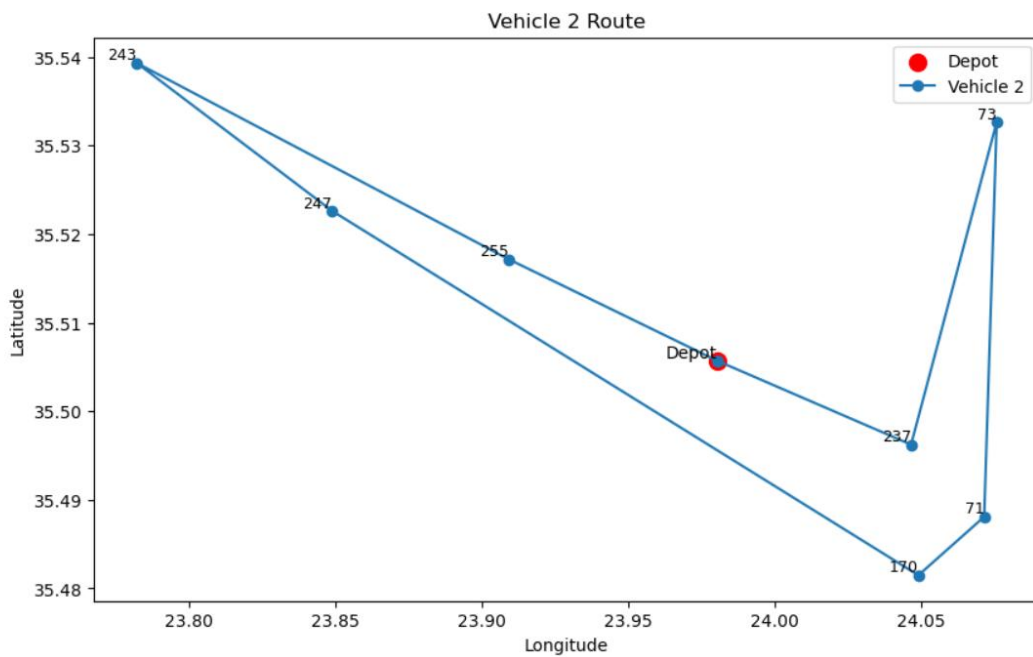
Εικόνα 7.4: Διάγραμμα απεικόνισης δρομολόγησης οχήματος 1 με χρήση Nearest Insertion μέσω Badger Maps.

Πίνακας 7.3: Αποτελέσματα δρομολόγησης οχήματος 2 από τους κώδικες.

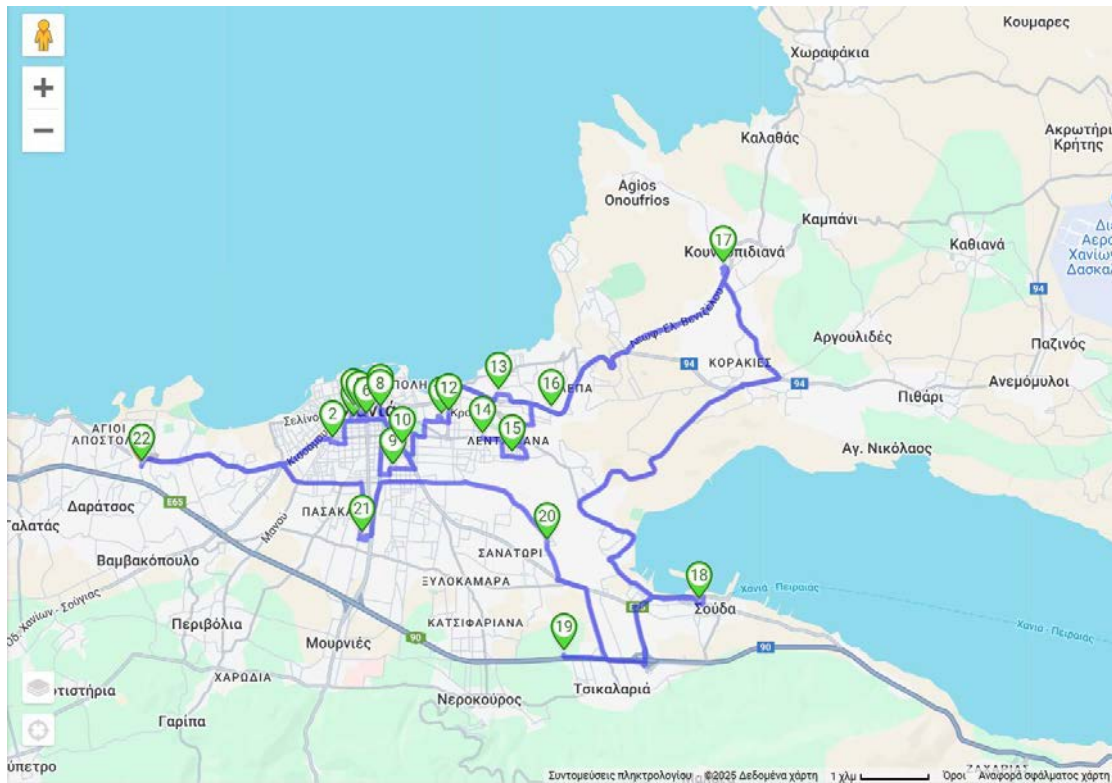
Route for vehicle 2:	
OR-TOOLS	NEAREST INSERTION
0 (CustomerID: Depot) Deliver(0) -> 9 (CustomerID: 32) Deliver(2571) -> 65 (CustomerID: 245) Deliver(208) -> 24 (CustomerID: 75) Deliver(26971) -> 81 (CustomerID: 282) Deliver(714) -> 73 (CustomerID: 256) Deliver(2511) -> 70 (CustomerID: 252) Deliver(1800) -> 71 (CustomerID: 253) Deliver(5248) -> 59 (CustomerID: 236) Deliver(5055) -> 57 (CustomerID: 234) Deliver(4138) -> 62 (CustomerID: 240) Deliver(3805) -> 61 (CustomerID: 239) Deliver(3894) -> 58 (CustomerID: 235) Deliver(3871) -> 69 (CustomerID: 251) Deliver(6174) -> 67 (CustomerID: 249) Deliver(3381) -> 83 (CustomerID: 292) Deliver(2595) -> 22 (CustomerID: 73) Deliver(1181) -> 21 (CustomerID: 71) Deliver(6644) -> 43 (CustomerID: 170) Deliver(24614) -> 60 (CustomerID: 237) Deliver(767) -> 23 (CustomerID: 74) Deliver(18295) -> 0 (CustomerID: Depot) Load(124437)	0 (CustomerID: Depot) Load(54231) -> 24 (CustomerID: 255) Deliver(7067) -> 16 (CustomerID: 243) Deliver(4914) -> 18 (CustomerID: 247) Deliver(9044) -> 7 (CustomerID: 170) Deliver(24614) -> 2 (CustomerID: 71) Deliver(6644) -> 3 (CustomerID: 73) Deliver(1181) -> 12 (CustomerID: 237) Deliver(767) -> 0 (CustomerID: Depot) Load(54231)



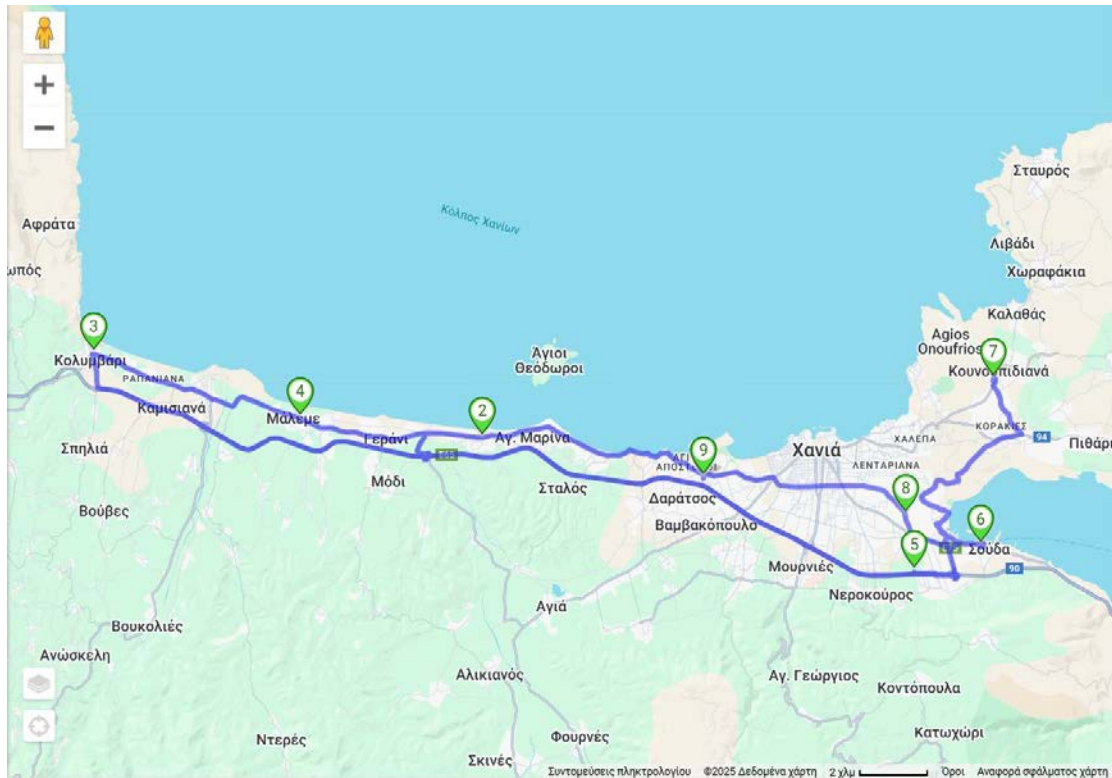
Εικόνα 7.5: Διάγραμμα απεικόνισης δρομολόγησης οχήματος 2 με χρήση OR-TOOLS.



Εικόνα 7.6: Διάγραμμα απεικόνισης δρομολόγησης οχήματος 2 με χρήση Nearest Insertion.



Εικόνα 7.7: Διάγραμμα απεικόνισης δρομολόγησης οχήματος 2 με χρήση OR-TOOLS μέσω Badger Maps.

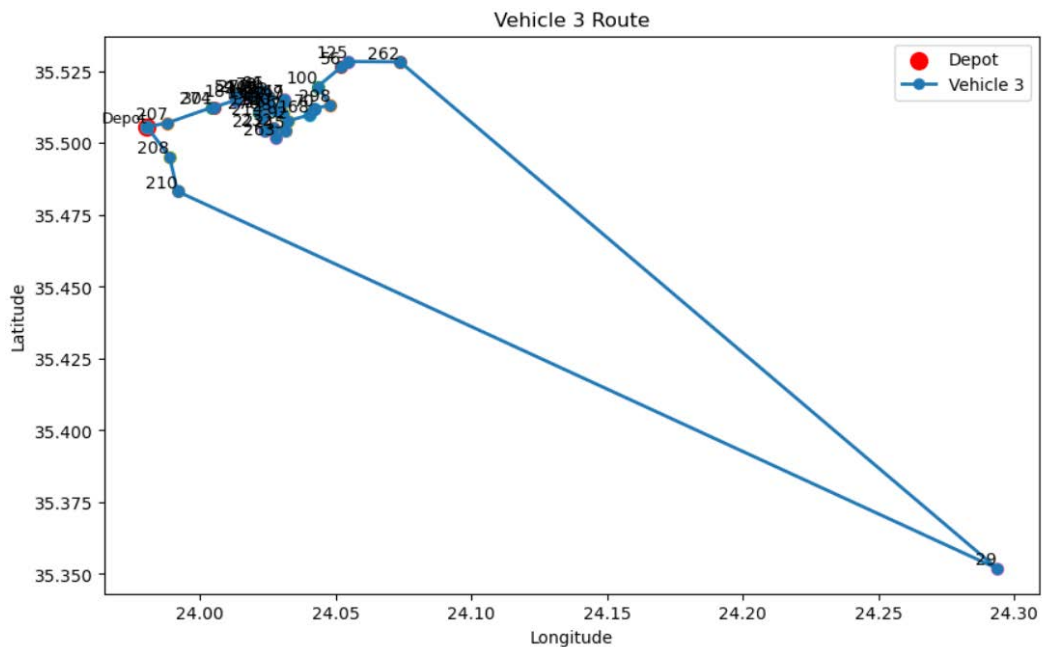


Εικόνα 7.8: Διάγραμμα απεικόνισης δρομολόγησης οχήματος 2 με χρήση Nearest Insertion μέσω Badger Maps.

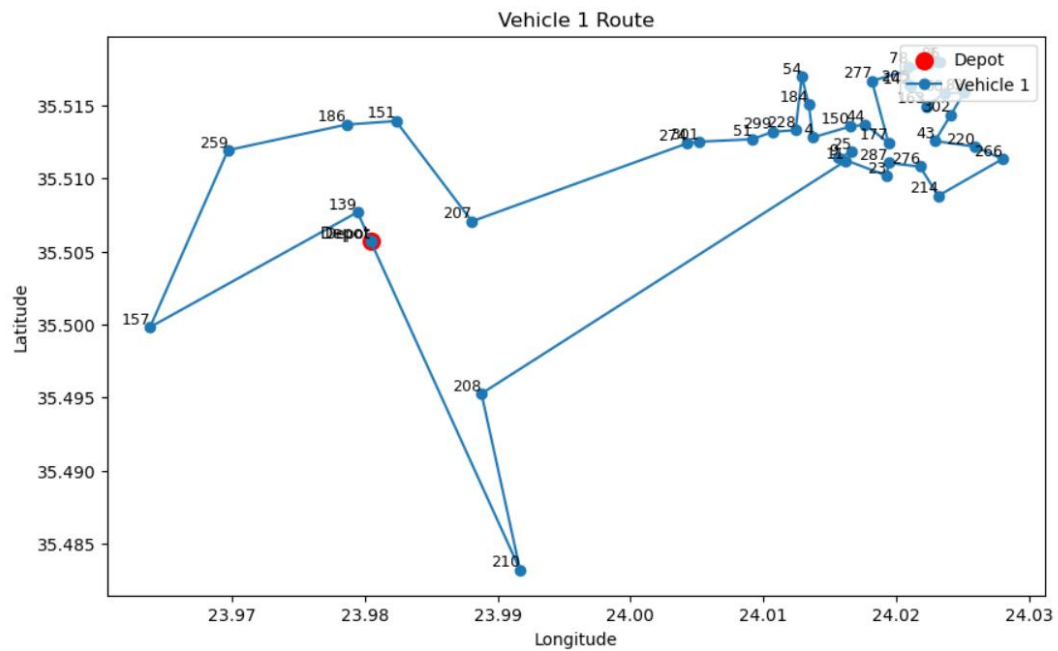
Πίνακας 7.4: Αποτελέσματα δρομολόγησης οχήματος 3 από τους κώδικες.

Route for vehicle 3:	
OR - TOOLS	NEAREST INSERTION
0 (CustomerID: Depot) Deliver(0) ->	0 (CustomerID: Depot) Load(40957) ->
48 (CustomerID: 207) Deliver(3375) ->	29 (CustomerID: 139) Deliver(1622) ->
78 (CustomerID: 274) Deliver(3712) ->	34 (CustomerID: 157) Deliver(1223) ->
86 (CustomerID: 301) Deliver(364) ->	50 (CustomerID: 259) Deliver(895) ->
45 (CustomerID: 184) Deliver(317) ->	40 (CustomerID: 186) Deliver(172) ->
15 (CustomerID: 54) Deliver(218) ->	33 (CustomerID: 151) Deliver(1929) ->
80 (CustomerID: 277) Deliver(372) ->	42 (CustomerID: 207) Deliver(3375) ->
25 (CustomerID: 78) Deliver(1151) ->	54 (CustomerID: 274) Deliver(3712) ->
1 (CustomerID: 1) Deliver(1759) ->	60 (CustomerID: 301) Deliver(364) ->
27 (CustomerID: 86) Deliver(894) ->	14 (CustomerID: 51) Deliver(2101) ->
47 (CustomerID: 205) Deliver(290) ->	59 (CustomerID: 299) Deliver(821) ->
35 (CustomerID: 142) Deliver(1101) ->	48 (CustomerID: 228) Deliver(1622) ->
40 (CustomerID: 163) Deliver(255) ->	15 (CustomerID: 54) Deliver(218) ->
87 (CustomerID: 302) Deliver(982) ->	39 (CustomerID: 184) Deliver(317) ->
17 (CustomerID: 60) Deliver(2001) ->	3 (CustomerID: 4) Deliver(1718) ->
26 (CustomerID: 82) Deliver(329) ->	32 (CustomerID: 150) Deliver(399) ->
18 (CustomerID: 67) Deliver(1905) ->	13 (CustomerID: 44) Deliver(419) ->
36 (CustomerID: 149) Deliver(682) ->	38 (CustomerID: 177) Deliver(698) ->
11 (CustomerID: 37) Deliver(1063) ->	56 (CustomerID: 277) Deliver(372) ->
77 (CustomerID: 266) Deliver(765) ->	23 (CustomerID: 86) Deliver(894) ->
52 (CustomerID: 220) Deliver(1718) ->	2 (CustomerID: 1) Deliver(1759) ->
12 (CustomerID: 43) Deliver(1349) ->	21 (CustomerID: 78) Deliver(1151) ->
79 (CustomerID: 276) Deliver(737) ->	41 (CustomerID: 205) Deliver(290) ->
51 (CustomerID: 214) Deliver(994) ->	30 (CustomerID: 142) Deliver(1101) ->
53 (CustomerID: 222) Deliver(449) ->	35 (CustomerID: 163) Deliver(255) ->
55 (CustomerID: 232) Deliver(999) ->	17 (CustomerID: 60) Deliver(2001) ->

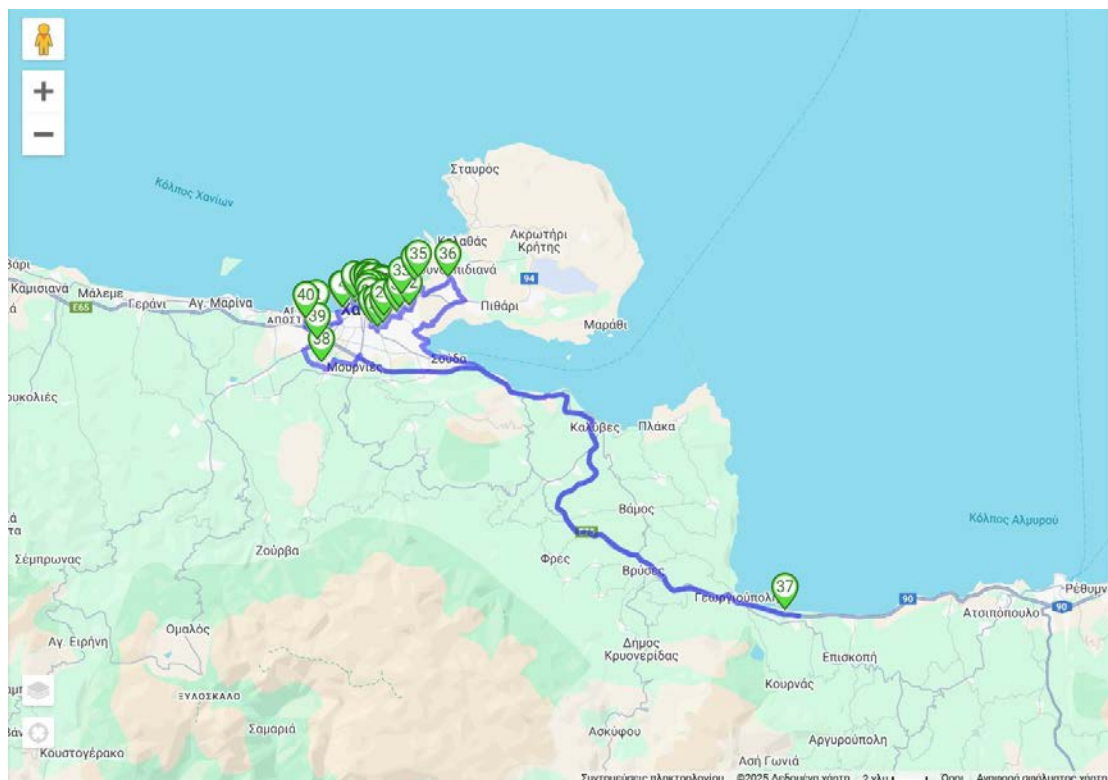
76 (CustomerID: 263) Deliver(558) ->	22 (CustomerID: 82) Deliver(329) ->
5 (CustomerID: 15) Deliver(271) ->	61 (CustomerID: 302) Deliver(982) ->
28 (CustomerID: 92) Deliver(363) ->	12 (CustomerID: 43) Deliver(1349) ->
42 (CustomerID: 168) Deliver(742) ->	46 (CustomerID: 220) Deliver(1718) ->
20 (CustomerID: 70) Deliver(214) ->	53 (CustomerID: 266) Deliver(765) ->
84 (CustomerID: 298) Deliver(990) ->	45 (CustomerID: 214) Deliver(994) ->
30 (CustomerID: 100) Deliver(345) ->	55 (CustomerID: 276) Deliver(737) ->
16 (CustomerID: 56) Deliver(397) ->	57 (CustomerID: 287) Deliver(306) ->
32 (CustomerID: 125) Deliver(580) ->	7 (CustomerID: 23) Deliver(590) ->
75 (CustomerID: 262) Deliver(7) ->	4 (CustomerID: 9) Deliver(1219) ->
8 (CustomerID: 29) Deliver(200) ->	8 (CustomerID: 25) Deliver(645) ->
50 (CustomerID: 210) Deliver(366) ->	5 (CustomerID: 11) Deliver(604) ->
49 (CustomerID: 208) Deliver(925) ->	43 (CustomerID: 208) Deliver(925) ->
0 (CustomerID: Depot) Load(33739)	44 (CustomerID: 210) Deliver(366) ->
	0 (CustomerID: Depot) Load(40957)



Εικόνα 7.9: Διάγραμμα απεικόνισης δρομολόγησης οχήματος 3 με χρήση OR-TOOLS.



Εικόνα 7.10: Διάγραμμα απεικόνισης δρομολόγησης οχήματος 3 με χρήση Nearest Insertion.

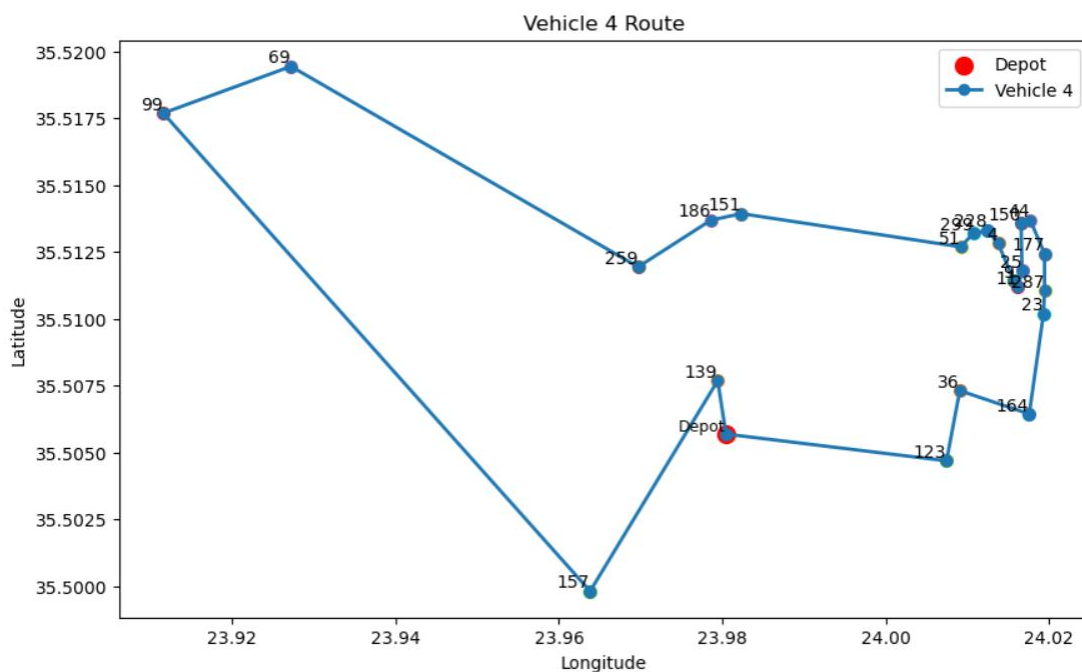


Εικόνα 7.11: Διάγραμμα απεικόνισης δρομολόγησης οχήματος 3 με χρήση OR-TOOLS μέσω Badger Maps.

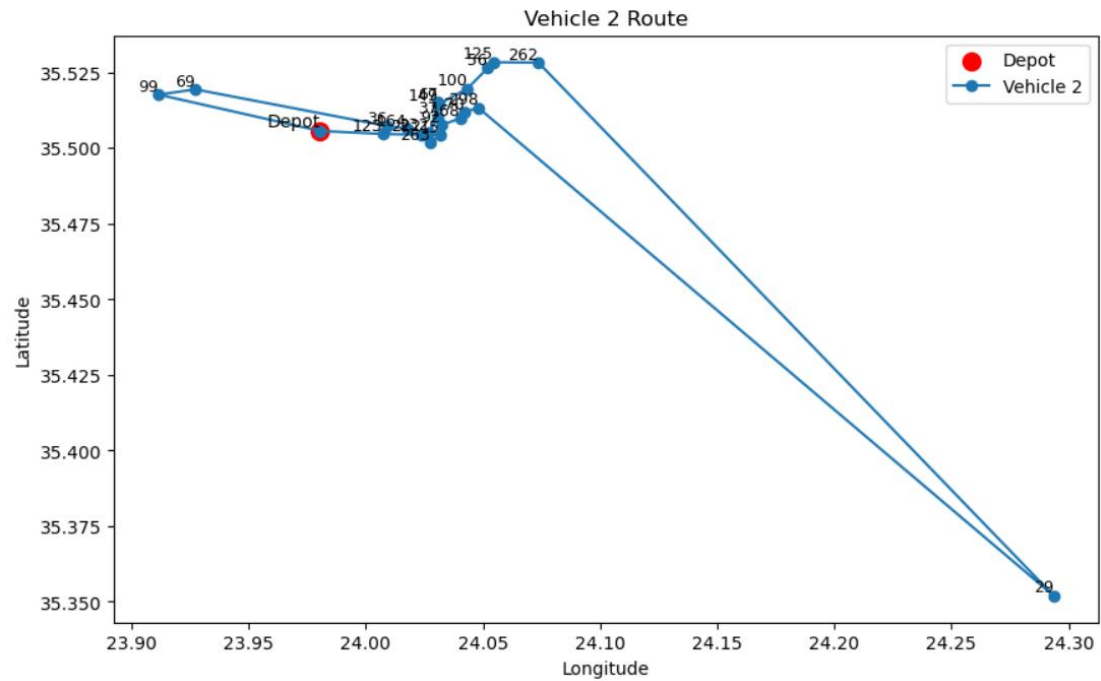


Route for vehicle 4:	
OR - TOOLS	NEAREST INSERTION
0 (CustomerID: Depot) Deliver(0) ->	0 (CustomerID: Depot) Load(13864) ->
34 (CustomerID: 139) Deliver(1622) ->	27 (CustomerID: 123) Deliver(598) ->
39 (CustomerID: 157) Deliver(1223) ->	47 (CustomerID: 222) Deliver(449) ->
29 (CustomerID: 99) Deliver(188) ->	52 (CustomerID: 263) Deliver(558) ->
19 (CustomerID: 69) Deliver(1026) ->	6 (CustomerID: 15) Deliver(271) ->
74 (CustomerID: 259) Deliver(895) ->	24 (CustomerID: 92) Deliver(363) ->
46 (CustomerID: 186) Deliver(172) ->	37 (CustomerID: 168) Deliver(742) ->
38 (CustomerID: 151) Deliver(1929) ->	20 (CustomerID: 70) Deliver(214) ->
14 (CustomerID: 51) Deliver(2101) ->	58 (CustomerID: 298) Deliver(990) ->
85 (CustomerID: 299) Deliver(821) ->	9 (CustomerID: 29) Deliver(200) ->

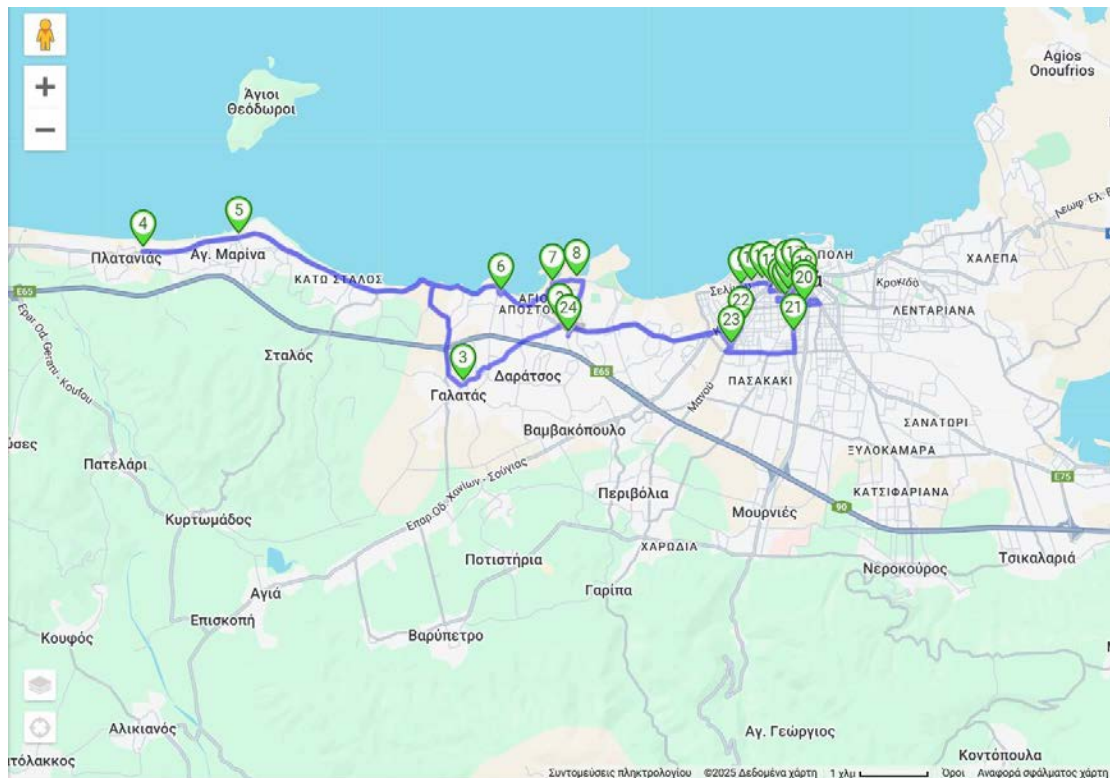
54 (CustomerID: 228) Deliver(1622) ->	51 (CustomerID: 262) Deliver(7) ->
2 (CustomerID: 4) Deliver(1718) ->	28 (CustomerID: 125) Deliver(580) ->
3 (CustomerID: 9) Deliver(1219) ->	16 (CustomerID: 56) Deliver(397) ->
4 (CustomerID: 11) Deliver(604) ->	26 (CustomerID: 100) Deliver(345) ->
7 (CustomerID: 25) Deliver(645) ->	31 (CustomerID: 149) Deliver(682) ->
37 (CustomerID: 150) Deliver(399) ->	18 (CustomerID: 67) Deliver(1905) ->
13 (CustomerID: 44) Deliver(419) ->	11 (CustomerID: 37) Deliver(1063) ->
44 (CustomerID: 177) Deliver(698) ->	49 (CustomerID: 232) Deliver(999) ->
82 (CustomerID: 287) Deliver(306) ->	36 (CustomerID: 164) Deliver(1267) ->
6 (CustomerID: 23) Deliver(590) ->	10 (CustomerID: 36) Deliver(1020) ->
41 (CustomerID: 164) Deliver(1267) ->	19 (CustomerID: 69) Deliver(1026) ->
10 (CustomerID: 36) Deliver(1020) ->	25 (CustomerID: 99) Deliver(188) ->
31 (CustomerID: 123) Deliver(598) ->	0 (CustomerID: Depot) Load(13864)
0 (CustomerID: Depot) Load(21082)	



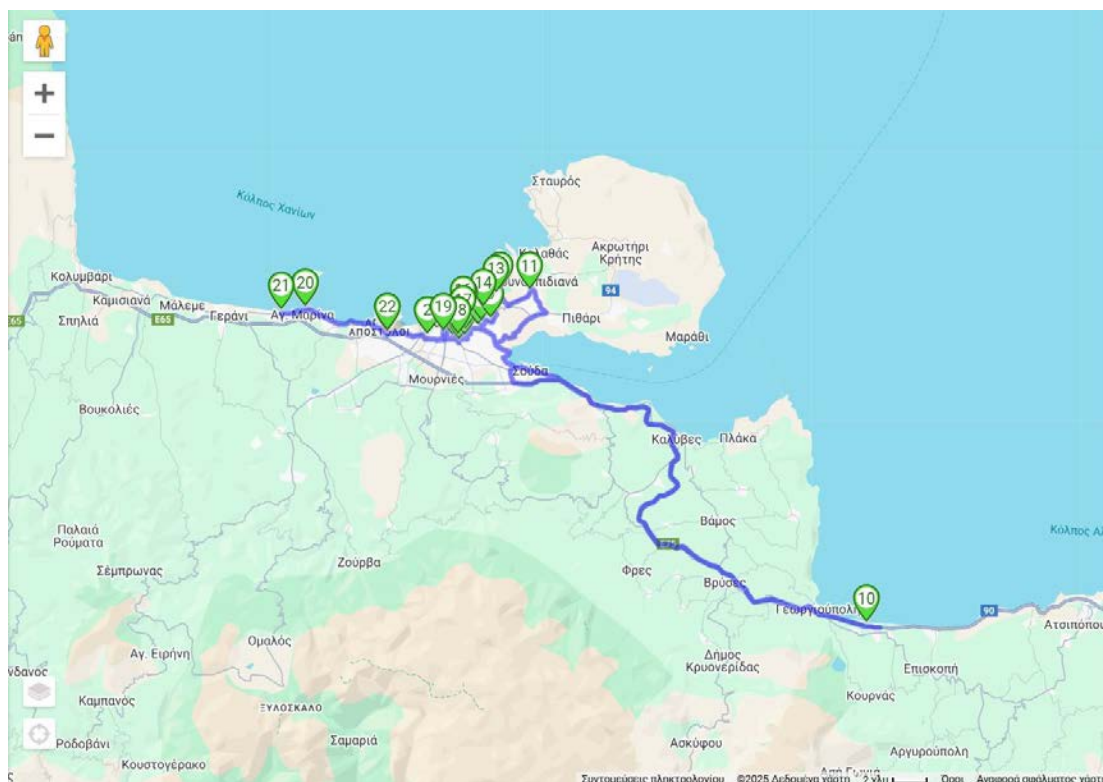
Εικόνα 7.13: Διάγραμμα απεικόνισης δρομολόγησης οχήματος 4 με χρήση OR-TOOLS.



Εικόνα 7.14: Διάγραμμα απεικόνισης δρομολόγησης οχήματος 4 με χρήση Nearest Insertion.



Εικόνα 7.15: Διάγραμμα απεικόνισης δρομολόγησης οχήματος 4 με χρήση OR-TOOLS μέσω Badger Maps.



Εικόνα 7.16: Διάγραμμα απεικόνισης δρομολόγησης οχήματος 4 με χρήση Nearest Insertion μέσω Badger Maps.

7.2 ΠΡΟΤΑΣΕΙΣ ΓΙΑ ΜΕΛΛΟΝΤΙΚΗ ΕΡΕΥΝΑ

Εν συνεχεία της παρούσας εργασίας μπορεί να πραγματοποιηθεί περαιτέρω έρευνα που θα εστιάζει στην επέκταση του μοντέλου για το συγκεκριμένο δίκτυο διανομής, έτσι ώστε να προσαρμοστεί καλύτερα στις απαιτήσεις διανομής των προϊόντων, οι οποίες ποικίλλουν ανάλογα με τα προϊόντα ή τις υπηρεσίες της εταιρίας ως προς τις συνεργαζόμενες εταιρίες. Για να καλυφθούν αυτές οι ανάγκες, απαιτείται η συλλογή μεγάλου όγκου πληροφοριών και δεδομένων.

Φυσικά υπάρχουν πιθανοί τρόποι βελτίωσης και επέκτασης της υφιστάμενης έρευνας, με την προσθήκη επιπλέον περιορισμών και παραμέτρων που θα επιτρέψουν την ανάλυση ενός ευρύτερου φάσματος περιπτώσεων, καθιστώντας τα αποτελέσματα πιο ρεαλιστικά και αξιόπιστα. Μερικές προτάσεις για μελλοντική έρευνα περιλαμβάνουν:

1. Βελτίωση της ακρίβειας των αποτελεσμάτων του εμπορικού λογισμικού μέσω διαχείρισης δελτίων αποστολής

Μια πιθανή βελτίωση των εξαγόμενων αποτελεσμάτων από το εμπορικό λογισμικό θα μπορούσε να επιτευχθεί με την προσαρμογή της διαχείρισης των δελτίων αποστολής. Στην τρέχουσα εργασία, έχει θεωρηθεί ότι κάθε δελτίο αποστολής που εκδίδεται κατά την τρέχουσα ημερομηνία προορίζεται για αποστολή την επόμενη ημέρα. Ωστόσο, υπάρχουν περιπτώσεις όπου τα δελτία εκδίδονται για παράδοση την ίδια ημέρα, με αποτέλεσμα στην τρέχουσα εργασία είτε να φαίνεται ότι χάνονται παραγγελίες, καθώς εμφανίζονται να δρομολογούνται για την επόμενη ημέρα, είτε να προστίθενται παραγγελίες που δεν ανήκουν στο τρέχον δρομολόγιο. Επιπλέον, παραγγελίες που τιμολογούνται και οι πελάτες παραλαμβάνουν απευθείας από την αποθήκη περιλαμβάνονται επίσης στη δρομολόγηση, παρόλο που δεν απαιτούν διανομή.

Εάν το ERP σύστημα είχε τη δυνατότητα εξαγωγής αποτελεσμάτων με βάση την πραγματική ημερομηνία παράδοσης, αντί μόνο της ημερομηνίας έκδοσης του δελτίου αποστολής, τα αποτελέσματα θα ήταν πιο ακριβή και αντιπροσωπευτικά της πραγματικής λειτουργίας. Αυτό θα επέτρεπε καλύτερη παρακολούθηση των παραδόσεων και βελτιστοποίηση του προγραμματισμού της διανομής.

2. Ενσωμάτωση κυκλοφοριακών συνθηκών σε τηλεματικές τεχνολογίες

Οι κυκλοφοριακές συνθήκες που επικρατούν στις περιοχές που πραγματοποιείται η δρομολόγηση είναι ένας από τους σημαντικότερους παράγοντες που επηρεάζουν τη δρομολόγηση, ιδιαίτερα στις αστικές περιοχές. Η ανάπτυξη ενός μοντέλου που εκτιμά τις κυκλοφοριακές συνθήκες και εξετάζει δυναμικά τη δρομολόγηση των οχημάτων, καθώς και εκπαιδεύει ένα μοντέλο κρατώντας ιστορικό βάση προηγούμενων δρομολογήσεων θα μπορούσε να αποτελέσει ουσιαστική επέκταση της υπάρχουσας έρευνας και αντικείμενο περαιτέρω μελέτης. Τα συστήματα τηλεματικής που χρησιμοποιούνται για τη δρομολόγηση του στόλου συνδυάζονται με συστήματα ασύρματης εποπτείας, προσφέροντας ολοκληρωμένες λύσεις στον τομέα των logistics.

Μέσω δεκτών GPS, υπάρχει έλεγχος για παρακολούθηση της θέσης, της κατάστασης και του χρόνου αναμονής καθώς και καταγραφή ενός ιστορικού των εκτελεσμένων διαδρομών

με παράλληλη ενημέρωση των κυκλοφοριακών συνθηκών σε πραγματικό χρόνο καθώς επιτυγχάνεται ασύρματη επικοινωνία των οχημάτων με το σύστημα δρομολόγησης που πραγματοποιείται. Αυτό το δυναμικό σύστημα επιτρέπει την πιο αποδοτική διαχείριση του στόλου, προσαρμόζοντας τις διαδρομές ανάλογα με τις τρέχουσες συνθήκες κυκλοφορίας και συμβάλλει στη βελτίωση των διαδικασιών της δρομολόγησής και διανομής.

3. Ενσωμάτωση χρονικών παραθύρων

Μια πιθανή βελτίωση του αλγορίθμου θα μπορούσε να επιτευχθεί με την ενσωμάτωση περιορισμών βασισμένων σε χρονικά παράθυρα στο σύστημα, όπως ο χρόνος φόρτωσης και εκφόρτωσης, ο χρόνος μετάβασης από πελάτη σε πελάτη, ο χρόνος επιστροφής στην αποθήκη, καθώς και ο χρόνος εξυπηρέτησης του πελάτη, που περιλαμβάνει τον χρόνο ελέγχου κατά την παραλαβή και την πληρωμή για την παραγγελία του. Η ενσωμάτωση αυτών των χρονικών παραθύρων θα επιτρέψει στον αλγόριθμο να προσαρμόσει με καλύτερη ακρίβεια τη δρομολόγηση και τον προγραμματισμό των δραστηριοτήτων, βελτιώνοντας τη συνολική αποδοτικότητα και ακρίβεια της διαδικασίας.

4. Ενσωμάτωση μέγιστης απόστασης διαδρομής για κάθε όχημα

Η προσθήκη ενός περιορισμού που αφορά τη μέγιστη απόσταση που μπορεί να διανύσει κάθε όχημα θα μπορούσε να παρέχει μια βελτίωση στον αλγόριθμο. Αυτός ο περιορισμός μπορεί να βασίζεται είτε στο ωράριο του οδηγού, περιορίζοντας τις αποστάσεις που μπορεί να καλύψει πριν την απαραίτητη ανάπαυση ή εναλλαγή με κάποιον συνοδό, είτε σε άλλους παράγοντες, όπως η χωρητικότητα του οχήματος ή τα όρια της διαδρομής.

5. Green Logistics

Η ανάλυση της σύνδεσης μεταξύ των μεταφορών και των περιβαλλοντικών επιπτώσεων που προκύπτουν εξαιτίας τους μέσω του σχεδιασμού διαδρομών μπορεί να προσφέρει πρακτικές και χρήσιμες προτάσεις για την προώθηση μιας πράσινης στρατηγικής.

Τα καυσάεργια των οχημάτων που κινούνται με συμβατικά ορυκτά καύσιμα και γενικά ο κλάδος των οδικών μεταφορών είναι από τις πιο σημαντικές πηγές εκπομπών NO_x, SO₂ και CO₂. Το πρόβλημα της ρύπανσης της δρομολόγησης (Pollution Routing Problem), έχει εξεταστεί ευρέως στη βιβλιογραφία, όπου η εκτίμηση της κατανάλωσης καυσίμου και των

παραγόμενων εκπομπών CO₂ από κινητές πηγές είναι το κύριο κριτήριο ελαχιστοποίησης [27]. Η έρευνα στον τομέα του Green-VRP επικεντρώνεται στη βελτιστοποίηση της κατανάλωσης ενέργειας στις μεταφορές, αναγνωρίζοντας ότι οι δραστηριότητες εφοδιαστικής, όπως η ανάπτυξη προϊόντων, η διαδικασία παραγωγής, οι μεταφορές και η διαχείριση αποβλήτων, έχουν σημαντικό αντίκτυπο στο περιβάλλον.

6. Reverse Logistics

Μέχρι σήμερα, λίγες προσπάθειες έχουν γίνει για τη συστηματική ανασκόπηση της δρομολόγησης οχημάτων στη διαδικασία αντίστροφης εφοδιαστικής. Οι δραστηριότητες της αντίστροφης εφοδιαστικής απαιτούνται για την εκτέλεση της διαδικασίας συλλογής με το ελάχιστο δυνατό κόστος χωρίς να τίθεται σε κίνδυνο το περιβάλλον και η κοινωνική ζωή.

Το πρόβλημα δρομολόγησης οχημάτων με παραδόσεις και παραλαβές (VRPDP) στο πλαίσιο της αντίστροφης εφοδιαστικής αφορά τον καθορισμό των καλύτερων διαδρομών για την εκτέλεση δύο διαφορετικών υπηρεσιών, την παράδοση προϊόντων σε πελάτες και την παραλαβή επιστρεπτών αντικειμένων από τους ίδιους πελάτες [28]. Σε αρκετά προβλήματα αντίστροφης εφοδιαστικής, απαιτούνται περιορισμοί ελεύθερης χωρητικότητας, είτε για να επιτραπεί η μετακίνηση του οδηγού στο εσωτερικό του οχήματος για την αναδιάταξη των φορτίων, είτε για να αποφευχθεί η ανάμεικτη τοποθέτηση των φορτίων παράδοσης και παραλαβής.

Στην υπό εξέταση μελέτη, θα μπορούσε να προστεθεί το ζήτημα της διαχείρισης των επιστρεφόμενων προϊόντων από τους πελάτες, είτε αυτά είναι σε καλή κατάσταση είτε ληγμένα. Συγκεκριμένα, κατά τη διάρκεια της επίσκεψης του οδηγού σε έναν πελάτη, μπορεί να πραγματοποιηθεί η παραλαβή των επιστρεφόμενων προϊόντων, τα οποία στη συνέχεια επιστρέφονται στην αποθήκη. Τα προϊόντα που είναι σε καλή κατάσταση θα επιστρέφονται στα ράφια της αποθήκης, ενώ τα προϊόντα που προορίζονται για απόρριψη θα ελέγχονται και στη συνέχεια θα πετάγονται, με την αξία τους να πιστώνεται αργότερα στους πελάτες. Η ενσωμάτωση αυτής της διαδικασίας στο μοντέλο δρομολόγησης θα μπορούσε να βελτιώσει την αποδοτικότητα της διαχείρισης επιστροφών και να συμβάλλει στην εφαρμογή πρακτικών Green logistics.

7. Επίλυση μέσω πραγματικών αποστάσεων

Στην επίλυση του προβλήματος που τέθηκε χρησιμοποιήθηκαν αποστάσεις υπολογισμένες βάση της σφαιρικής απόστασης με χρήση του τύπου Haversine. Φυσικά εάν οι αποστάσεις που χρησιμοποιήθηκαν ήταν οι πραγματικές αποστάσεις που διασχίζει κάθε όχημα κατά τη διάρκεια του ταξιδιού από το οδικό δίκτυο του Google maps τα αποτελέσματα που θα εμφανιζόντουσαν θα προσέφεραν μια πιο ρεαλιστική εικόνα του προβλήματος. Η χρήση των πραγματικών αποστάσεων του οδικού δικτύου, όπως χρησιμοποιούνται από το OR-Tools διαφέρει από την ευκλείδεια απόσταση μεταξύ των κόμβων και προσφέρει μια προσέγγιση πιο κοντά στην πραγματική.

Η συγκέντρωση αυτών των δεδομένων μπορεί να οδηγήσει σε πιο ακριβείς εκτιμήσεις της ζήτησης και των χαρακτηριστικών της, κάνοντας τα αποτελέσματα πιο ρεαλιστικά. Η ζήτηση επηρεάζει άμεσα και τις διαδικασίες διαχείρισης των φορτίων. Επιπλέον, η ενσωμάτωση στοιχείων σχετικά με τους χρόνους αναμονής, εισόδου και εξόδου των οχημάτων από και προς τους χώρους φόρτωσης, μπορεί να βοηθήσει στον καθορισμό των μέσων χρόνων που απαιτούνται για αυτές τις διαδικασίες. Έτσι, το μοντέλο θα μπορεί να αποτυπώνει τις πραγματικές συνθήκες με μεγαλύτερη ακρίβεια.

8 ΒΙΒΛΙΟΓΡΑΦΙΑ

- [1]: *SCM definitions and glossary of terms*. SCM Definitions and Glossary of Terms. (n.d.).
https://cscmp.org/CSCMP/CSCMP/Educate/SCM_Definitions_and_Glossary_of_Terms.aspx
- [2]: Ballou, R. H. (2007). The evolution and future of logistics and Supply Chain Management. *European Business Review*, 19(4), 332–348.
<https://doi.org/10.1108/09555340710760152>
- [3]: Sheffi, Y. (n.d.). Jumping the barriers of the logistics function - researchgate.
https://www.researchgate.net/profile/Yossi_Sheffi/publication/266603079_Logistics_at_Large_Jumping_the_Barriers_of_the_Logistics_Function/links/5480910c0cf20f081e725ad7/Logistics-at-Large-Jumping-the-Barriers-of-the-Logistics-Function.pdf
- [4]: Editorial Team. (n.d.-a). *cogoport Global Trade Platform*. <https://www.cogoport.com/en-IN/blogs/transport-cost>
- [5]: *Distribution channels in logistics: Definition, types, and components*. Inbound Logistics. (n.d.). <https://www.inboundlogistics.com/articles/distribution-channels-in-logistics/>
- [6]: Chen, D., & Yang, Z. (2017). Multiple depots vehicle routing problem in the context of total urban traffic equilibrium. *Journal of Advanced Transportation*, 2017, 1–14.
<https://doi.org/10.1155/2017/8524960>
- [7]: Psaraftis, H. N., Wen, M., & Kontovas, C. A. (2016). Dynamic vehicle routing problems: Three decades and counting. *Networks*, 67(1), 3–31. <https://doi.org/10.1002/net.21628>,
<https://doi.org/10.1002/net.21628>
- [8]: Dantzig, G. B., & Ramser, J. H. (1959). The truck dispatching problem. *Management Science*, 6(1), 80–91. <https://doi.org/10.1287/mnsc.6.1.80>
- [9]: Ιωάννης Μαρινάκης, Μαγδαληνή Μαρινάκη, Αθανάσιος Μυγδαλάς (2019), *Προβλήματα Δρομολόγησης Οχημάτων στη Διαχείριση της Εφοδιαστικής Αλυσίδας*
- [10]: Kallehauge, B., Larsen, J., Madsen, O. B. G., & Solomon, M. M. (n.d.). Vehicle routing problem with time windows. *Column Generation*, 67–98. https://doi.org/10.1007/0-387-25486-2_3

- [11]: Mirabi, M., Fatemi Ghomi, S. M. T., & Jolai, F. (2010). Efficient stochastic hybrid heuristics for the multi-depot vehicle routing problem. *Robotics and Computer-Integrated Manufacturing*, 26(6), 564–569. <https://doi.org/10.1016/j.rcim.2010.06.023>
- [12]: Goetschalckx, M., & Jacobs-Blecha, C. (1989). The vehicle routing problem with backhauls. *European Journal of Operational Research*, 42(1), 39–51. [https://doi.org/10.1016/0377-2217\(89\)90057-x](https://doi.org/10.1016/0377-2217(89)90057-x)
- [13]: Sapti Wahyuningsih, Darmawan S/ (2015) The Characteristics Study *THE CHARACTERISTICS STUDY OF SOLVING VARIANTS OF VEHICLE ROUTING PROBLEM AND ITS APPLICATION ON DISTRIBUTION PROBLEM*
- [14]: Zobolas, G. I., Tarantilis, C. D., & Ioannou, G. (2008). Exact, heuristic and meta-heuristic algorithms for solving shop scheduling problems. *Metaheuristics for Scheduling in Industrial and Manufacturing Applications*, 1–40. https://doi.org/10.1007/978-3-540-78985-7_1
- [15]: Daniel Delahaye, Supatcha Chaimatanan, Marcel Mongeau. *Simulated annealing: From basics to applications*. Gendreau, Michel; Potvin, Jean-Yves. Handbook of Metaheuristics, 272, Springer, pp.1- 35.ISBN 978-3-319-91085-7, 2019, International Series in Operations Research & Management Science (ISOR), 978-3-319-91086-4. [ff10.1007/978-3-319-91086-4_1](https://doi.org/10.1007/978-3-319-91086-4_1)ff. fffhal-01887543
- [16]: *Simulated annealing*. Simulated Annealing - an overview | ScienceDirect Topics. (n.d.). <https://www.sciencedirect.com/topics/computer-science/simulated-annealing>
- [17]: Pal, A., LaGrow, T., & Commenter, A. W. (n.d.). *Home*. OMSCS 7641 Machine Learning. <https://sites.gatech.edu/omscs7641/2024/02/19/simulated-annealing-methods-and-real-world-applications/>
- [18]: Glover, F., Laguna, M., & Martí, R. (2018). Principles and strategies of Tabu Search. *Handbook of Approximation Algorithms and Metaheuristics, Second Edition*, 361–377. <https://doi.org/10.1201/9781351236423-21>
- [19]: Baker, B. M., & Ayechew, M. A. (2003). A genetic algorithm for the vehicle routing problem. *Computers & Operations Research*, 30(5), 787–800. [https://doi.org/10.1016/s0305-0548\(02\)00051-5](https://doi.org/10.1016/s0305-0548(02)00051-5)

- [20]: Çatay, B. (2009). Ant colony optimization and its application to the vehicle routing problem with pickups and deliveries. *Studies in Computational Intelligence*, 219–244. https://doi.org/10.1007/978-3-642-04039-9_9
- [21]: *Overview of constraint programming*. Overview of constraint programming - IBM® Decision Optimization CPLEX® Modeling for Python (DOcplex) V2.25 documentation. (n.d.). <https://ibmdecisionoptimization.github.io/docplex-doc/cp.html>
- [22]: Google. (n.d.). *Solving an LP problem | or-tools | google for developers*. Google. https://developers.google.com/optimization/lp/lp_example
- [23]: Voudouris, C., Tsang, E. P. K., & Alsheddy, A. (2010). Guided Local Search. *International Series in Operations Research & Management Science*, 321–361. https://doi.org/10.1007/978-1-4419-1665-5_11
- [24]: Määtä, P. (2021, September 23). Comparison of solution methods for the dynamic pickup and delivery problem with time windows. https://sal.aalto.fi/publications/pdf-files/theses/mas/tmaa21_public.pdf
- [25]: Boussaïd, I., Lepagnot, J., & Siarry, P. (2013). A survey on optimization metaheuristics. *Information Sciences*, 237, 82–117. <https://doi.org/10.1016/j.ins.2013.02.041>
- [26]: Chris Veness, www.movable-type.co.uk. (n.d.). *Movable type scripts*. Calculate distance and bearing between two Latitude/Longitude points using haversine formula in JavaScript. <https://www.movable-type.co.uk/scripts/latlong.html>
- [27]: Asghari, M., & Mirzapour Al-e-hashem, S. M. (2021). Green Vehicle Routing Problem: A state-of-the-art review. *International Journal of Production Economics*, 231, 107899. <https://doi.org/10.1016/j.ijpe.2020.107899>
- [28]: Santos, M. J., Jorge, D., Ramos, T., & Barbosa-Póvoa, A. (2023). Green Reverse Logistics: Exploring the vehicle routing problem with deliveries and pickups. *Omega*, 118, 102864. <https://doi.org/10.1016/j.omega.2023.102864>
- [29]: Rosenkrantz, D.J., R.E. Stearns and P.M. Lewis, "An Analysis of Several Heuristics for the Traveling Salesman Problem", *SIAM Journal of Computing*, 6 (1977)pp. 563-581.