ΠΟΛΥΤΕΧΝΕΙΟ ΚΡΗΤΗΣ

ΣΧΟΛΗ ΜΗΧΑΝΙΚΩΝ ΠΑΡΑΓΩΓΗΣ ΚΑΙ ΔΙΟΙΚΗΣΗΣ

Μεταπτυχιακή εργασία

Μεθευρετικός Αλγόριθμος για την επίλυση του Δυναμικού Προβλήματος Δρομολόγησης Οχημάτων

ΟΝΟΜΑΤΕΠΩΝΥΜΟ: Ταξίδου Ανδρομάχη

Επιβλέπων: Δρ. Ιωάννης Μαρινάκης, Επίκουρος καθηγητής

Χανιά 2017

Η τριμελής εξεταστική επιτροπή

Ιωάννης Μαρινάκης, επίκουρος καθηγητής σχολής Μηχανικών Παραγωγής και Διοίκησης, Πολυτεχνείο Κρήτης (επιβλέπων)

Αριστομένης Αντωνιάδης, καθηγητής σχολής Μηχανικών Παραγωγής και Διοίκησης, Πολυτεχνείο Κρήτης

Γεώργιος Ε. Σταυρουλάκης καθηγητής σχολής Μηχανικών Παραγωγής και Διοίκησης, Πολυτεχνείο Κρήτης

Ευχαριστίες

Θεωρώ καθήκον μου να εκφράσω την ευγνωμοσύνη και τις θερμές μου ευχαριστίες σε όλους όσοι συνέβαλαν στην εκπόνηση της παρούσης μελέτης.

Ανεκτίμητη ήταν η συμβολή του Επιβλέποντα Επίκουρου Καθηγητή κ. Γιάννη Μαρινάκη, και της κ. Μαγδαλινής Μαρινάκη για την εμπιστοσύνη που μου έδειξαν για τον σχεδιασμό και την ανάθεση της μελέτης, για την επιστημονική καθοδήγηση, την αμέριστη βοήθεια και ηθική συμπαράσταση σε όλη τη διάρκεια της μελέτης, καθώς και τα μέλη της Τριμελούς Εξεταστικής Επιτροπής.

Επιπλέον θα ήθελα να ευχαριστήσω τον υποψήφιο διδάκτωρ Ιατρικής φυσικής Παναγιώτη-Εμμανουήλ Καρτσίδη, την Βαχανελίδου Αιμιλία, και τον Στεφανόπουλο Λέανδρο, την Τραχανατζή Δήμητρα, τη Ραπανάκη Εμμανουέλλα, τον Ρηγάκη Μανούσο και τον Τσακιράκη Λευτέρη για την ουσιαστική βοήθεια και υποστήριξη σε όλη τη διάρκεια της διπλωματικής μου εργασίας.

Τέλος θα ήθελα να ευχαριστήσω τον άντρα μου και την οικογένεια μου, για τη αμέριστη ηθική τους συμπαράσταση, την αγάπη τους και την υπομονή τους.

1 ΠΕΡΙΕΧΟΜΕΝΑ

ΠΕΡΙΕΧΟΜΕΝΑ ΕΙΚΟΝΩΝ

Περίληψη

Λόγω της ανταγωνιστικότητας που υπάρχει στην αγορά, οι ιδιωτικές και δημόσιες επιχειρήσεις θέτουν πολλαπλούς στόχους ώστε να αποκομίσουν το μέγιστο δυνατό κέρδος. Σε πολλές περιπτώσεις όμως, κάποιοι από αυτούς τους στόχους είναι αντικρουόμενοι. Η περιπλοκότητα στη λήψη αποφάσεων σε αυτά τα συστήματα αυξάνεται καθώς η λήψη των αποφάσεων αυτών γίνεται σε περιβάλλον αβεβαιότητας και οι παράμετροι που πρέπει να συνυπολογιστούν είναι άγνωστες.

Το πρόβλημα τη δρομολόγησης οχημάτων (Vehicle Routing Problem) είναι ένα από τα σημαντικότερα προβλήματα της συνδυαστικής βελτιστοποίησης και της διαχείρισης της εφοδιαστικής αλυσίδας. Το πρόβλημα δρομολόγησης οχημάτων παρουσιάζει ένα μεγάλο εύρος επεκτάσεων που οφείλεται στις παρά πολλές εφαρμογές του σε πραγματικά προβλήματα. Για κάθε ένα πρόβλημα, εισάγονται και νέοι περιορισμοί, ώστε οι λύσεις που προκύπτουν να ανταποκρίνονται στο εκάστοτε πρόβλημα.

Η βέλτιστη λύση του προβλήματος της δρομολόγησης οχημάτων δύναται να βρεθεί μέσω ευρετικών και μεθευρετικών αλγορίθμων.

Στην παρούσα εργασία επιλύθηκε αρχικά το στατικό πρόβλημα δρομολόγησης οχημάτων (Vehicle Routing Problem – VRP) και στη συνέχεια επιλύσαμε το δυναμικό VRP μετατρέποντας τη λύση του στατικού σε λύση του δυναμικού, και έτσι έγινε η επίλυση του σε πραγματικό χρόνο μέσω ενός απλού γρήγορου και αποτελεσματικού αλγορίθμου.

Η επίλυση του στατικού προβλήματος πραγματοποιήθηκε με τη χρήση του αλγορίθμου ''πλησιέστερος γείτονας'' όπου προέκυψε η αρχική λύση του προβλήματος. Στη συνέχεια χρησιμοποιώντας τον μεθευρετικό αλγόριθμο της προσομοιωμένης ανόπτησης με τη χρήση εσωτερικά των αλγορίθμων τοπικής αναζήτησης 2-opt, 1-1 ανταλλαγή (1-1 exchange), και 1-0

επανατοποθέτηση (1-0 relocate) πραγματοποιήθηκε βελτίωση της αρχικής λύσης-κόστους.

Summary

Due to market competition, private and public companies set multiple goals to achieve the greatest possible profit. In many cases, however, some of these goals are contradictory. The complexity behind decision making in this type of systems is increasing as making these decisions takes place in an environment of uncertainty, and the parameters that have to be taken into consideration are unknown.

The Vehicle routing problem is one of the most important problems of combinational optimization and of supply chain management. The vehicle routing problem has a large scale of variants due to the fact that it is applied in many real problems. New restrictions are entered for every problem, so the solutions that are provided can respond to each problem.

A very good solution of the vehicle routing problem can be found using heuristic and meta-heuristic algorithms. In this thesis we used the meta-heuristic algorithm of simulated annealing, the heuristic algorithms nearest neighbor and the local search algorithms: 2-opt, 1-1 exchange, και 1-0 relocate. Finally we carried out the modeling of the dynamic problem of vehicle routing.

It is possible to find a near optimal solution to the vehicle routing problem through heuristic and meta-heuristic algorithms. In this thesis, we have solved a static VRP, which was then converted to a dynamic VRP, by way of solving it with an efficient and fast algorithm in real time. Solving the static VRP was accomplished by using the nearest-neighbor algorithm, which gave us the initial solution. Following this, using the Simulated Annealling metaheuristic algorithm and utilizing the following local search algorithms: 2-opt, 1-1 exchange and 1-0 relocate, the initial solution and the cost it incurred was improved.

1 ΚΕΦΑΛΑΙΟ 1 : Εισαγωγή

1.1 Logistics

Η παγκοσμιοποίηση της παραγωγής και του εμπορίου έχουν επιφέρει πολλά οφέλη αλλά και πολλές προκλήσεις. Οι εταιρίες στο παρελθόν εξυπηρετούσαν μόνο τοπικές αγορές ενώ τώρα δίνεται η δυνατότητα τα προϊόντα τους να φτάνουν σε πελάτες που βρίσκονται σε πολύ μακρινή απόσταση. Ως αποτέλεσμα, έχει αυξηθεί η περιπλοκότητα και η αβεβαιότητα του δικτύου διανομής. Το έργο της διαχείρισης και του συντονισμού του παγκόσμιου δικτύου φυσικών αγαθών και ροής πληροφοριών έχει γίνει πρώτη προτεραιότητα των επιχειρήσεων, καθώς προσπαθούν να παραμείνουν ανταγωνιστικοί στην συνεχώς μεταβαλλόμενη αγορά. Ως συνέπεια, η ανάγκη για αποτελεσματικότερη διαχείριση της εφοδιαστικής αλυσίδας, είναι μεγαλύτερη από ποτέ [1].

Ο όρος logistics μεταφράζεται σε Εφοδιαστική. Ο όρος αυτός προέρχεται από το “λογιστική” , που σήμερα στην Ελλάδα κυρίως αναφέρεται στην επιστήμη που ασχολείται με τη διαχείριση οικονομικών στοιχείων, δηλαδή δεν έχει άμεση σχέση με τον όρο ‘’logistics’’[2].

Η απλούστερη περιγραφή του όρου “logistics’’ είναι η διαδικασία παράδοσης ενός προϊόντος ή υπηρεσίας σε έναν πελάτη σε συγκεκριμένες ποσότητες, σε καθορισμένο χρόνο και χώρο. Η αυξανόμενη τάση της αποκέντρωσης της παραγωγής έφερε την επέκταση της εφοδιαστικής, η οποία στη σύγχρονη εποχή είναι παρούσα σε πολλούς τομείς της κοινωνίας [3].

1.2 Συστήματα υποστήριξης λήψης αποφάσεων (ΣΥΑ)

Λόγω της ανταγωνιστικότητας που υπάρχει στην αγορά, οι ιδιωτικές και δημόσιες επιχειρήσεις έχουν πολλαπλούς στόχους ώστε να αποκομίσουν το μέγιστο δυνατό κέρδος. Σε πολλές περιπτώσεις όμως, κάποιοι από αυτούς τους στόχους είναι αντικρουόμενοι. Η περιπλοκότητα στη λήψη αποφάσεων σε αυτά τα συστήματα αυξάνεται καθώς η λήψη των αποφάσεων αυτών γίνεται σε περιβάλλον αβεβαιότητας και οι παράμετροι που πρέπει να συνυπολογιστούν είναι άγνωστες.

Η υλοποίηση ενός κατάλληλου ΣΥΑ δύναται να είναι η λύση στα περίπλοκα επιχειρησιακά προβλήματα. Το ΣΥΑ είναι ένα πληροφοριακό σύστημα που υποστηρίζει τη λύση προβλημάτων που αντιμετωπίζουν οντότητες που διαχειρίζονται μια η περισσότερες διαδικασίες ή λειτουργίες. Τις περισσότερες φορές το ΣΥΑ αλλάζει τον τρόπο που οργανώνεται η επιχείρηση και ενεργοποιεί νέες διαδικασίες διαχείρισης. Το βασικό πλεονέκτημα ενός ΣΥΑ είναι ότι βοηθούν στη λήψη αποφάσεων διερευνώντας αποτελεσματικά τον χώρο όλων των αποτελεσματικών λύσεων του προβλήματος [2].

1.3 Εφοδιαστική αλυσίδα (Supply Chain Management)

Παρά την αυξανόμενη δημοτικότητα του όρου ‘’supply chain management’’, τόσο στον ακαδημαϊκό χώρο όσο και στην πράξη, εξακολουθεί να υπάρχει μεγάλη σύγχυση ως προς το νόημα της. Ο όρος πρωτοαναφέρθηκε από συμβούλους στις αρχές της δεκαετίας του 1980. Ακολούθως κέρδισε τεράστια προσοχή. Μέχρι τις αρχές της δεκαετίας του 1990 οι ακαδημαϊκοί προσπαθούσαν να δώσουν μια ολοκληρωμένη ερμηνεία στον όρο SCM. Μέχρι και σήμερα ο όρος logistics συγχέεται με τον όρο SCM. Το συμβούλιο Logistics Management (CLM) έχει ορίσει για τον όρο ''logistics'' ότι αντιπροσωπεύει τον προσανατολισμό της εφοδιαστικής αλυσίδας από το σημείο της παραγωγής μέχρι το σημείο της κατανάλωσης. Ο αναθεωρημένος ορισμός: Logistics είναι το κομμάτι της διαχείρισης της εφοδιαστικής αλυσίδας, το οποίο υλοποιεί και ελέγχει την απόδοση, την αποτελεσματική ροή και αποθήκευση αγαθών, υπηρεσιών και σχετικών πληροφοριών από το σημείο της παραγωγής στο σημείο της κατανάλωσης με στόχο να πληροί τις απαιτήσεις των καταναλωτών. Η διαχείριση της εφοδιαστικής αλυσίδας είναι μια περίπλοκη διαδικασία, η οποία ακόμα και όταν διαχειρίζεται από Logistics αποτελεί μια πρόκληση [4] [5].

Οι Scott and Westbrook (1991) και New and Payne (1995) περιγράφουν την εφοδιαστική αλυσίδα, ως την αλυσίδα που συνδέει κάθε στοιχείο της διαδικασίας παραγωγής και προμήθειας από τις πρώτες ύλες μέχρι τον τελικό χρήστη και περιλαμβάνει και πολλά οργανωτικά όρια [6].

Ο στόχος μιας επιτυχημένης εφοδιαστικής αλυσίδας είναι:

α) Μείωση του κόστους

Ελαχιστοποιώντας τα κόστη που προκύπτουν από όλη τη διαδικασία, πχ μεταφορικό κόστος ή απευθείας αποστολή των προϊόντων λόγω μη ύπαρξης αποθήκης (δεν υπάρχει το κόστος αποθήκευσης).

β) Βελτίωση της εξυπηρέτησης των πελατών

Η βελτίωση των παροχών στους πελάτες, κυρίως σε σχέση με τις προσφερόμενες υπηρεσίες από τους ανταγωνιστές, μπορούν να αποφέρουν το μέγιστο κέρδος που αποσκοπεί η εταιρία.

Εικόνα 1: Διάγραμμα εφοδιαστικής αλυσίδας

1.4 Δίκτυα διανομής

Ο στόχος των επιχειρήσεων για βέλτιστο σχεδιασμό δικτύων διανομής ώστε να πετύχουν σημαντική μείωση στο κόστος, αλλά και καλύτερη εξυπηρέτηση πελατών μπορεί να επιτευχθεί μέσα από τη λύση των προβλημάτων δικτύων διανομής. Το βασικότερο πρόβλημα είναι η διανομή προϊόντων από το σημείο της παραγωγής ή αποθήκευσης τους μέχρι τον τελικό καταναλωτή. Τα προβλήματα δικτύων διανομής καλείται να λύσει ο τομέας της επιχειρησιακής έρευνας.

Η περιγραφή του βασικότερου προβλήματος είναι η εξής. Τα δεδομένα του προβλήματος είναι κόμβοι-πελάτες που βρίσκονται σε γνωστές γεωγραφικά διεσπαρμένες τοποθεσίες και ενός κόμβου αφετηρίας-αποθήκης σε γνωστή θέση. Όλοι οι κόμβοι πρέπει βρίσκονται σε τέτοια θέση ώστε να είναι εφικτά τα δρομολόγια οχημάτων μεταξύ τους. Τα δρομολόγια θα πρέπει να σχεδιαστούν με τέτοιο τρόπο ώστε να ικανοποιούν τη ζήτηση που έχει ο κάθε κόμβος-πελάτης. Σε κάποια προβλήματα που αφορούν στην δρομολόγηση οχημάτων δύναται να υπάρχει περιορισμός στη χωρητικότητα των οχημάτων, στο μήκος και στον χρόνο της διαδρομής (ώρες εργασίας). Βέβαια μπορεί να υπάρξουν και άλλοι πιο ειδικευμένοι περιορισμοί ανάλογα με το εκάστοτε πρόβλημα.

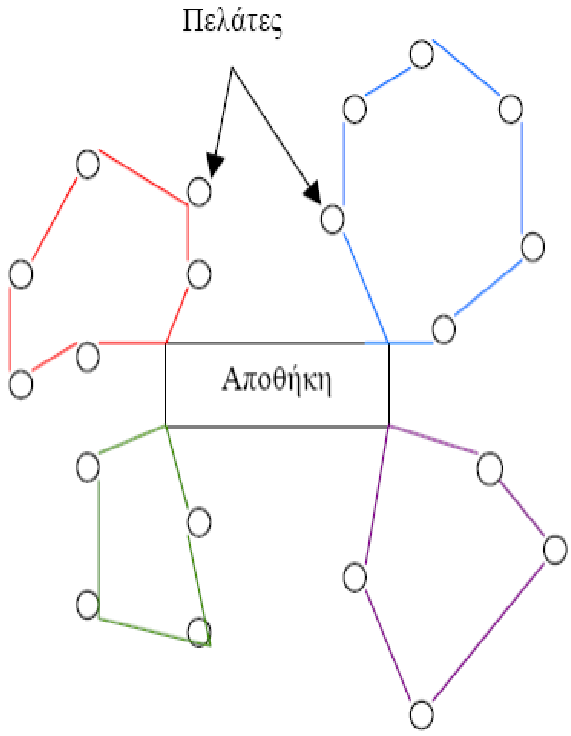
Σε γενικές γραμμές η λύση των προβλημάτων αυτών αποτελείται από μία αντικειμενική συνάρτηση η οποία θα πρέπει να βελτιστοποιηθεί. Τα προβλήματα αυτά παρουσιάζουν τόσο επιστημονικό όσο και πρακτικό ενδιαφέρον. Έχουν προταθεί πολλές νέες μεθοδολογίες και αποτελούν έναν πολλά υποσχόμενο τομέα. Οι επιχειρήσεις οι οποίες χρησιμοποιούν πληροφοριακά συστήματα που αντιμετωπίζουν τέτοιου είδους προβλήματα έχουν τη δυνατότητα να γνωρίζουν ανά πάσα στιγμή σημαντικές πληροφορίες, όπως τη ζήτηση των πελατών τους, τις ιδιαίτερες προτιμήσεις τους, τα προϊόντα που διακινούνται , όπως και πληροφορίες που σχετίζονται με τα οχήματα διανομής. Σε πολλές περιπτώσεις μάλιστα δύναται να εντοπίζεται η ακριβής θέση του οχήματος σε πραγματικό χρόνο, των συνθηκών που γίνεται η μεταφορά και η παράδοση ακόμα και η επικοινωνία με τα οχήματα για εκ νέου οδηγίες, για παράδειγμα αν μια παραγγελία ακυρώθηκε, ή έγινε προσθήκη μιας καινούργιας κτλ (δυναμικό πρόβλημα) [2].

2 ΚΕΦΑΛΑΙΟ 2: Πρόβλημα δρομολόγησης οχημάτων-Vehicle Routing Problem

2.1 Εισαγωγή

Το πρόβλημα της δρομολόγησης οχημάτων (VRP) είναι ένα από τα σημαντικότερα προβλήματα διανομής της συνδυαστικής βελτιστοποίησης. Παρουσιάστηκε πρώτη φορά από τους Dantzig και Ramser [7]. Είναι μια επέκταση του προβλήματος ‘’πλανόδιος πωλητής’’ το οποίο περιλαμβάνει ένα δίκτυο κόμβων (πόλεις) και στόχος είναι να βρεθεί μόνο μια διαδρομή που να συνδέει τις πόλεις με τέτοιο τρόπο ώστε να ελαχιστοποιείται η απόσταση που έχει να διανύσει το όχημα. Ο πωλητής επισκέπτεται κάθε πόλη-κόμβο ακριβώς μία φορά. Στο συγκεκριμένο πρόβλημα είναι δύσκολο να επιτευχθεί η βέλτιστη λύση.

Στο πρόβλημα της δρομολόγησης οχημάτων τα δεδομένα είναι τα εξής. Υπάρχουν n διαθέσιμα οχήματα, τα οποία έχουν συγκεκριμένη χωρητικότητα, και προϊόντα που πρέπει να διανεμηθούν σε κόμβους-πελάτες από έναν αρχικό κόμβο αφετηρία-αποθήκη. Οι αποστάσεις μεταξύ των κόμβων είναι γνωστές. Τα οχήματα αφού ταξιδεύουν σε όλο το δίκτυο αφήνοντας απόθεμα και στο τέλος γυρίζουν σε έναν κόμβο αποθήκη. Το όχημα επισκέπτεται κάθε πελάτη που βρίσκεται στο δίκτυο ακριβώς μια φορά. Οι πελάτες συνήθως βρίσκονται στους κόμβους του δικτύου. Ο στόχος του προβλήματος είναι να βρεθούν όλες οι κυκλικές διαδρομές που ξεκινούν και καταλήγουν στην αφετηρία με το ελάχιστο κόστος, όπως για παράδειγμα η ελαχιστοποίηση του μήκους της διαδρομής [8].



Εικόνα 2: Κυκλικές διαδρομές που προκύπτουν από τη λύση του προβλήματος της δρομολόγησης οχημάτων

Το πρόβλημα δρομολόγησης οχημάτων παρουσιάζει ένα μεγάλο εύρος προεκτάσεων που οφείλεται στις παρά πολλές εφαρμογές του σε πραγματικά προβλήματα. Για κάθε ένα πρόβλημα, κάθε φορά εισάγονται και νέοι περιορισμοί, ώστε οι λύσεις που προκύπτουν να ανταποκρίνονται στο εκάστοτε πρόβλημα.

Οι στόχοι του προβλήματος της δρομολόγησης οχημάτων ποικίλουν και εξαρτώνται από το ζητούμενο του κάθε προβλήματος. Οι κυριότεροι στόχοι είναι:

α) Ελαχιστοποίηση συνολικού κόστους διαδικασίας μεταφοράς αγαθών ή υπηρεσιών. Σε αυτή την περίπτωση το κόστος εξαρτάται από την συνολική απόσταση που έχει διανύσει το όχημα ή από τη συνολική διάρκεια της διαδικασίας. Πολλές φορές συμπεριλαμβάνονται στο συνολικό κόστος και το κόστος για την χρήση των οχημάτων και ο μισθός των οδηγών.

β) Ελαχιστοποίηση του συνολικού αριθμού των οχημάτων του στόλου, άρα και προσωπικού.

γ) Ελαχιστοποίηση των ποινών-προστίμων που προκύπτουν από τη μερική εξυπηρέτηση των πελατών.

2.2 Περιορισμοί του προβλήματος

Πολλές φορές δύναται να υπάρχουν κάποιοι περιορισμοί στον σχεδιασμό του προβλήματος, ώστε κάθε εταιρία να πετύχει το επίπεδο εξυπηρέτησης που επιθυμεί:

α) Μπορεί το κάθε όχημα να έχει διαφορετική χωρητικότητα, όπως και τη δυνατότητα για μεταφορά διαφορετικών ειδών προϊόντων.

β) Μπορεί να υπάρχουν οχήματα τα οποία έχουν διαφορετικό κόστος για την επιχείρηση (όπως για παράδειγμα ιδιόκτητα φορτηγά ή φορτηγά με μισθωτούς οδηγούς).

γ) Μπορεί να υπάρχουν περιορισμοί που αφορούν στο μέγιστο επιτρεπτό μήκος διαδρομής, στο μέγιστο επιτρεπτό φορτίο ή στην μέγιστη ώρα που πραγματοποιείται το δρομολόγιο.

δ) Μπορεί να υπάρχει περιορισμός που αφορά το ποσοστό των διαθέσιμων φορτηγών που θα χρησιμοποιηθούν [2].

Το πρόβλημα της δρομολόγησης οχημάτων παρουσιάζει πολλές εφαρμογές σε πραγματικά προβλήματα. Για κάθε τέτοιο πρόβλημα υπάρχουν διαφορετικοί περιορισμοί τέτοιοι ώστε οι σχετικές λύσεις που θα προκύψουν θα ανταποκρίνονται στις επιθυμητές κατασκευές των διαδρομών. Επειδή υπάρχουν πολλές παραλλαγές προβλημάτων VRP, το πρώτο βήμα για την επίλυση του είναι η αποτελεσματική διαμόρφωση του, που μαζί με έναν ισχυρό αλγόριθμο, συνήθως μεθευρετικό, δίνει την ευκαιρία για την εύρεση σχεδόν μιας βέλτιστης λύσης, χωρίς μεγάλο υπολογιστικό κόστος [9].

2.3 Στατικό και δυναμικό πρόβλημα δρομολόγησης οχημάτων

Στα στατικά και στα αιτιοκρατικά προβλήματα, τα δεδομένα που εισάγονται είναι γνωστά εκ των προτέρων. Ως εκ τούτου οι διαδρομές των οχημάτων δεν αλλάζουν όσο αυτές βρίσκονται σε εξέλιξη.

Στα στατικά και στοχαστικά προβλήματα, τα δεδομένα που εισάγονται είναι μερικώς γνωστές ως τυχαίες μεταβλητές, οι οποίες ‘’αποκαλύπτονται ‘’ κατά τη διάρκεια εκτέλεσης της διαδρομής. Οι διαδρομές των οχημάτων σχεδιάζονται a priori και μόνο [7] [10].

Στα δυναμικά και αιτιοκρατικά προβλήματα, ένα μέρος ή το σύνολο των δεδομένων εισόδου είναι άγνωστα και αποκαλύπτονται μόνο κατά τον σχεδιασμό ή την εκτέλεση των δρομολογίων. Σε αυτήν την περίπτωση, οι διαδρομές επαναπροσδιορίζονται συνεχώς, απαιτώντας τεχνολογική υποστήριξη για επικοινωνία σε πραγματικό χρόνο μεταξύ οχημάτων ή και οχημάτων και κέντρου διαχείρισης δρομολογίων [11].

Επιπλέον στο δυναμικό πρόβλημα οι αλγόριθμοι που το υλοποιούν πρέπει να παρέχουν γρήγορη ανταπόκριση ώστε οι τροποποιήσεις στις διαδρομές να διαβιβάζονται εγκαίρως στον στόλο [12], διότι σε αντίθετη περίπτωση θα υπάρξει κενός χρόνος, που μπορεί να μεταφράζεται σε αύξηση του κόστους.

2.4 Δυναμικό πρόβλημα δρομολόγησης οχημάτων

Η πρώτη αναφορά στο δυναμικό πρόβλημα δρομολόγησης οχημάτων έγινε από τους Wilson και Colvin. Αρχικά χρησιμοποιήθηκαν ευρετικοί αλγόριθμοι οι οποίοι απαιτούν χαμηλή υπολογιστική προσπάθεια. Η τεχνολογική εξέλιξη έχει οδηγήσει στον πολλαπλασιασμό των εφαρμογών δρομολόγησης οχημάτων σε πραγματικό χρόνο. Για παράδειγμα η εισαγωγή του Global Positioning System (GPS), η ανάπτυξη και η ευρεία χρήση των έξυπνων τηλεφώνων σε συνδυασμό με τα ακριβή γεωγραφικά συστήματα πληροφοριών, οι εταιρίες είναι σε θέση όχι μόνο να παρακολουθούν, αλλά και να διαχειρίζονται τον στόλο τους σε πραγματικό χρόνο. Η δυναμική δρομολόγηση οχημάτων δίνει την ευκαιρία για μείωση λειτουργικού κόστους, βελτίωση εξυπηρέτησης πελατών και για μειωμένη περιβαλλοντολογική επίπτωση [7].

Το δυναμικό πρόβλημα μπορεί να διαιρεθεί στις επόμενες κατηγορίες:

α) Many-to-many όπου κάθε αίτημα περιλαμβάνει τοποθεσίες παραλαβής και παράδοσης σε αντίθεση με το one-to-many (ή many-to-one) όπου κάθε αίτημα περιλαμβάνει μια τοποθεσία παραλαβής (παράδοσης).

β) Κάθε όχημα δύναται να έχει σταθερή χωρητικότητα ή μεταβλητή [13].

2.5 Διαφορά στατικού - δυναμικού προβλήματος δρομολόγησης οχημάτων

Στο στατικό πρόβλημα δρομολόγησης οχημάτων το τυπικό κριτήριο αποτίμησης μιας λύσης είναι η ελαχιστοποίηση του κόστους. Στο δυναμικό πρόβλημα όμως, υπάρχει αυξημένη αβεβαιότητα και η αντικειμενική συνάρτηση δεν είναι πάντα προφανής.

Στα δυναμικά προβλήματα δύναται να περιλαμβάνονται:

- Κόστος δρομολόγησης (όπως κόστος φορτηγών, προσωπικού, συντήρησης, λειτουργικό κόστος, κτλ).
- Κέρδος. Στο δυναμικό πρόβλημα μπορεί ένα ποσοστό των πελατών να μην εξυπηρετούνται (μπορεί να εμφανιστεί η βέλτιστη λύση όπου ένας πελάτης θεωρείται απομακρυσμένος και δεν εξυπηρετείται). Αντίθετα στο στατικό πρόβλημα, όλοι οι πελάτες εξυπηρετούνται. Γενικά το ενδεχόμενο να μην εξυπηρετηθεί ένας πελάτης, μπορεί να επιφέρει

μείωση στο κέρδος και εκφράζεται τις περισσότερες φορές με τη μορφή προστίμου.

- Χρόνος παραμονής στο δίκτυο. Πολλές φορές υπάρχει περιορισμός στον μέγιστο επιτρεπόμενο χρόνο μιας κυκλικής διαδρομής [14].

Το δυναμικό πρόβλημα δρομολόγησης οχημάτων σε σχέση με το αντίστοιχο στατικό, παρουσιάζει αυξημένη περιπλοκότητα και πολυπλοκότητα στη λήψη αποφάσεων και εισάγει νέες προκλήσεις. Για παράδειγμα στις ταχυμεταφορές μια εταιρία μπορεί να απορρίψει το αίτημα ενός πελάτη, είτε επειδή το κόστος εξυπηρέτησης είναι πολύ υψηλό, είτε γιατί είναι αδύνατο να τον εξυπηρετήσουν.

Στη δυναμική δρομολόγηση, η δυνατότητα να ανακατευθυνθεί ένα κινούμενο όχημα σε ένα νέο αίτημα πελάτη, με στόχο το μικρότερο κόστος, απαιτεί τη γνώση της θέσης του οχήματος σε πραγματικό χρόνο ώστε να υπάρχει επικοινωνία με τους οδηγούς. Στο στατικό πρόβλημα κάτι τέτοιο είναι αδύνατο να συμβεί.

Ενώ και τα δυο προβλήματα έχουν κοινό στόχο την ελαχιστοποίηση του κόστους, στο δυναμικό πρόβλημα μπορεί να γίνει εισαγωγή και άλλων εννοιών όπως το επίπεδο υπηρεσιών ή η μεγιστοποίηση των εσόδων [7].

2.6 Τεχνικές επίλυσης ενός VRP

Σχεδόν το σύνολο των τεχνικών επίλυσης προβλημάτων δρομολόγησης οχημάτων αποτελούν ευρετικές και μεθευρετικές μέθοδοι. Κανένας ακριβής αλγόριθμος δεν μπορεί να εγγυηθεί την εύρεση της βέλτιστης λύσης. Αυτό οφείλεται στο γεγονός ότι το πρόβλημα δρομολόγησης οχημάτων αποτελεί ένα NP-hard πρόβλημα.

Οι ευρετικοί αλγόριθμοι, χαρακτηρίζονται από χαμηλή υπολογιστική πολυπλοκότητα και δίνουν τη δυνατότητα για περιορισμένη εξερεύνηση του χώρου των εφικτών λύσεων. Η κυριότερη μέθοδος επίλυσης του προβλήματος με ευρετική τεχνική είναι η κατασκευαστική μέθοδος. Η εφικτή λύση δημιουργείται σταδιακά με ταυτόχρονη παρατήρηση στη μεταβολή του κόστους. Δεν πραγματοποιείται βελτιστοποίηση της λύσης σε κάθε φάση της διαδικασίας.

Οι μεθευρετικοί αλγόριθμοι παράγουν πολύ καλύτερες λύσεις σε σχέση με τους ευρετικούς. Υπόσχονται την σε βάθος εξερεύνηση των υποπεριοχών του χώρου των εφικτών λύσεων του προβλήματος.

Η λογική που χρησιμοποιούν οι μεθευρετικοί αλγόριθμοι είναι οι εξής:

1. Εύρεση μιας αρχικής λύσης Α
2. Υπολογισμός του κόστους της Α
3. Κατασκευή μιας νέας λύσης Β, στη γειτονιά της Α
4. Υπολογισμός κόστους της Β
5. Σύγκριση κόστους. Η λύση που έχει το μικρότερο κόστος θεωρείται τώρα η αρχική λύση του προβλήματος
6. Επανάληψη στο από το βήμα 3
7. Τέλος [15]

2.7 Μεθευρετικοί αλγόριθμοι επίλυσης ενός VRP

Οι κυριότεροι μεθευρετικοί αλγόριθμοι είναι οι εξής:

a) Tabu search (TS)

Ο Tabu search είναι ένας αλγόριθμος τοπικής αναζήτησης. Εξερευνά τον χώρο των λύσεων με την μετακίνηση σε κάθε επανάληψη από μια λύση S, σε μία καλύτερη λύση στην γειτονιά της S, με τη χρήση τριών διαφορετικών μνημών με στόχο να θυμάται καλές και κακές λύσεις που είχε επισκεφθεί ο αλγόριθμος κατά τη διάρκεια των εκτελέσεων ώστε να κινηθεί ταχύτερα προς τη σωστή κατεύθυνση βελτίωσης της λύσης. Η τρέχουσα λύση μπορεί να είναι χειρότερη από τη μία επανάληψη στην επόμενη [16].

b) Αλγόριθμος της Προσομοιωμένης Ανόπτησης (Simulated Annealing (SA))

Αποτελεί μια μέθοδο τυχαιοποιημένης τοπικής αναζήτησης όπου η αύξηση του κόστους μιας λύσης μπορεί να γίνει δεκτή με μία πιθανότητα. Προσομοιώνει την ψύξη ενός υλικού σε θερμό νερό, με αποτέλεσμα να προκύπτουν οι ελάχιστες δυνατές ενεργειακές καταστάσεις των μορίων του υλικού. Κατά τη διάρκεια της διαδικασίας, δηλαδή καθώς γίνονται οι επαναλήψεις του αλγορίθμου, η θερμοκρασία μειώνεται με αργό ρυθμό. Αυτή είναι η διαδικασία της ανόπτησης [17].

Η ανόπτηση περιλαμβάνει τη θέρμανση και την ψύξη ενός υλικού (μέταλλο) για να αλλάξει τις φυσικές του ιδιότητες λόγω των αλλαγών στην εσωτερική του δομή. Καθώς το μέταλλο ψύχεται, η νέα δομή που προκύπτει είναι πιο σταθερή από την πρότερη κατάσταση που βρισκόταν με αποτέλεσμα το μέταλλο να διατηρεί τις νέες του ιδιότητες. Στην διαδικασία της προσομοιωμένης ανόπτησης θεωρούμε τη θερμοκρασία ως μια μεταβλητή, ώστε να πραγματοποιηθεί η διαδικασία της θέρμανσης.

Αρχικά ορίζουμε μια υψηλή θερμοκρασία και στη συνέχεια αφήνουμε να πέσει η θερμοκρασία, μέσω της συνάρτησης μείωσης της θερμοκρασίας, αργά καθώς ο αλγόριθμος τρέχει. Καθώς ο αλγόριθμος εξελίσσεται ενδέχεται κάποιες λύσεις να είναι χειρότερες από την αρχική μας λύση, οι οποίες γίνονται δεκτές με μια μικρή πιθανότητα, επιτρέποντας στον αλγόριθμο να εστιάζει σταδιακά σε μία περιοχή του χώρου αναζήτησης. Η σταδιακή διαδικασία ψύξης είναι το χαρακτηριστικό του αλγορίθμου που συνεισφέρει στην εύρεση ακόμα και της βέλτιστης λύσης.

c) Γενετικοί Αλγόριθμοι (GA)

Οι γενετικοί αλγόριθμοι αποτελούν μία μέθοδο προσαρμοστικής ευρετικής αναζήτησης, που μιμείται την εξέλιξη μέσω φυσικής επιλογής. Χρησιμοποιούν την ιδέα της εξέλιξης, μέσω της γενετικής μετάλλαξης, της φυσικής επιλογής και της διασταύρωσης [18]. Βασίζονται στην βιολογική διαδικασία κατά την οποία οι νέοι και καλύτεροι πληθυσμοί των διαφορετικών ειδών αναπτύσσονται κατά τη διάρκεια της εξέλιξης.

Παρουσιάζουν αρκετά πλεονεκτήματα σε σχέση με άλλες μεθόδους. Τα βασικότερα είναι τα εξής:

- Μπορούν να χρησιμοποιηθούν σε προβλήματα με διακριτές και συνεχείς μεταβλητές
- Δύναται να πραγματοποιηθεί ταυτόχρονη αναζήτηση σε ένα μεγάλο μέρος του χώρου των λύσεων
- Παρουσιάζουν συντομία στη λύση του προβλήματος
- Δύναται να προκύψουν παραπάνω από μία λύση και να ξεφύγουν από τοπικό ελάχιστο.

d) Βελτιστοποίηση Αποικίας Μυρμηγκιών (Ant Colony Optimization)

Ο αλγόριθμος αυτός βασίζεται στην επικοινωνία των μυρμηγκιών μέσω χημικών ουσιών, της φερομόνης, κατά την αναζήτηση συλλογής τροφής [19]. Πραγματοποιείται τυχαία αναζήτηση γύρω από την πηγή αφήνοντας πίσω τους φερομόνη ώστε να αναγνωρίσουν πιο μονοπάτι ακολούθησαν και επιπλέον να αναγνωρίσει το επόμενο μυρμήγκι το μονοπάτι αυτό. Όσο αυξάνεται η ποσότητα της φερομόνης, τόσο περισσότερα μυρμήγκια περνάνε από αυτό το μονοπάτι. Με την πάροδο της ώρας από τα μονοπάτια που δεν περνάνε πολλά μυρμήγκια, η ποσότητας της φερομόνης μειώνεται ώσπου μόνο σε ένα μονοπάτι υπάρχει φερομόνη και όλα τα μυρμήγκια ακολουθούν αυτό.

e) Αλγόριθμος Βελτιστοποίησης Σμήνος Σωματιδίων (PSO)

Αποτελεί μια υπολογιστική μέθοδο που βελτιστοποιεί ένα πρόβλημα με επαναληπτικό τρόπο, ώστε να βελτιωθεί μία αρχική λύση σε σχέση με ένα μέτρο ποιότητας [19]. Προσομοιώνει την κοινωνική συμπεριφορά κάποιων οργανισμών όπως την ομαδική κίνηση ψαριών.

f) Αλγόριθμοι που βασίζονται στη συμπεριφορά μελισσών

Οι αλγόριθμοι αυτοί χωρίζονται σε δυο κατηγορίες. Στη συμπεριφορά των μελισσών κατά την αναζήτηση τροφής και κατά τη διάρκεια ζευγαρώματος.

3 ΚΕΦΑΛΑΙΟ 3 : Περιγραφή και λύση του προβλήματος

3.1 Μοντελοποίηση του δυναμικού προβλήματος δρομολόγησης οχημάτων

Ένα στατικό πρόβλημα δρομολόγησης οχημάτων μπορεί να μοντελοποιηθεί μαθηματικά χρησιμοποιώντας έναν μη κατευθυνόμενο γράφο G=(C,E), όπου C είναι το σύνολο των κορυφών και E είναι οι ακμές του. Εκφράζονται ως C={c_0,c_1,….,c_n} και E={(c_i,c_j)/c_i,c_j ϵ C, $i<j$}. Επιπλέον αποτελείται από ένα σύνολο V από m ομογενή οχήματα καθένα με χωρητικότητα Q που προέρχεται από μια αποθήκη που αντιστοιχεί σε μια κορυφή c_0, και πρέπει να εξυπηρετήσει όλους τους πελάτες που αντιστοιχούν στο σύνολο C. Κάθε πελάτης i $(i>1)$ επιθυμεί μια ποσότητα από αγαθά q_i και σχετίζεται με την αντίστοιχη κορυφή. Ορίζουμε ως E ένα σύνολο ακμών που σε κάθε μία ακμή αντιστοιχεί μια μη αρνητική ποσότητα (απόσταση, χρόνο διαδρομής) D=(d_{ij}) μεταξύ των πελατών c_i και c_j. Ο στόχος είναι να βρεθεί ένα εφικτό σύνολο από διαδρομές με την μικρότερη δυνατή διανυόμενη απόσταση. Η λύση σε ένα πρόβλημα δρομολόγησης οχημάτων είναι ένα σύνολο *m* οχημάτων που πραγματοποιούν διαδρομές ελαχίστου κόστους, ξεκινώντας και καταλήγοντας στην αποθήκη, έτσι ώστε κάθε κορυφή στο C (κάθε πελάτης) να εξυπηρετείται από ακριβώς ένα όχημα. Το άθροισμα της ζήτησης που σχετίζεται με τις κορυφές δεν υπερβαίνει την αντίστοιχη χωρητικότητα του οχήματος Q. Σαν χωρητικότητα ορίζουμε την ποσότητα των αγαθών (αντικειμένων) που μπορεί να μεταφέρει το όχημα κατά τη διάρκεια των διαδρομών. Μια λύση S=R_1,…..,R_m έχει κόστος διαδρομής, R_j={c_0,c_1,….,c_{k+1} }, όπου c_i ϵC και c_0=c_{k+1} δίνεται από

$$Cost(R_j) = \sum_{\iota=1}^{\kappa} d_{i,i+1} \quad (1)$$

Και το κόστος της λύσης του προβλήματος (S) είναι :

$$F_{VRP}(S) = \sum_{j=1}^{m} Cost(R_j) \quad (2)$$

Η συνολική ζήτηση για οποιαδήποτε διαδρομή δεν μπορεί να υπερβαίνει την χωρητικότητα του οχήματος

$$Demand(R_j) = \sum_{i=1}^{k} q_i * y_i^j \leq Q^j \quad (3)$$

Όπου q_i είναι η σχετική ποσότητα που αφορά τον πελάτη c_i (αντικείμενα που πρέπει να παραδοθούν ή να παραληφθούν), Q^j είναι η χωρητικότητα του οχήματος j και y^j_i= 1 αν το c_i εξυπηρετείται από το όχημα *j* ή y^j_i= 0 αν δεν εξυπηρετείται από το όχημα *j.* (4)

Θεωρούμε σαν χρόνο εξυπηρέτησης δ_i (χρόνος που χρειάζεται για την φόρτωση/εκφόρτωση όλων των αγαθών) και απαιτείται από ένα όχημα ώστε να φορτώσει την ποσότητα q_i στο c_i. Απαιτείται η συνολική διάρκεια του δρομολογίου του κάθε οχήματος να μην ξεπερνάει ένα όριο *T*, έτσι ώστε η διαδρομή $R_j=\{c_0,c_1,....,c_{k+1}\}$ να είναι εφικτή. Εάν το όχημα σταματάει ακριβώς μια φορά σε κάθε πελάτη και ο χρόνος της διαδρομής δεν υπερβαίνει το προκαθορισμένο όριο Τ που αντιστοιχεί στο τέλος της ημερήσιας εργασίας.

$$Time(Rj) = \sum_{i=0}^{k} d_{i,i+1} + \sum_{i=1}^{k} \delta_i \leq T \quad (5)$$

Δύναται να υπάρχουν και ορισμένοι περιορισμοί όπως για παράδειγμα η χωρητικότητα κάθε οχήματος, ο συνολικός χρόνος της διαδρομής για κάθε όχημα, το χρονικό περιθώριο που έχει ένα όχημα ώστε να επισκεφτεί έναν συγκεκριμένο πελάτη. Το βασικό VRP σχετίζεται με πελάτες οι οποίοι είναι γνωστοί εκ των προτέρων, όλες οι υπόλοιπες πληροφορίες όπως για παράδειγμα η απόσταση μεταξύ των πελατών και ο χρόνος εξυπηρέτησης του καθενός είναι επίσης γνωστά εκ των προτέρων.

Το δυναμικό πρόβλημα δρομολόγησης οχημάτων (DVRP) σχετίζεται πολύ με το στατικό VRP, καθώς μπορεί να περιγραφεί σαν ένα πρόβλημα δρομολόγησης στο οποίο οι σχετικές πληροφορίες δύναται να αλλάξουν κατά την εξέλιξη της διαδικασίας βελτιστοποίησης. Καθώς το συμβατικό στατικό VRPs είναι NP-hard, το DVRP επίσης ανήκει στην κλάση των NP-hard προβλημάτων. Ένα διακριτού χρόνου δυναμικό πρόβλημα μπορεί να θεωρηθεί σαν μια ακολουθία από P περιστατικά, όπου κάθε περιστατικό είναι ένα στατικό πρόβλημα, το οποίο ξεκινά την χρονική στιγμή t και πρέπει να επιλυθεί μέσα σε ένα χρονικό περιθώριο Δ_t.

$$P = \left\{ \left(P_i, t_i, \Delta_\tau\right) \middle/ \iota = 0,1,\ldots,i_{\max} \right\}. \quad (6)$$

Με αυτές τις πληροφορίες η διάρκεια του παραδείγματος i είναι $t_{i+1} - t_i$. Ο μέγιστος αριθμός από παραδείγματα i_{max} μπορεί να είναι άπειρος αν τα χρονικά διαστήματα του προβλήματος δεν είναι σταθερά. Ένα νέο παράδειγμα P_{i+1} δημιουργείται από τη δράση της αλλαγής περιβάλλοντος ρ_i στο παράδειγμα i. Αυτό εκφράζεται με τη σχέση: $P_{I+1}=\rho_i \oplus P_i$. Αυτή η αλλαγή στο περιβάλλον μπορεί να οφείλεται σε διάφορους παράγοντες. Για παράδειγμα ο χρόνος της διαδρομής μπορεί να εξαρτάται από τον χρόνο ή την κίνηση, οι παραγγελίες μπορεί να κυρώνονται ή να αλλάζουν.

Η σχεδίαση ενός πραγματικού χρόνου αλγορίθμου δρομολόγησης εξαρτάται σε μεγάλο βαθμό κατά πόσο δυναμικό είναι το πρόβλημα. Για να προσδιορίσουμε αυτή την έννοια, ορίζεται ό βαθμός της δυναμικότητας του προβλήματος (dod). Χωρίς την απώλεια της γενικότητας, θεωρούμε ότι ο χρονικός ορίζοντας είναι το δοθέν διάστημα $[0,T]$ πιθανώς θα διαιρείται σε ένα πεπερασμένο αριθμό από μικρότερα διαστήματα. Ορίζουμε n_s και n_d τον αριθμό της στατικής και δυναμικής ζήτησης αντίστοιχα. Επιπλέον ας είναι $\tau_i \epsilon [0,T]$ ο χρόνος εμφάνισης της ζήτησης για εξυπηρέτηση i. Η στατική ζήτηση προκύπτει από $\tau_i=0$ αλλά στη δυναμική $\tau_i \epsilon [0,T]$. Ο βαθμός δυναμικότητας ορίζεται από τον τύπο

$$dod = \frac{n_d}{n_s + n_d} \quad (7)$$

που μπορεί να κυμαίνεται μεταξύ 0 και 1 [20].

3.2 Επίλυση του προβλήματος

Αρχικά επιλύουμε ένα στατικό πρόβλημα με όλους τους πελάτες. Ο λόγος που γίνεται αυτό είναι γιατί θέλαμε να έχουμε μια αρχική λύση που να είναι όσο το δυνατό πιο ικανοποιητική στο στατικό πρόβλημα που είναι το πρόβλημα του οποίου έχουμε όλα τα δεδομένα εκ των προτέρων. Οι αλγόριθμοι υλοποιήθηκαν στο προγραμματιστικό περιβάλλον matlab. Αρχικά βρίσκουμε μια λύση του προβλήματος με τον αλγόριθμο του πλησιέστερου γείτονα και στη συνέχεια με τη χρήση του αλγορίθμου της προσομοιωμένης ανόπτησης (και με αλγορίθμους τοπικής αναζήτησης τους 2-opt, 1-0 relocate, 1-1 exchange) γίνεται προσπάθεια να βρεθεί μια όσο το δυνατό καλύτερη λύση του στατικού προβλήματος.

Για την καλύτερη κατανόηση της διαδικασίας στη συνέχεια θα γίνει η παρουσίαση ενός παραδείγματος αναφοράς και βάση αυτού θα παρουσιαστούν οι αλγόριθμοι και στο στατικό πρόβλημα και στο δυναμικό πρόβλημα. Έχουμε αρχικά ένα πλήθος από οχήματα ίδιας χωρητικότητας και 51 κόμβους. Ο κόμβος νούμερο ένα, αποτελεί την αποθήκη από όπου ξεκινάει και καταλήγουν τα οχήματα και επισκέπτεται μια φορά ακριβώς όλους τους υπόλοιπους κόμβους. Κάθε αριθμημένος κόμβος συμβολίζει και έναν πελάτη. Ο στόχος του προβλήματος είναι το όχημα να επισκεφθεί όλους τους κόμβους από μια φορά ακριβώς με το μικρότερο δυνατό κόστος. Με τον όρο βέλτιστος τρόπος στο συγκεκριμένο πρόβλημα εννοείται ότι θα πρέπει το όχημα να διανύσει τη μικρότερη δυνατή απόσταση ικανοποιώντας όλους τους πελάτες χωρίς να παραβιάζονται οι περιορισμοί του προβλήματος.

Οι περιορισμοί που δόθηκαν στο συγκεκριμένο πρόβλημα είναι ότι ο μέγιστος χρόνος διαδρομής δεν μπορεί να υπερβαίνει τις 200 μονάδες, όπως και η χωρητικότητα του οχήματος δεν μπορεί να υπερβαίνει το μέγιστο φορτίο που είναι 160 μονάδες προϊόντος.

Αρχικά τα δεδομένα που αφορούν τις συντεταγμένες του κάθε κόμβου, και βρίσκονται αποθηκευμένα σε ένα φύλλο excel, διαβάστηκαν από το matlab. Στη συνέχεια βρέθηκαν οι ευκλείδειες αποστάσεις μεταξύ των κόμβων και αποθηκεύτηκαν σε έναν πίνακα. Το όχημα ξεκινώντας από την αφετηρία επισκέφθηκε τον κοντινότερο σε αυτήν κόμβο (γείτονας). Πριν προστεθεί στη διαδρομή ως γείτονας, πρέπει να γίνει ο εξής έλεγχος: αν ο χρόνος από τον προηγούμενο κόμβο (αφετηρία) μέχρι τον γείτονα προσθέτοντας τον χρόνο εξυπηρέτησης του γείτονα και τον χρόνο επιστροφής από τον γείτονα στην αποθήκη επαρκεί, δεν παραβιάζει τον περιορισμό του χρόνου, και αν το απόθεμα που έχει το όχημα φτάνει ώστε να εξυπηρετήσει τη ζήτηση του γείτονα. Εάν δεν παραβιάζεται κανένας από τους δύο περιορισμούς το όχημα επισκέπτεται τον γείτονα. Εκεί παραδίδει τις x μονάδες αποθέματος που έχει ως ζήτηση ο γείτονας με αποτέλεσμα να μειωθεί κατά x μονάδες το διαθέσιμο υπόλοιπο αποθέματος που διαθέτει το όχημα. Ἑχει μειωθεί και ο διαθέσιμος χρόνος του οχήματος, που μπορεί να εκτελεί το δρομολόγιο. Επιπλέον σε ένα διάνυσμα προστίθεται κάθε φόρα ο συνολικός χρόνος που έχει διανύσει το όχημα ώστε σε κάθε βήμα να γίνεται εύκολα ο έλεγχος για παραβίαση των περιορισμών. Η διαδικασία συνεχίζεται μέχρι το όχημα να έχει εξυπηρετήσει όλους τους κόμβους ακριβώς μια φορά.

Εάν παραβιάζεται κάποιος από τους παραπάνω περιορισμούς το όχημα επιστρέφει στην αφετηρία για να ξεκινήσει πάλι από την αρχή η διαδικασία, έχοντας αρχικοποιήσει το χρόνο και το απόθεμα. Στην ουσία είτε έχουμε ένα όχημα το οποίο επιστρέφει στην αποθήκη, είτε έχουμε πολλά οχήματα τα οποία κινούνται παράλληλα η διαδικασία είναι ίδια. Ο λόγος είναι ότι αρχικά σχεδιάζουμε τις διαδρομές εκ των προτέρων, δηλαδή με τους πελάτες και τη ζήτησή τους πριν ξεκινήσουν οι διαδρομές. Άρα αυτό που μας ενδιαφέρει να βρούμε είναι η ανάθεση των πελατών στα οχήματα (στο όχημα) και η σειρά με την οποία θα επισκεφτούμε τους πελάτες που έχουν ανατεθεί στο κάθε όχημα.

Βρέθηκε ότι η συνολική απόσταση που διένυσε το όχημα είναι 784,5020 μονάδες μήκους και ο συνολικός χρόνος 1294,5 μονάδες χρόνου. Μέσα σε αυτό το χρονικό διάστημα το όχημα πραγματοποίησε 7 κυκλικές διαδρομές που αρχίζουν και καταλήγουν στην αφετηρία.

Στην παρακάτω εικόνα φαίνεται η σειρά με την οποία το όχημα επισκέφτηκε τους κόμβους

```
path =

  Columns 1 through 23

    1    47    13    48     5    18    38    16    45    43    20     1    28     2    33    12    39     6    50    10    51    17     1

  Columns 24 through 46

    7    49     9    27    32    29     4    21     1    19    15    26    14    42    41     1    23     3    30    22    35    31    11

  Columns 47 through 58

    1    24     8    44    25    46     1    34    40    36    37     1
```

Εικόνα 3: Οι κυκλικές διαδρομές που προέκυψαν από τον αλγόριθμο του πλησιέστερου γείτονα

3.3 Αναζήτηση καλύτερης λύσης

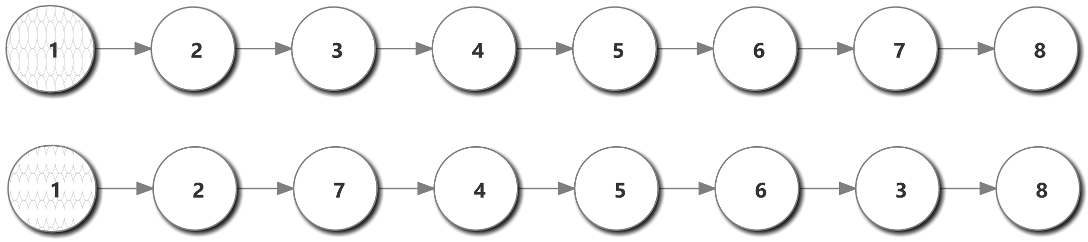
Αφού έχει βρεθεί μια αρχική εφικτή λύση με τον αλγόριθμο του πλησιέστερου γείτονα, ακολουθεί η διαδικασία εύρεσης καλύτερης λύσης από την αρχική μέσω του μεθευρετικού αλγορίθμου της προσομοιωμένης ανόπτησης και των αλγορίθμων της τοπικής αναζήτησης.

3.3.1 Τοπική αναζήτηση 1-1 ανταλλαγή (1-1 exchange)

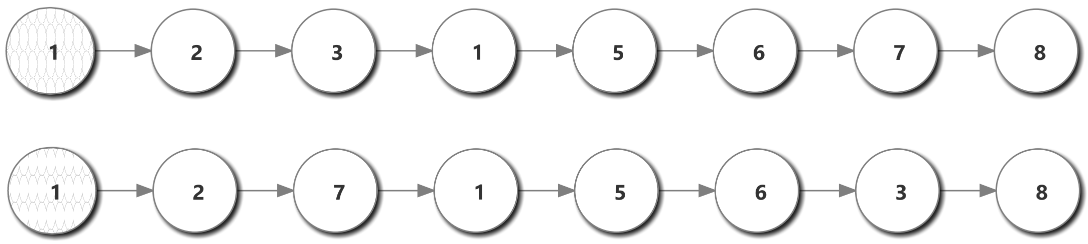
Στο συγκεκριμένο αλγόριθμο παίρνουμε σαν είσοδο την αρχική διαδρομή που προέκυψε από τον πλησιέστερο γείτονα. Διαλέγουμε τυχαία δύο διαφορετικούς κόμβους που δεν είναι η αφετηρία, και τους αντιμεταθέτουμε. Η ανταλλαγή αυτή γίνεται είτε μεταξύ κόμβων της ίδιας κυκλικής διαδρομής, είτε διαφορετικής κυκλικής διαδρομής. Προκύπτει έτσι μια νέα διαδρομή για την οποία πραγματοποιείται σχετικός έλεγχος αν παραβιάζονται οι παραπάνω περιορισμοί και αν η τελική απόσταση που διένυσε το όχημα είναι μικρότερη από την αρχική (αυτή που προέκυψε από τον αλγόριθμο του πλησιέστερου γείτονα).

Εάν παραβιάζεται κάποιος από τους περιορισμούς του προβλήματος ή αν η απόσταση που έχει διανύσει είναι μικρότερη από την αρχική, η παρούσα διαδρομή θεωρείται μη αποδεκτή και δεχόμαστε ως βέλτιστη λύση το αρχικό μονοπάτι-διαδρομή. Στην αντίθετη περίπτωση κρατάμε σαν βέλτιστη λύση το νέο μονοπάτι που προέκυψε.

Πραγματοποιήθηκαν 10000 επαναλήψεις και βρέθηκε ότι η συνολική απόσταση που διένυσε το όχημα είναι 700,0574 μονάδες μήκους και η συνολική διάρκεια του δρομολογίου είναι 1233,5 μονάδες χρόνου. Μέσα σε αυτό το χρονικό διάστημα το όχημα πραγματοποίησε 6 κυκλικές διαδρομές που αρχίζουν και καταλήγουν στην αφετηρία. Άρα η αναζήτηση βέλτιστης λύσης με τον αλγόριθμο 1-1 exchange δεν βελτίωσε σημαντικά την αρχική λύση του προβλήματος.



Εικόνα 4: 1-1 exchange όταν οι κόμβοι βρίσκονται στην ιδία κυκλική διαδρομή



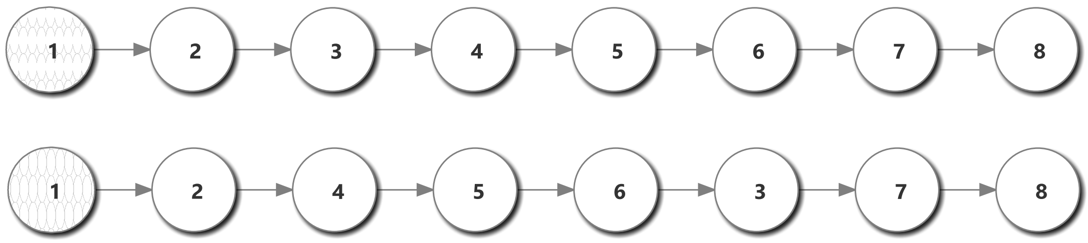
Εικόνα 5: 1-1 exchange όταν οι κόμβοι δεν βρίσκονται στην ιδία κυκλική διαδρομή

3.3.2 Τοπική αναζήτηση 1-0 επανατοποθέτηση(1-0 relocate)

Σε αυτήν τη μέθοδο, επιλέγουμε τυχαία έναν κόμβο, τον αφαιρούμε από τη θέση που βρίσκεται και τον τοποθετούμε ανάμεσα σε δύο άλλους κόμβους. Ο αλγόριθμος αυτός βασίζεται στην ιδέα της διαγραφής ενός πελάτη από τη θέση του και την επανατοποθέτηση του σε μια άλλη κυκλική διαδρομή η οποία θα επιφέρει μικρότερο κόστος. Οι επιλεγόμενοι κόμβοι δύναται να επιλεχτούν από την ίδια ή από διαφορετική κυκλική διαδρομή.

Μετά από δοκιμές που έγιναν στη σχετική επανατοποθέτηση του επιλεγόμενου κόμβου το μικρότερο δυνατό κόστος προήλθε από την εξής επανατοποθέτηση. Επιλέγουμε τυχαία δυο διαφορετικούς κόμβους. Ο κόμβος με τον μικρότερο αύξοντα αριθμό διαγράφεται από τη θέση που βρίσκεται και τοποθετείται ακριβώς μετά τον δεύτερο επιλεγόμενο κόμβο με τον μεγαλύτερο αύξοντα αριθμό.

Για κάθε τυχαία επανατοποθέτηση κόμβων στο μονοπάτι γίνεται έλεγχος ώστε να διαπιστωθεί αν παραβιάζονται οι περιορισμοί του προβλήματος. Αν ικανοποιούνται όλοι οι περιορισμοί τότε γίνεται έλεγχος αν η συνολική απόσταση που διένυσε το όχημα είναι μικρότερη από την αρχική (η απόσταση που προήλθε από τον αλγόριθμο του πλησιέστερου γείτονα). Εάν είναι μικρότερη την δεχόμαστε και η αντίστοιχη διαδρομή θεωρείται η βέλτιστη μέχρι τώρα. Εάν παραβιάζεται θεωρούμε καλύτερη διαδρομή την αρχική λύση του προβλήματος.



Εικόνα 6: 1-0 relocate

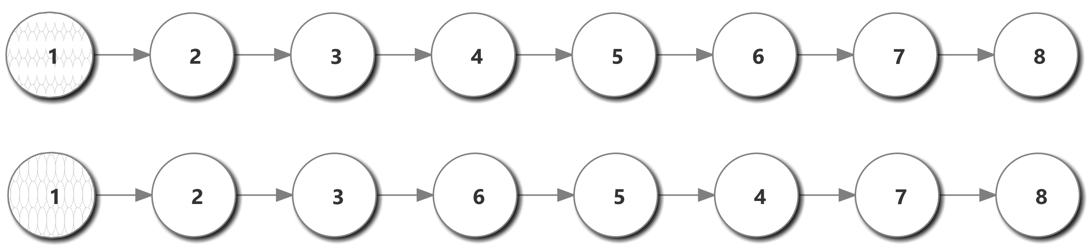
Πραγματοποιήθηκαν 10000 επαναλήψεις και βρέθηκε ότι η συνολική απόσταση που διένυσε το όχημα είναι 610,7889 μονάδες μήκους και ο συνολική διάρκεια είναι 1284,5 μονάδες χρόνου. Μέσα σε αυτό το χρονικό διάστημα το όχημα πραγματοποίησε 6 κυκλικές διαδρομές που αρχίζουν και καταλήγουν στην αφετηρία. Άρα η αναζήτηση βέλτιστης λύσης με τον αλγόριθμο 1-0 relocate βελτίωσε αισθητά την αρχική λύση του προβλήματος και επιπλέον μειώθηκε ο αριθμός των κυκλικών διαδρομών.

3.3.3 Τοπική αναζήτηση 2-opt

Η διαδικασία εφαρμογής του αλγορίθμου είναι η ακόλουθη. Αρχικά έγινε τυχαία επιλογή δυο κόμβων, είτε αυτοί βρίσκονται στην ίδια ή σε διαφορετική κυκλική διαδρομή. Τα στοιχεία που βρίσκονται μεταξύ των δύο κόμβων αντιστρέφονται αλλάζοντας κατεύθυνση.

Για κάθε νέα διαδρομή που προκύπτει μετά από την εφαρμογή του αλγορίθμου 2-opt, γίνεται έλεγχος ώστε να διαπιστωθεί αν παραβιάζονται οι περιορισμοί του προβλήματος. Αν ικανοποιούνται όλοι οι περιορισμοί τότε γίνεται έλεγχος αν η συνολική απόσταση που διένυσε το όχημα είναι μικρότερη από την αρχική. Εάν είναι μικρότερη την δεχόμαστε και η αντίστοιχη διαδρομή θεωρείται η βέλτιστη μέχρι τώρα. Εάν παραβιάζεται θεωρούμε καλύτερη διαδρομή την αρχική λύση του προβλήματος.

Πραγματοποιήθηκαν 10000 επαναλήψεις και βρέθηκε ότι η συνολική απόσταση που διένυσε το όχημα είναι 632,7444 μονάδες μήκους και η διάρκεια είναι 1284,5 μονάδες χρόνου. Μέσα σε αυτό το χρονικό διάστημα το όχημα πραγματοποίησε 7 κυκλικές διαδρομές που αρχίζουν και καταλήγουν στην αφετηρία. Άρα η αναζήτηση βέλτιστης λύσης με τον αλγόριθμο 2-opt βελτίωσε την αρχική λύση του προβλήματος αλλά δεν μειώθηκε ο αριθμός των κυκλικών διαδρομών.



Εικόνα 7: 2-opt

3.4 Βελτιστοποίηση του προβλήματος με μεθευρετικό αλγόριθμο

3.4.1 Επίλυση του προβλήματος

Μετά την υλοποίηση της τοπικής αναζήτησης πραγματοποιήθηκε βελτιστοποίηση του προβλήματος με τον μεθευρετικό αλγόριθμο προσομοιωμένης ανόπτησης σε συνδυασμό με τις παραπάνω μεθόδους τοπικής αναζήτησης. Ο ψευδοκώδικας του αλγορίθμου παρουσιάζεται ακολούθως:

Αλγόριθμος Προσομοιωμένη ανόπτηση

Επίλεξε μια αρχική λύση s0

Επίλεξε μια αρχική θερμοκρασία t0

Επίλεξε τη συνάρτηση μείωσης της θερμοκρασίας a(t),

- $\alpha(t)=t\cdot e^{-a\cdot i}$με α=0.95 ή α=0.98
- $\alpha(t)=t\cdot a^{i}$
- a(t)=a·t (a<1) με ≤0.8α≤0.99 όπου $t_i=t_0/\log(i)$

Repeat

Repeat

τυχαία επιλογή μιας γειτονιάς s є N(s0)

δ=f(s)-f(s0), όπου f η αντικειμενική συνάρτηση του προβλήματος

if δ<0 τότε

s0=s

else

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

if $x<e^{-\delta/t}$ τότε

s0=s

endif

endif

until να ολοκληρωθεί ο αριθμός των επαναλήψεων

t=a(t)

until μέχρι να ικανοποιηθεί κάποιο κριτήριο τερματισμού

Επίστρεψε τη βέλτιστη λύση

Αναλυτικά τα βήματα της μεθόδου συνοψίζονται παρακάτω:

1. Γίνεται επιλογή μιας αρχικής λύσης x0 (υπάρχουσα διαδρομή προέκυψε από τον αλγόριθμο πλησιέστερος γείτονας) και ο υπολογισμός της f(x0) (κόστος διαδρομής), όπου f είναι η αντικειμενική συνάρτηση του προβλήματος, δηλαδή το κόστος της υπάρχουσας διαδρομής.
2. Ορίζονται οι παράμετροι. Γίνεται καθορισμός της αρχικής θερμοκρασίας t0 και της τελικής ως κριτήριο τερματισμού του αλγορίθμου, η συνάρτηση μείωσης της θερμοκρασίας a(t) και ένα κριτήριο διάρκειας του αλγορίθμου (στο παρόν πρόβλημα έχουμε ορίσει μέγιστο αριθμό επαναλήψεων).
3. Έχοντας υπολογίσει την αρχική λύση και το αντίστοιχο κόστος πραγματοποιείται η εύρεση μιας νέας γειτονικής λύσης x_{k+1}. Με μια ανατάραξη του συστήματος προκύπτει μια γειτονική λύση (στο παρόν πρόβλημα οι γειτονικές λύσεις έχουν προκύψει με την τοπική αναζήτηση 1-0 relocate, 2-opt, 1-1 exchange). Το μέγεθος της περιοχής των γειτονικών λύσεων μεταβάλλεται τις περισσότερες φορές κατά τη διαδικασία για επιτάχυνση της σύγκλησης.
4. Γίνεται ο υπολογισμός της αντικειμενικής συνάρτησης $f(x_{k+1})$,δηλαδή το κόστος που αντιστοιχεί στη νέα λύση.
5. Σύγκριση $f(x_κ)$ και $f(x_{k+1})$ και επιλογή αν θα γίνει αποδεκτή η νέα λύση του προβλήματος. Αν δεν γίνει αποδεκτή γίνεται επιστροφή στο βήμα

4. Εάν γίνει αποδεκτή, την θεωρούμε ως την βέλτιστη λύση μέχρι τώρα.

6. Έλεγχος κριτηρίων σύγκλισης. Εάν υπάρχει σύγκλιση τερματίζει η διαδικασία. Εάν δεν υπάρχει γίνεται η μεταβολή της διαδικασίας και επιστροφή στο βήμα 3.

Πριν ξεκινήσει ο αλγόριθμος, έχει οριστεί η συνολική απόσταση που έχει διανύσει το όχημα από τον αλγόριθμο του πλησιέστερου γείτονα ως bestdist, δηλαδή ως η μικρότερη απόσταση μέχρι τώρα, και το μονοπάτι που διένυσε ως bestpath, δηλαδή ως η βέλτιστη διαδρομή-ακολουθία κόμβων.

3.4.2 Περιγραφή αλγορίθμου

- <u>Βήμα 1</u>

  Υπολογισμός αντικειμενικής συνάρτησης του προβλήματος. Ο αλγόριθμος σαν όρισμα δέχεται την παρούσα διαδρομή που έχει διανύσει το όχημα και το κόστος της διαδρομής αποτελεί την αντικειμενική συνάρτηση του προβλήματος

- <u>Βήμα 2</u>

  Ορίζεται η αρχική θερμοκρασία t=1000

  Ορίζεται η σταθερά a=0.95

  Κριτήριο τερματισμού του αλγόριθμου

  Επιλέχτηκε ως πρώτο κριτήριο τερματισμού ο αριθμός των μέγιστων επαναλήψεων να είναι 10.000 επαναλήψεις

  Δεύτερο κριτήριο τερματισμού

  Επιλέχτηκε ως δεύτερο κριτήριο τερματισμού ο περιορισμός στην τιμή της θερμοκρασίας, t<5

  Συνάρτηση μείωσης της θερμοκρασίας :a_t=a*t με a=0,88

- <u>Βήμα 3</u>

Εύρεση μιας νέας λύσης στην γειτονιά της αρχικής λύσης

Η νέα λύση προκύπτει από τους αλγόριθμους τοπικής αναζήτησης

- Βήμα 4

  Υπολογισμός αντικειμενικής συνάρτησης με όρισμα την νέα λύση που προέκυψε

- Βήμα 5

  Επιλογή δεκτής λύσης

  d=λύση που πρόεκυψε στην τωρινή επανάληψη από τον ευρετικό αλγόριθμο- την καλύτερη λύση μέχρι τώρα.

  Δεχόμαστε την μικρότερη λύση μέχρι τώρα, αλλά με μια πιθανότητα η οποία προκύπτει από τον τύπο x>exp(-d/t), δεχόμαστε και την χειρότερη λύση

- Βήμα 6

  Αλλαγή θερμοκρασίας: t=a_t

- Βήμα 7

  Επανάληψη Βήμα 3 και έλεγχος αν ισχύουν τα κριτήρια τερματισμού [21].

Πραγματοποιήθηκαν δοκιμές με όλους τους συνδυασμούς της προσομοιωμένης ανόπτησης με τους ευρετικούς αλγορίθμους τοπικής αναζήτησης και το μικρότερο κόστος που προέκυψε από την παραπάνω διαδικασία είναι 712.6843 μονάδες μήκους και διάρκεια είναι 1225,5 μονάδες χρόνου. Αυτή η τιμή προέκυψε από τον συνδυασμό της προσομοιωμένης ανόπτησης με αρχική θερμοκρασία t=1000, a=0,98, κριτήρια τερματισμού του αλγορίθμου 10.000 επαναλήψεις, ή t>5 και του ευρετικού αλγορίθμου 1-0 relocate.

3.4.3 Λύση του δυναμικού προβλήματος δρομολόγησης οχημάτων

Το βασικό χαρακτηριστικό του δυναμικού προβλήματος είναι ότι εξελίσσεται σε πραγματικό χρόνο, άρα οι κυκλικές διαδρομές που θα διανύσουν τα οχήματα δεν μπορούν να σχεδιαστούν εκ των προτέρων, καθώς δεν είναι γνωστές. Αυξάνεται η πολυπλοκότητα του προβλήματος και για αυτόν τον λόγο παρακάτω παρουσιάζεται αναλυτικά η λύση ενός ενδεικτικού παραδείγματος, ώστε να γίνει εύκολα κατανοητό.

Τα δεδομένα του προβλήματος είναι:

- Μέγιστη χωρητικότητα : 50
- Αριθμός οχημάτων:1
- Μέγιστο μήκος διαδρομής (maxroute): 150

Πελάτης	Ζήτηση (Demand)	Χρόνος εξυπηρέτησης (Service time)	Χρονική στιγμή που Τηλεφωνούν για εξυπηρέτηση (Call time)
1	0	0	0
2	20	10	10
3	10	15	20
4	15	10	40
5	10	5	45
6	20	10	60
7	15	15	90
8	10	5	110
9	5	10	120
10	25	15	130

Εικόνα 8: Δεδομένα προβλήματος

Πίνακας αποστάσεων										
ΚΟΜΒΟΙ	1	2	3	4	5	6	7	8	9	10
1	0	12	19	31	22	16	22	11	24	34
2	12	0	15	37	21	28	34	22	16	28
3	19	15	0	49	36	35	35	21	31	43
4	31	37	49	0	20	21	37	37	32	31
5	22	21	36	20	0	25	40	33	12	14
6	16	28	35	21	25	0	16	18	34	39
7	22	34	35	37	40	16	0	14	46	54
8	11	22	21	37	33	18	14	0	35	45
9	24	16	31	32	12	34	46	35	0	12
10	34	28	43	31	14	39	54	45	12	0

Εικόνα 9: Πίνακας αποστάσεων μεταξύ των κόμβων

Αρχικά γίνεται η σχεδίαση του στατικού προβλήματος, με την μέθοδο του πλησιέστερου γείτονα και προκύπτουν οι αντίστοιχες σχετικές κυκλικές διαδρομές. Ο λόγος που επιλύσαμε το στατικό πρόβλημα είναι ότι θέλουμε να δούμε πόσα περίπου οχήματα θα χρειαστούν για να κάνουμε τις εξυπηρετήσεις όλων των πελατών ούτως ώστε να έχει γίνει μια αρχική κατανομή. Αν δεν είχαμε χρησιμοποιήσει αυτή τη διαδικασία τότε θα δημιουργούταν το ακόλουθο πρόβλημα. Οι αλγόριθμοι που χρησιμοποιούμε έχουν τη λογική ότι όταν θα παραβιαστεί κάποιος περιορισμός στο όχημα τότε το όχημα θα επιστρέψει στην αποθήκη και το ίδιο ή ένα διαφορετικό όχημα θα ξεκινήσει μία καινούρια διαδρομή. Όπως αναφέραμε και προηγούμενα στο δυναμικό πρόβλημα έχουμε δύο είδη πελατών αυτούς που ξέρουμε από πριν ότι πρέπει να εξυπηρετηθούν στα σχεδιασμένα δρομολόγια και αυτούς που η ζήτηση τους εμφανίζεται κατά τη διάρκεια των δρομολογίων. Έτσι αν επιλύαμε το πρόβλημα μόνο με τους πελάτες που είχαμε από πριν χωρίς να αφήσουμε περιθώριο στο κάθε όχημα για τους επιπλέον πελάτες που θα εμφανιστούν κατά τη διάρκεια της διαδικασίας τότε δεν θα μπορούσαμε να προσθέσουμε καινούριους πελάτες στη ήδη υπάρχουσες διαδρομές. Αυτό το πρόβλημα θα μπορούσε να αντιμετωπιστεί με διάφορους τρόπους, θα μπορούσαμε να αλλάξουμε τους περιορισμούς στα οχήματα και να πούμε ότι τα οχήματα ξεκινάνε την εξυπηρέτηση έχοντας ως στόχο να εξυπηρετήσουν πελάτες μέχρι το άθροισμα της ζήτησης των πελατών να είναι ίσο με το σύνολο της χωρητικότητας του οχήματος Q πολλαπλασιασμένο με ένα κατώφλι (threshold), για παράδειγμα 0.5, που σημαίνει πως μόλις η εξυπηρέτηση ξεπερνούσε τη τιμή στόχο (threshold * Q) το όχημα θα επέστρεφε στην αποθήκη και με αυτόν τον τρόπο τα οχήματα θα είχαν τη δυνατότητα να μπορούν να εξυπηρετήσουν κάποιους από τους επιπλέον πελάτες που θα εμφανιζόταν κατά τη διάρκεια των δρομολογίων. Αυτή είναι η πιο συνήθης τακτική που ακολουθείται στη βιβλιογραφία. Το κατώφλι που χρησιμοποιείται συνήθως υπολογίζεται από το βαθμό δυναμικότητας που περιγράψαμε προηγούμενα, που σημαίνει όσο πιο μεγάλος είναι ο βαθμός δυναμικότητας, δηλαδή όσο λιγότεροι είναι οι πελάτες που θα μπουν στις εκ των προτέρων διαδρομές, τόσο το κατώφλι θα προσεγγίζει το 0. Το κατώφλι μπορεί να πάρει τιμές από 0 εώς 1. Αυτός όμως ο τρόπος απαιτεί να ξέρουμε το βαθμό δυναμικότητας. Εμείς προσπαθήσαμε να κάνουμε ένα τρόπο ο οποίος να είναι πιο στατικός, που είναι και η πιο σημαντική πρωτοτυπία της συγκεκριμένης έρευνας, δηλαδή θεωρήσαμε ότι έχουμε ένα αρχικό πλάνο στο μυαλό μας, είτε αυτό το αρχικό πλάνο προέρχεται από κάποια εκ των προτέρων γνώση είτε από ιστορικά δεδομένα και δημιουργήσαμε κάποιες εκ των προτέρων διαδρομές με περισσότερους πελάτες από αυτούς που πραγματικά έχουμε, τους πραγματικούς πελάτες (real customers) και τους τεχνητούς πελάτες (artificial customers). Στη συνέχεια αφαιρέσαμε τους τεχνητούς πελάτες και είχαμε πλέον μια αρχική κατανομή των πελατών σε οχήματα. Τι κερδίσαμε με αυτή τη διαδικασία. Κερδίσαμε ότι αφού τα οχήματα ξεκινάνε γεμάτα τις διαδρομές οι περιορισμοί χειρίζονται σωστά. Δηλαδή δύο πελάτες που βρίσκονται πολύ κοντά και με την κλασική διαδικασία αντιμετώπισης του προβλήματος θα βρισκόταν σε διαφορετικές διαδρομές λόγω της τιμής κατωφλίου, τώρα με τον προτεινόμενο τρόπο αντιμετώπισης βρίσκονται στην ίδια διαδρομή όπως θα έπρεπε να βρίσκονται και δεν θα χρειαστεί κάποια διαδικασία τοπικής αναζήτησης για να ξαναβρεθούν πολύ κοντά όπως θα πρέπει να συμβεί στην κλασσική περίπτωση αντιμετώπισης του προβλήματος. Στο παράδειγμα υπάρχει η παραδοχή ότι ο χρόνος που χρειάζεται το όχημα για να πάει από τον έναν κόμβο στον άλλο ισούται με την αντίστοιχη απόσταση.

Πρώτη κυκλική διαδρομή

Ξεκινάμε από την αποθήκη (κόμβος 1) και από τον πίνακα αποστάσεων βλέπουμε ποιος κόμβος είναι πλησιέστερος σε αυτόν. Πριν τον προσθέσουμε στην διαδρομή, θα πρέπει να ελέγξουμε αν πληρούνται δυο κριτήρια, δηλαδή αν δεν παραβιάζονται οι περιορισμοί του προβλήματος. Το πρώτο είναι ότι θα πρέπει να γίνει έλεγχος εάν η ζήτηση που έχει ο κόμβος 8 δεν ξεπερνά την διαθέσιμη χωρητικότητα του οχήματος, ώστε να μπορέσει να τον εξυπηρετήσει. Δηλαδή η διαθέσιμη χωρητικότητα του οχήματος είναι 50 μονάδες προϊόντος και ο κόμβος 8 έχει ζήτηση 10 μονάδες προϊόντος άρα το πρώτο κριτήριο ικανοποιείται. Το δεύτερο κριτήριο είναι ο μέχρι τώρα χρόνος της διαδρομής δηλαδή ο χρόνος που χρειάζεται από την αποθήκη μέχρι τον κόμβο 8 και ο χρόνος εξυπηρέτησης του κόμβου 8 και ο χρόνος από τον κόμβο 8 μέχρι την αποθήκη (πρέπει να επαρκεί ο χρόνος να γυρίσει στην αποθήκη) να μην ξεπερνά το μέγιστο χρόνο της διαδρομής που είναι 150 μονάδες χρόνου. Άρα ικανοποιείται και αυτό το κριτήριο και ο κόμβος 8 προστίθεται στην διαδρομή.

Άρα path=(1,8)

Η ζήτηση του 8 είναι 10 μονάδες προϊόντος. Άρα από τη συνολική χωρητικότητα του οχήματος αφαιρούμε τη ζήτηση του κόμβου 8 για να υπολογιστεί η διαθέσιμη χωρητικότητα.

Άρα capacity =(50,40)

Το routtime προκύπτει: Απόσταση από την αποθήκη (κόμβος 1) μέχρι τον κόμβο 8, που ισούται με 11 και χρόνος εξυπηρέτησης του 8 που ισούται με 5. Άρα

routtime=(0,16)

- Ο επόμενος κόμβος που θα προστεθεί είναι ο 7, πλησιέστερος στον κόμβο 8 αφού πρώτα ικανοποιεί τα δύο σχετικά κριτήρια.
  ζήτηση (7)=15
  ζήτηση (μέχρι τώρα) + ζήτηση (7)=10+15=25 <50
  routtime= routtime (μέχρι τώρα) + απόσταση (8,7) + χρόνος εξυπηρέτησης (7) + απόσταση (7,1)=16+14+15+22=67<150
  Άρα path=(1,8,7)
  Routtime=(0,16,45)
  Capacity=(50,40,25)
- Ο επόμενος κόμβος που θα γίνει έλεγχος ώστε προστεθεί στη διαδρομή είναι ο κόμβος 6 (κοντινότερος στον κόμβο 7).
  ζήτηση(6)=20
  ζήτηση (μέχρι τώρα) + ζήτηση (6)=20+25=45<50
  routtime= routtime (μέχρι τώρα) + απόσταση (7,6) + χρόνος εξυπηρέτησης (6) + απόσταση (6,1)=67+16+10+16= 109<150
  path=(1,8,7,6)
  routtime=(0,16,45,71)
  capacity=(50,40,25,5)
- Ο επόμενος κόμβος που θα γίνει έλεγχος αν θα προστεθεί είναι ο 4 (κοντινότερος στον κόμβο 6).
  Ζήτηση(4)=10
  Ζήτηση (μέχρι τώρα) + ζήτηση (4)=45+10=55. Ξεπερνάει την χωρητικότητα του οχήματος. Άρα ο κόμβος 4 δεν προστίθεται στην διαδρομή και το όχημα γυρνάει στην αφετηρία. Σε αυτό το σημείο μπορούμε να κάνουμε την εξής παραδοχή. Αντί να γυρίσει στην

αφετηρία το όχημα να επισκεφτεί τον αμέσως κοντινότερο εφικτό κόμβο.

- Ο επόμενος κόμβος που θα γίνει έλεγχος αν θα προστεθεί είναι ο κόμβος 9.

  Ζήτηση(9)=5

  Ζήτηση (μέχρι τώρα) + ζήτηση (9)=45+5=50

  routtime= routtime (μέχρι τώρα) + απόσταση (6,9) + χρόνος εξυπηρέτησης (9) + απόσταση (9,1)= 71+34+10+24=139<150

  path=(1,8,7,6,9)

  routtime=(0,16,45,71,115)

  capacity=(50,40,25,5,0)

  Επειδή δεν υπάρχει άλλη χωρητικότητα στο όχημα, δηλαδή δεν μπορεί να εξυπηρετήσει τη ζήτηση οποιουδήποτε άλλου κόμβου το όχημα επιστρέφει στην αφετηρία να ''γεμίσει'' χρόνο και εμπόρευμα.

  Άρα η πρώτη κυκλική διαδρομή είναι:

  path=(1,8,7,6,9,1)

  routtime=(0,16,45,71,115)

  capacity=(50,40,25,5,0)

<u>Δεύτερη κυκλική διαδρομή</u>

Το όχημα ξεκινάει από την αφετηρία με διαθέσιμο χρόνο ίσο με το μέγιστο χρόνο διαδρομής και χωρητικότητα ίση με τη μέγιστη. Η διαδικασία που ακολουθείται είναι η ίδια με αυτή της πρώτης διαδρομής.

- Ξεκινάμε από την αφετηρία και βλέπουμε ποιος κόμβος είναι ο πιο κοντινός. Φυσικά δεν μπορούμε να επιλέξουμε τους κόμβους που το όχημα έχει επισκεφτεί στην προηγούμενη διαδρομή καθώς ο στόχος είναι να επισκεφτεί κάθε κόμβο από μια φορά.
- Ο επόμενος κόμβος που θα γίνει έλεγχος αν θα προστεθεί είναι ο 2 (κοντινότερος στον κόμβο 1).

Ζήτηση (2)=20

Ζήτηση (μέχρι τώρα) + ζήτηση (2)=0+20=20<50

routtime= routtime (μέχρι τώρα) + απόσταση (1,2) + χρόνος εξυπηρέτησης (2) + απόσταση (2,1)= 0+12+10+12<150

path=(1,2)

routtime=(0,22)

capacity=(50,30)

- Ο επόμενος κόμβος που θα γίνει έλεγχος αν θα προστεθεί είναι ο 3 (κοντινότερος στον κόμβο 2).

  Ζήτηση(3)=10

  Ζήτηση (μέχρι τώρα)+ ζήτηση (3)=20+10=30<50

  routtime= routtime (μέχρι τώρα) + απόσταση (2,3)+ χρόνος εξυπηρέτησης (3) + απόσταση (3,1)= 22+15+15+19<150

  path=(1,2,3)

  routtime=(0,22,52)

  capacity=(50,30,20)

- Ο επόμενος κόμβος που θα γίνει έλεγχος αν θα προστεθεί είναι ο 5 (κοντινότερος στον κόμβο 3)

  Ζήτηση (5)=10

  Ζήτηση (μέχρι τώρα)+ ζήτηση(5)=30+10<50

  routtime= routtime (μέχρι τώρα) + απόσταση (3,5) + χρόνος εξυπηρέτησης (5) + απόσταση (5,1)= 52+36+5+22<150

  path=(1,2,3,5)

  routtime=(0,22,52,59)

  capacity=(50,30,20,10)

Ο κοντινότερος κόμβος στον κόμβο 5 είναι ο 10. Αλλά το όχημα έχει διαθέσιμη χωρητικότητα 10 μονάδες προϊόντος και ο 10 έχει ζήτηση 25 μονάδες προϊόντος. Άρα ο κόμβος 10 δεν θα εξυπηρετηθεί σε αυτή τη διαδρομή. Θα γίνει έλεγχος για τον επόμενο κοντινό κόμβο που είναι ο 4. Η ζήτηση του 4 είναι 15 μονάδες και η διαθέσιμη χωρητικότητα του οχήματος 10, άρα ούτε ο 4 μπορεί να εξυπηρετηθεί.

Άρα η δεύτερη κυκλική διαδρομή που προκύπτει θα είναι:

path=(1,2,3,5)

routtime=(0,22,52,59)

capacity=(50,30,20,10)

Τρίτη κυκλική διαδρομή

Το όχημα ξεκινάει από την αφετηρία με διαθέσιμο χρόνο ίσο με maxroute και χωρητικότητα ίση με τη μέγιστη. Η διαδικασία που ακολουθείται είναι η ίδια με αυτή της πρώτης και της δεύτερης διαδρομής.

- Ο πρώτος κόμβος είναι η αφετηρία.
- Ο επόμενος κόμβος που θα γίνει έλεγχος αν θα προστεθεί είναι ο 4 (κοντινότερος στον κόμβο 1).

  Ζήτηση(4)=15

  Ζήτηση (μέχρι τώρα)+ ζήτηση (4)=15<50

  routtime= routtime (μέχρι τώρα) + απόσταση (1,4)+ χρόνος εξυπηρέτησης (4)+απόσταση (4,1)= 0+31+10+31<150

  path=(1,4)

  routtime=(0,41)

  capacity=(50,35)
- Ο επόμενος κόμβος που θα γίνει έλεγχος αν θα προστεθεί είναι ο 10 (κοντινότερος στον κόμβο 4).

  Ζήτηση (10)=25

  Ζήτηση (μέχρι τώρα)+ ζήτηση (10)=15+25=40<50

  routtime= routtime (μέχρι τώρα)+ απόσταση (1,10)+ χρόνος εξυπηρέτησης (10) + απόσταση (10,1)= 41+34+15+34=124<150

  path=(1,4,10)

  routtime=(0,41,87)

capacity=(50,35,10)

Το όχημα γυρνάει στην αποθήκη αφού τελείωσαν όλοι οι κόμβοι

Άρα

path=(1,4,10,1)

routtime=(0,41,87,121)

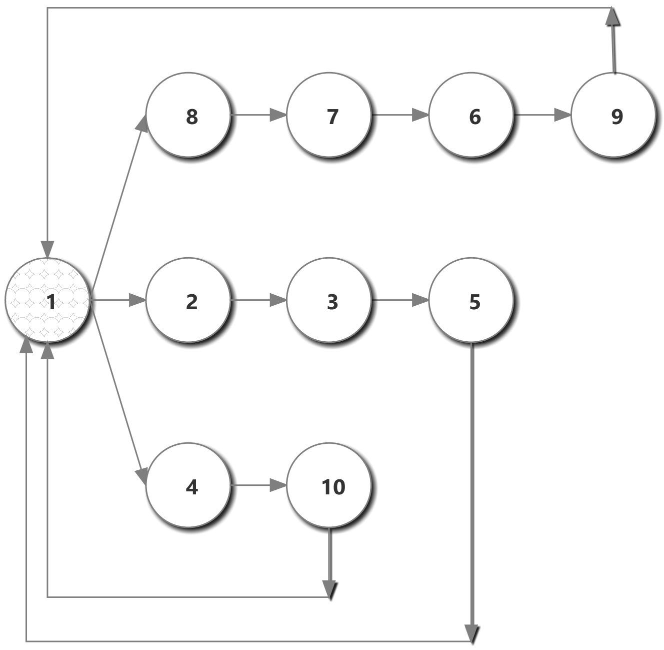
capacity=(50,35,10).

Άρα οι κυκλικές διαδρομές που προέκυψαν είναι:

Διαδρομη1=[1,8,7,6,9,1]

Διαδρομή2=[1,2,3,5,1]

Διαδρομή3=[1,4,10,1].



Εικόνα 10: Οι διαδρομές που προέκυψαν από το αλγόριθμο του πλησιέστερου γείτονα

Το επόμενο βήμα στη λύση του προβλήματος αποτελεί η δεύτερη φάση του αλγορίθμου και συγκεκριμένα το δυναμικό πρόβλημα δρομολόγησης οχημάτων.

Στο δυναμικό μέρος του προβλήματος υπάρχει ένα επιπλέον δεδομένο σε σχέση με το αντίστοιχο στατικό και είναι η ώρα που κάλεσε κάθε πελάτης για να εξυπηρετηθεί.

Τα βήματα που ακολουθήθηκαν για την επίλυση του προβλήματος είναι τα εξής :

- **<u>Βήμα 1</u>**

Διαίρεση του συνολικού χρόνου στα δύο. Όπως είπαμε προηγούμενα σε ένα δυναμικό πρόβλημα δρομολόγησης οχημάτων έχουμε δύο κατηγορίες πελατών, αυτούς που γνωρίζουμε εκ των προτέρων ότι πρέπει να τους εξυπηρετήσουμε άρα χρησιμοποιούνται για να σχεδιάσουμε τις αρχικές διαδρομές και αυτούς που μας ενημερώνουν κατά τη διάρκεια των δρομολογίων ότι θα πρέπει να τους εξυπηρετήσουμε οπότε αναπροσαρμόζουμε τα ήδη δημιουργημένα δρομολόγια προσθέτοντας τον κάθε ένα από τους πελάτες στις ήδη υπάρχουσες διαδρομές. Από τη στιγμή που έχουμε ένα σύνολο δεδομένων και θέλαμε να δοκιμάσουμε ένα αλγόριθμο, ακολουθήσαμε τον τρόπο που εργάζονται στα περισσότερα δυναμικά προβλήματα δρομολόγησης οχημάτων στη βιβλιογραφία. Θεωρητικά και ουσιαστικά όταν έχουμε ένα σύνολο από οχήματα τα οποία πρέπει να ξεκινήσουν και να επιστρέψουν στην αποθήκη μια συγκεκριμένη χρονική στιγμή, για να μπορέσει να προστεθεί ένας πελάτης σε αυτά τα δρομολόγια θα πρέπει να κάνει την παραγγελία του αρκετά νωρίς μέσα στη μέρα για να έχουμε το χρόνο να αναπροσαρμόσουμε τα δρομολόγια. Οπότε αυτό που κάναμε ήταν χωρίσαμε τους πελάτες σε 2 υποσύνολα αυτούς που έχουν καλέσει μέχρι κάποια χρονική στιγμή και αυτούς που κάλεσαν μετά. Το δεύτερο σύνολο των πελατών θεωρείται ότι δεν μπορεί να εξυπηρετηθεί την συγκεκριμένη μέρα οπότε θεωρείται ότι αυτοί οι πελάτες δεν εξυπηρετούνται σήμερα. Οπότε καθώς ένα δυναμικό πρόβλημα δρομολόγησης οχημάτων θεωρείται ως ένα πρόβλημα μακρινού ορίζοντα, κοινώς θεωρούμε ότι σχεδιάζουμε δρομολόγια για παραπάνω από μία μέρες, ξεκινάμε την πρώτη μέρα με κενά δρομολόγια και συμπληρώνουμε μόνο τους πελάτες που έρχονται κατά τη διάρκεια της ημέρας μέχρι τη χρονική στιγμή που θεωρούμε ότι δεν μπορούμε να προχωρήσουμε παρακάτω γιατί δεν θα προλάβουμε να κάνουμε ενημέρωση στα δρομολόγια. Φυσικά, έτσι μένουν ανικανοποίητοι κάποιοι πελάτες. Αυτοί οι πελάτες είναι αυτοί που θα εξυπηρετηθούν με προτεραιότητα την επόμενη μέρα. Αυτό σημαίνει ότι την επόμενη μέρα θα σχεδιάσουμε την εκ των προτέρων διαδρομή με τους πελάτες που θα έχουν μείνει ανικανοποίητοι την προηγούμενη μέρα. Αυτός είναι λογικός τρόπος σκέψης, γιατί καθώς εμάς μας ενδιαφέρει να επιλύσουμε ένα πρόβλημα πραγματικού χρόνου είναι δύσκολο να μπορούμε να εξυπηρετήσουμε κάποιους πελάτες που έχουν δώσει την παραγγελία τους μετά από κάποια χρονική στιγμή. Άρα από τις διαδρομές που προέκυψαν στην πρώτη φάση του αλγορίθμου αφαιρούμε όλους τους πελάτες που με βάση τα δεδομένα θα καλέσουν μέχρι μια χρονική στιγμή. Ο λόγος που αφήνουμε τους άλλους πελάτες είναι γιατί παρουσιάζουμε την μορφή του αλγορίθμου που έχουμε από προηγούμενη μέρα κάποιους πελάτες που δεν ικανοποιήθηκαν και άρα θα ξεκινήσουμε τις διαδρομές από αυτούς. Άρα το σύνολο των πελατών είναι οι πελάτες που έχουν μείνει ανικανοποίητοι την προηγούμενη μέρα, δηλαδή αυτοί που κάλεσαν αργά και δεν προλάβαμε να τους εξυπηρετήσουμε, και με αυτούς τους πελάτες θα ξεκινήσουμε τις διαδρομές και οι υπόλοιποι πελάτες που είναι οι πελάτες οι οποίοι θα καλέσουν κατά τη διάρκεια της ημέρας μέχρι τη χρονική στιγμή που έχουμε βάλει ως όριο. Όσοι καλέσουν αργότερα θα εξυπηρετηθούν στα δρομολόγια της επόμενης μέρας.

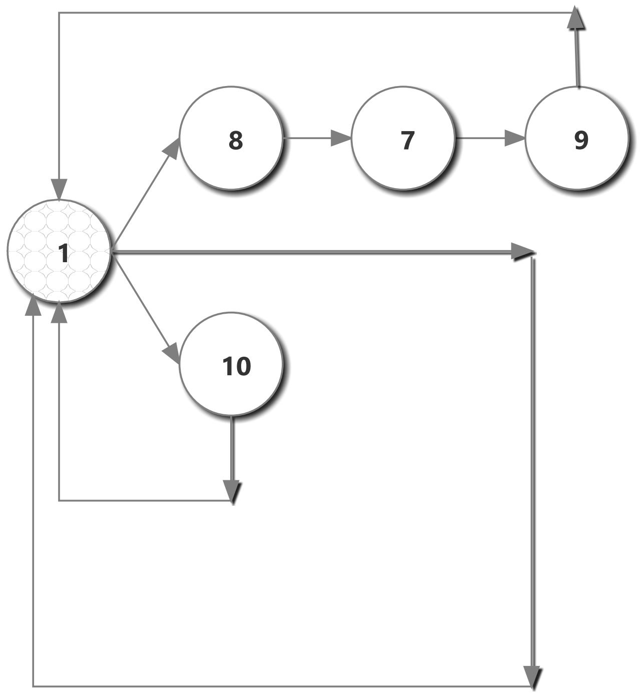
Βήμα 2

Από τις αρχικές διαδρομές που έχουν προκύψει από το στατικό πρόβλημα αφαιρούμε όλους τους πελάτες που δεν ξέρουμε αν θα εξυπηρετηθούν, δηλαδή τους πελάτες που θα καλέσουν μέχρι κάποια χρονική στιγμή. Η χρονική στιγμή που επιλέχθηκε για το συγκεκριμένο παράδειγμα είναι 75. Άρα τους κόμβους 2,3,4,5,6. Έτσι, οι διαδρομές που προκύπτουν είναι οι ακόλουθες:

Πρώτη διαδρομή=[1,8,7,9,1]

Δεύτερη διαδρομή=[1,1]

Τρίτη διαδρομή=[1,10,1]



Εικόνα 11: Οι κυκλικές διαδρομές αφού έχουν αφαιρεθεί οι πελάτες που θα εξυπηρετηθούν αργότερα

- **<u>Βήμα 3</u>**

Από τις νέες διαδρομές που προκύπτουν υπολογίζουμε το κόστος, το routtime, και τη χωρητικότητα κάθε διαδρομής μετά την αφαίρεση των κόμβων.

Πρώτη διαδρομή

- Κόστος1=απόσταση (1,8) + απόσταση (8,7) + απόσταση (7,9) + απόσταση (9,1)=11+14+46+24=95
- Ζήτηση1=[50,40,25,20]
- Routtime1=[0,16,45,101]
- Διαθέσιμο προϊόν=20

Δεύτερη διαδρομή

- Κόστος2= απόσταση(1,1)=0
- Ζήτηση2=[0]
- Routtime2=[0]
- Διαθέσιμο προϊόν=50

Τρίτη διαδρομή

- Κόστος3= απόσταση (1,10) + απόσταση (10,1)=34+34=68
- Ζήτηση3=[25]
- Routtime3=[0,59]
- Διαθέσιμο προϊόν=25

- **Βήμα 4**

Χωρίζουμε το πρώτο μισό του χρόνου σε χρονικά διαστήματα (updates) που προκύπτουν από το timestep που δίνεται από τα δεδομένα του προβλήματος. Στο συγκεκριμένο σημείο είχαμε δύο τρόπους για να αντιμετωπίσουμε το πρόβλημα της εμφάνισης μιας καινούριας ζήτησης. Ο ένας τρόπος είναι να κάνουμε συνεχή ενημερώσεις των διαδρομών κάθε φορά που θα ερχόταν μια καινούρια ζήτηση. Ο συγκεκριμένος τρόπος δεν είναι λάθος, αλλά είναι απαγορευτικός λόγω του ότι δεν μας είναι γνωστός ο αριθμός των καινούριων ζητήσεων και άρα ο αριθμός των φορών που θα πρέπει να εκτελεστεί ο αλγόριθμος προσεγγίζει το άπειρο. Άρα άμεσα ο αλγόριθμος γίνεται αργός και μη αποτελεσματικός. Ο τρόπος που συνήθως αντιμετωπίζεται ένα τέτοιο πρόβλημα σε δυναμικά προβλήματα βελτιστοποίησης είναι να χωρίζουμε τον χρονικό ορίζοντα σε διακριτά χρονικά διαστήματα. Αν τα χρονικά διαστήματα είναι πολύ μικρά πλησιάζουμε την προηγούμενη περίπτωση. Στην περίπτωση μας θα γίνουν 3 updates καθώς χωρίσαμε τον χρόνο t=75 σε timestep=3 χρονικά διαστήματα.

- **<u>Βήμα 5</u>**

Στο τέλος των χρονικών διαστημάτων θα γίνουν τα updates του προβλήματος. Δηλαδή θα εξετάσουμε στο χρονικό διάστημα που διαρκεί το update, ποιοι πελάτες έχουν εξυπηρετηθεί από αυτούς που έχουν πάρει τηλέφωνο για να εξυπηρετηθούν.

- **<u>Βήμα 6</u>**

Κάθε ένας από τους πελάτες που τηλεφωνεί θα πρέπει να προστεθεί σε κάποια από τις κυκλικές διαδρομές. Αρχικά γίνεται έλεγχος αν η χωρητικότητα του οχήματος επαρκεί ώστε να ικανοποιηθεί η ζήτηση που έχει ο πελάτης που τηλεφώνησε. Εάν υπάρχει διαθέσιμος χώρος, βρίσκουμε ποιος είναι ο τελευταίος πελάτης που έχει εξυπηρετηθεί τη χρονική στιγμή που γίνεται η ενημέρωση και αμέσως μετά από αυτόν προσθέτουμε τον πελάτη που τηλεφώνησε.

Για τη νέα διαδρομή υπολογίζουμε κόστος, routtime και τη συνολική ζήτηση. Αφού ελέγξουμε αυτή τη διαδρομή τοποθετούμε τον πελάτη μετά από όλους τους υπόλοιπους κόμβους της διαδρομής υπολογίζοντας για κάθε πιθανή διαδρομή το κόστος και επιλέγουμε τη βέλτιστη, η οποία μας δίνει τη μικρότερη αύξηση κόστους αφού γίνει έλεγχος ότι ο χρόνος της διαδρομής δεν ξεπερνάει το maxroute. Το ίδιο γίνεται για όλες τις υπόλοιπες κυκλικές διαδρομές. Τοποθετούμε όλους τους πελάτες στις βέλτιστες θέσεις και υπολογίζουμε το τελικό κόστος της διαδρομής.

Υλοποίηση updates

Update1 → t=(0,25]

Μέσα στο χρονικό διάστημα t έχουν τηλεφωνήσει για να εξυπηρετηθούν οι πελάτες 2 και 3.

- Για τον πελάτη 2:

Σε κάθε διαδρομή βλέπουμε ποιος είναι ο τελευταίος πελάτης που εξυπηρετήθηκε τη χρονική στιγμή που γίνεται η ενημέρωση (από το πίνακα routtime). Στην πρώτη διαδρομή ο τελευταίος που εξυπηρετήθηκε είναι ο πελάτης 8 (routtime=16). Στην δεύτερη διαδρομή υπάρχει μόνο μία πιθανή θέση. Στην τρίτη διαδρομή δεν έχει εξυπηρετηθεί κανένας πελάτης άρα οι πιθανές θέσεις που μπορεί να τοποθετηθεί ο πελάτης 2 θα είναι όλες μετά την αποθήκη.:

Διαδρομή1=[1,8,2,7,9,1]   Διαδρομή2=[1,2,1]

Διαδρομή1=[1,8,7,2,9,1]   Διαδρομή3=[1,2,10,1]

Διαδρομή1=[1,8,7,9,2,1]   Διαδρομή3=[1,10,2,1]

Πραγματοποιείται έλεγχος για τις πιθανές διαδρομές αν είναι εφικτές και από τις εφικτές ποια έχει τη μικρότερη αύξηση κόστους.

Πρώτη διαδρομή

Πραγματοποιείται έλεγχος για την πρώτη διαδρομή αν είναι εφικτό να προστεθεί ο κόμβος 2: Η χωρητικότητα του οχήματος είναι 20 μονάδες προϊόντος και η ζήτηση του πελάτη 2 είναι 20 μονάδες προϊόντος, άρα ως προς την ζήτηση ο πελάτης 2 είναι εφικτό να μπει στην διαδρομή 1.

Διαδρομή1=[1,8,2,7,9,1]

Routtime=184>150 άρα η διαδρομή είναι μη εφικτή.

Διαδρομή1=[1,8,7,2,9,1]

Routtime=139<150. Η διαδρομή εφικτή ως προς τον χρόνο.

Κόστος=89

Διαδρομή1=[1,8,7,9,2,1]

Routtime=138<150. Η διαδρομή εφικτή ως προς τον χρόνο.

Κόστος=89.

Δεύτερη διαδρομή

Διαδρομή2=[1,2,1]

Routtime=63<150. Η διαδρομή εφικτή ως προς τον χρόνο.

Κόστος=24.

Η χωρητικότητα του οχήματος είναι η μέγιστη και ο πελάτης έχει ζήτηση 20 μονάδες προϊόντος, άρα ως προς την ζήτηση ο πελάτης 2 είναι εφικτό να μπει στην διαδρομή 2.

Τρίτη διαδρομή

Διαδρομή3=[1,2,10]

Έλεγχος για την τρίτη διαδρομή αν είναι εφικτό να προστεθεί ο κόμβος 2: Η χωρητικότητα του οχήματος είναι 25 μονάδες προϊόντος και η ζήτηση του πελάτη 2 είναι 20 μονάδες προϊόντος, άρα ως προς την ζήτηση ο πελάτης 2 είναι εφικτό να μπει στην διαδρομή 3.

Αν ο κόμβος που ξεκινάει το όχημα έχει μικρότερο routtime από τη χρονική στιγμή του update, θεωρούμε ότι έχει ξεκινήσει τη στιγμή που ισούται με το άνω άκρο του update.

Routtime=124<150. Η διαδρομή εφικτή ως προς τον χρόνο.

Κόστος=74.

Διαδρομή3=[10,2,1]

Routtime=109<150. Η διαδρομή εφικτή ως προς τον χρόνο.

Κόστος=74.

Από όλες τις εφικτές διαδρομές διαλέγω αυτή που αυξάνει λιγότερο το συνολικό κόστος. Αν δύο διαδρομές έχουν το ίδιο κόστος διαλέγω τυχαία μια.

Η βέλτιστη θέση για να εξυπηρετηθεί ο κόμβος 2 είναι διαδρομή

Διαδρομή1=[1,8,7,9,2,1]

Routtime=[0,16,45,101,127]

Κόστος=89

Διαθέσιμο προϊόν=[50,40,25,20,0]

Διαδρομή2=[1,1]

Routtime=[0]

Κόστος=0

Διαθέσιμο προϊόν=50

Διαδρομή3=[1,10,1]

Routtime=[0,49,83]

Κόστος=68

Διαθέσιμο προϊόν=[50,25]

- Για τον πελάτη 3:

Οι πιθανές θέσεις που μπορεί να τοποθετηθεί ο κόμβος 3 είναι:

Διαδρομή1=[1,8,3,7,9,2,1] Διαδρομή2=[1,3,1]

Διαδρομή1=[1,8,7,3,9,2,1] Διαδρομή3=[1,3,10]

Διαδρομή1=[1,8,7,9,3,2,1] Διαδρομή3=[10,3,1]

Διαδρομή1=[1,8,7,9,2,3,1]

Πρώτη διαδρομή

Έλεγχος για την πρώτη διαδρομή αν είναι εφικτό να προστεθεί ο κόμβος 3: Η χωρητικότητα του οχήματος είναι 0 μονάδες προϊόντος και η ζήτηση του πελάτη 3 είναι 10 μονάδες προϊόντος, άρα ως προς την ζήτηση ο πελάτης 3 δεν είναι εφικτό να μπει στην διαδρομή 1.

Δεύτερη διαδρομή

Έλεγχος για την δεύτερη διαδρομή αν είναι εφικτό να προστεθεί ο κόμβος 3: Η χωρητικότητα του οχήματος είναι 50 μονάδες προϊόντος και η ζήτηση του πελάτη 3 είναι 10 μονάδες προϊόντος, άρα ως προς την ζήτηση ο πελάτης 3 είναι εφικτό να μπει στην διαδρομή 2.

Διαδρομή2=[1,3,1]

Κόστος=38

Τρίτη διαδρομή

Έλεγχος για την τρίτη διαδρομή αν είναι εφικτό να προστεθεί ο κόμβος 3: Η χωρητικότητα του οχήματος είναι 25 μονάδες προϊόντος και η ζήτηση του πελάτη 3 είναι 10 μονάδες προϊόντος, άρα ως προς την ζήτηση ο πελάτης 3 είναι εφικτό να μπει στην διαδρομή 3.

Διαδρομή3=[1,3,10,1]

Routtime=[0,59,117,151]

Κόστος=28

Η διαδρομή είναι μη εφικτή καθώς δεν επαρκεί ο χρόνος για να γυρίσει στην αποθήκη. (Την στιγμή 117 έχει φύγει από τον κόμβο 10, και 34 μονάδες μήκους που χρειάζεται για να γυρίσει πίσω στην αποθήκη ξεπερνάνε το μέγιστο χρόνο της διαδρομής).

Διαδρομή3=[1,10,3,1]

Κόστος=28

Routtime=[0,49,107,126]

Άρα ο κόμβος 3 θα μπει στην τρίτη διαδρομή.

Διαδρομή1=[1,8,7,9,2,1]

Routtime=[0,16,45,101,127]

Κόστος=99

Διαδρομή2=[1,1]

Routtime=[0]

Κόστος=0

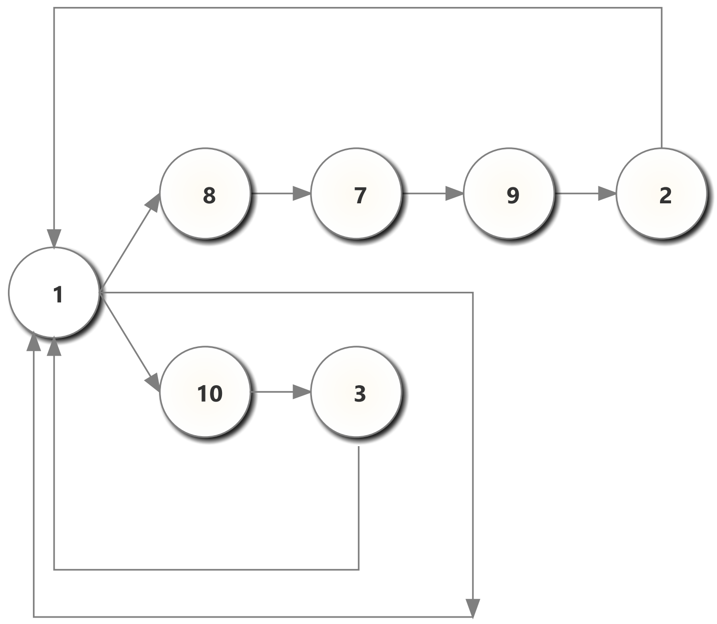
Διαδρομή3=[1,10,3,1]

Routtime=[0,49,107,126]

Κόστος=96

Διαθέσιμο προϊόν=[50,25,15]

Τέλος του πρώτου update. Παρατηρούμε ότι η πρώτη διαδρομή δεν έχει διαθέσιμη χωρητικότητα για να εξυπηρετήσει έναν νέο κόμβο.



Εικόνα 12 Οι διαδρομές του πρώτου update

Update2 → t=(25,50]

Μέσα στο χρονικό διάστημα t έχουν τηλεφωνήσει για να εξυπηρετηθούν οι πελάτες 4 και 5.

- Για τον πελάτη 4:

Οι πιθανές θέσεις που μπορεί να τοποθετηθεί ο κόμβος 4 είναι:

Διαδρομή2=[1,4,1] Κόστος=62

Routtime=[0,91,122]

Διαθέσιμο προϊόν=35

Διαδρομή3=[1,10,4,3,1] Κόστος=37

Routtime=[0,49,90,154,173]

Διαθέσιμο προϊόν=0

Διαδρομή3=[1,10,3,4,1] Κόστος=61

Routtime=[0,49,107,166,197]

Διαθέσιμο προϊόν=0

Μόνο η διαδρομή2 είναι εφικτή ως προς τον χρόνο και εφικτή και ως προς την χωρητικότητα.

Άρα για τον πελάτη 4:

Διαδρομή1=[1,8,7,9,2,1]

Διαδρομή2=[1,4,1]

Routtime=[0,41,72]

Κόστος=62

Διαθέσιμο προϊόν=[50,35]

Διαδρομή3=[1,10,3,1]

- Για τον πελάτη 5

Οι πιθανές θέσεις που μπορεί να τοποθετηθεί ο κόμβος 5 είναι:

Διαδρομή2=[1,10,5,3,1] Κόστος=103

Routtime=[0,49,69,120,139]

Ζήτηση=45

Διαθέσιμο προϊόν=5

Διαδρομή3=[1,10,3,5,1] Κόστος=135

Διαδρομή2=[1,5,4,1] Κόστος=73

Διαδρομή2=[1,4,5,1] Κόστος=73

Τη μικρότερη αύξηση κόστους έχει η Διαδρομή3=[1,10,5,3,1]. Είναι εφικτή διαδρομή ως προς τον χρόνο και ως προς την χωρητικότητα.

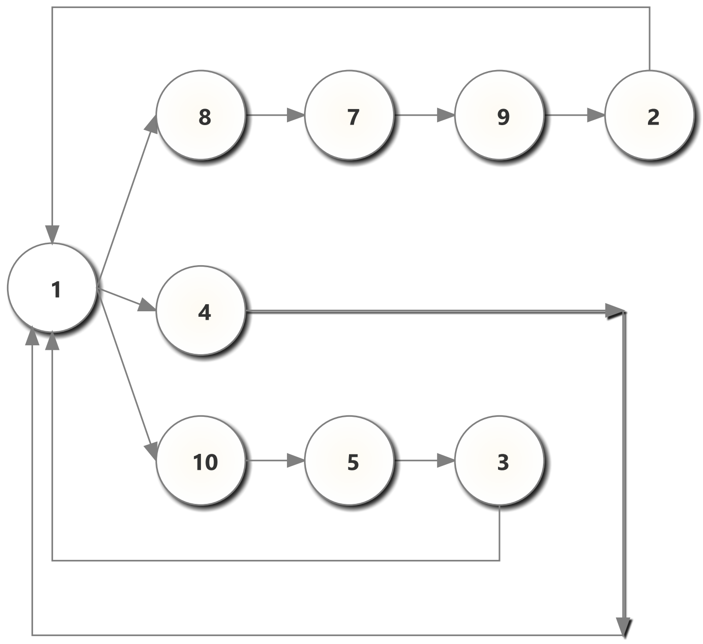
Άρα για τον πελάτη 5:

Διαδρομή1=[1,8,7,9,2,1]

Διαδρομή2=[1, 4,1]

Διαδρομή3=[1,10,5,3,1]

Τέλος δεύτερου update. Η δεύτερη διαδρομή έχει διαθεσιμότητα 5 μονάδες προϊόντος και η τρίτη διαδρομή έχει 35 μονάδες προϊόντος.



Εικόνα 13 Οι διαδρομές του δεύτερου update

Update3 → t=(50,75]

Μέσα στο χρονικό διάστημα t έχει τηλεφωνήσει για να εξυπηρετηθεί ο πελάτης 6.

Η ζήτηση του πελάτη 6, είναι 20 μονάδες προϊόντος, άρα μόνο στη δεύτερη διαδρομή είναι εφικτό να εξυπηρετηθεί.

Διαδρομή2=[1,6,4,1] Κόστος=68

Routtime=[0,101,122,132,163]

Διαθέσιμο προϊόν=15

Διαδρομή2=[1,4,6,1] Κόστος=68

Routtime=[0,91,122,132]

Διαθέσιμο προϊόν=15

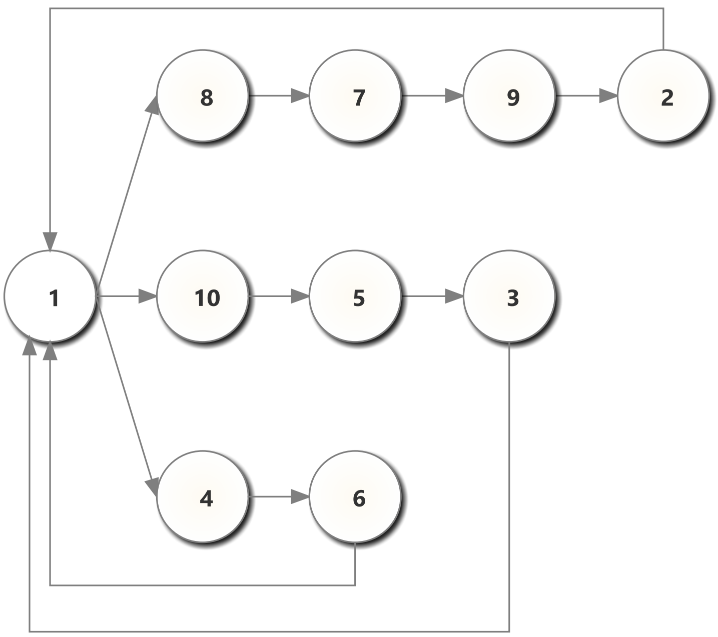
Η μόνη εφικτή διαδρομή ως προς τον περιορισμό του χρόνου είναι η Διαδρομή2=[1,6,4,1]. Τέλος τρίτου update. Άρα οι βέλτιστες διαδρομές είναι

Διαδρομή1=[1,8,7,9,2,1]   Κόστος=99

Διαδρομή3=[1,10,5,3,1]   Κόστος=103

Διαδρομή2=[1,4,6,1]   Κόστος=68

Το συνολικό κόστος διαδρομής δυναμικού προβλήματος που προκύπτει είναι 270 μονάδες κόστους. Το κόστος του δυναμικού προβλήματος είναι αυξημένο σε σχέση με το αντίστοιχο στατικό πρόβλημα το οποίο είναι αναμενόμενο διότι το δυναμικό πρόβλημα παρουσιάζει αυξημένη πολυπλοκότητα σε σχέση με το στατικό πρόβλημα.



Εικόνα 14: Οι τελικές διαδρομές

3.4.4 Επίλυση δυναμικού προβλήματος σε περιβάλλον matlab

Επιλύθηκε ένα ακόμα πρόβλημα με περισσότερους κόμβους σε περιβάλλον matlab.

Τα δεδομένα του προβλήματος είναι τα εξής:

Αριθμός κόμβων:75

Αριθμός οχημάτων:1

Μέγιστο μήκος διαδρομής:769

Χρονικά διαστήματα που γίνεται update της διαδρομής (timesteps):15

Επιπλέον στα δεδομένα του προβλήματος περιλαμβάνονται οι συντεταγμένες των κόμβων, η ζήτηση και ο χρόνος εξυπηρέτησης του κάθε κόμβου όπως και η χρονική στιγμή που έχει καλέσει ο καθένας ώστε να εξυπηρετηθεί. Τα βήματα που ακολουθούνται για την επίλυση του προβλήματος είναι τα ίδια με του ανωτέρω αναλυτικού παραδείγματος.

Τα αποτελέσματα του προέκυψαν είναι τα εξής.

Το στατικό πρόβλημα έχει τελικό κόστος 2079,3 μονάδες κόστους ενώ το δυναμικό 2639,6 μονάδες κόστους. Ακολούθως παρουσιάζεται η εξέλιξη του κόστους καθώς εξελίσσονται τα updates του προβλήματος

Αύξοντος αριθμός update	Κόστος (μονάδες κόστους)
1	1569,9
2	1684,1
3	1684,1
4	1734,6
5	1888,8

6	2078,4
7	2085,4
8	2142,3
9	2186,1
10	2191,9
11	2224,1
12	2245,9
13	2290,9
14	2290,5
15	2639,6

Εικόνα 15 εξέλιξη κόστους καθώς εξελίσσονται τα updates

Πραγματοποιήθηκαν πολλές παραλλαγές του προβλήματος όπως για παράδειγμα να αλλάξει ο αριθμός των updates, ώστε να διαπιστωθεί αν το τελικό κόστος μεταβάλλεται αν μεταβληθεί ο αριθμός τους.

Σε κάθε περίπτωση το τελικό κόστος παρέμεινε το ίδιο υποδηλώνοντας σταθερότητα του αλγόριθμου. Με αυτόν τον τρόπο επιτυγχάνεται και ένας από τους στόχους που τέθηκαν πριν την κατασκευή του αλγορίθμου και σχετίζεται με την σταθερότητα αυτού.

Αριθμός update	Κόστος (μονάδες κόστους)
5	2639,6
10	2639,6
15	2639,6
20	2639,6
25	2639,6

30	2639,6
35	2639,6
40	2639,6

Εικόνα 16 Το κόστος παραμένει αμετάβλητο καθώς αλλάζει ο αριθμός των updates

Επιπλέον πραγματοποιήθηκε αλλαγή στην συνθήκη επιλογής βέλτιστης θέσης εισαγωγής του κόμβου που πρόκειται να εξυπηρετηθεί. Αντί να επιλεχθεί η θέση που το κόστος της διαδρομής παρουσιάσει τη μικρότερη αύξηση, έγινε υπολογισμός του κόστους για όλες τις πιθανές θέσεις και επιλέχθηκε αυτή η θέση που το κόστος της διαδρομής είναι το ελάχιστο. Το αποτέλεσμα ήταν να μειωθεί το κόστος της διαδρομής σε 2348 μονάδες κόστους.

4 Σύνοψη-Συμπεράσματα

Ο στόχος της παρούσας εργασίας είναι να μετατραπεί το στατικό πρόβλημα δρομολόγησης οχημάτων σε ένα πραγματικού χρόνου δυναμικό πρόβλημα, μέσω ενός απλού, γρήγορου, σταθερού και αποτελεσματικού αλγορίθμου.

Η πολυπλοκότητα του δυναμικού προβλήματος παρατηρείται όταν κατά την εξέλιξη των διαδρομών που πραγματοποιεί ένα όχημα υπάρχει δυναμική ζήτηση και ενδεχομένως να αλλάξει η κατεύθυνση που θα ακολουθήσει το όχημα κατά την εξυπηρέτηση των πελατών. Τότε θα πρέπει να επανασχεδιαστούν σε πραγματικό χρόνο κάθε φορά. Δηλαδή δεν δύναται να γίνει ο σχεδιασμός των διαδρομών a priori, άλλα γίνεται σταδιακά σε πραγματικό χρόνο.

Το ερευνητικό μας ενδιαφέρον είναι να παρατηρήσουμε πως τα στατικά μοντέλα μπορούν να μετατραπούν σε πραγματικού χρόνου προβλήματα και πως αντιμετωπίζεται η δυναμική ζήτηση όταν προκύπτει. Σε πραγματικές συνθήκες είναι σύνηθες να συμβαίνει ένα έκτακτο γεγονός (ακύρωση παραγγελίας, προσθήκη νέας, έκτακτες καταστάσεις, κτλ) οι οποίες θα πρέπει να αντιμετωπιστούν με επιτυχία.

Στην παρούσα εργασία ο αλγόριθμος που αναπτύξαμε είναι δύο φάσεων. Την πρώτη φάση αποτελεί το στατικό πρόβλημα στο οποίο οι διαδρομές σχεδιάζονται εκ των προτέρων και δεν αλλάζουν μέχρι το τέλος του προβλήματος. Σχεδιάστηκαν οι διαδρομές με τον αλγόριθμο του πλησιέστερου γείτονα λαμβάνοντας υπόψη τους περιορισμούς του προβλήματος. Σε αυτή τη φάση του προβλήματος προκύπτει η αρχική λύση του προβλήματος.

Στη συνέχεια μέσω του μεθευρετικού αλγορίθμου προσομοιωμένη ανόπτηση σε συνδυασμό με τοπική αναζήτηση με τους ευρετικούς αλγορίθμους 1-1 exchange, 1-0 relocate, προς τα πίσω 1-0 relocate, 2 opt βελτιώθηκε η αρχική λύση του προβλήματος.

Στη δεύτερη φάση του προβλήματος οι βέλτιστες διαδρομές σχεδιάζονται σε πραγματικό χρόνο. Χωρίζεται ο χρόνος σε χρονικά διαστήματα (timesteps). Γίνεται έλεγχος ποιοι έχουν εξυπηρετηθεί μέσα σε αυτό το χρονικό διάστημα και ποιοι από αυτούς που θέλουν να εξυπηρετηθούν είναι εφικτό να προστεθούν στη διαδρομή στη βέλτιστη θέση. Η βέλτιστη θέση καθορίζεται από την ελάχιστη αύξηση του κόστους σε κάθε διαδρομή. Στο τέλος κάθε update γίνεται επανασχεδιασμός των διαδρομών. Η ίδια διαδικασία συνεχίζεται μέχρι να εξυπηρετηθούν όλοι οι πελάτες ακριβώς μία φορά.

Έγιναν δοκιμές στον αριθμό των updates για να διαπιστωθεί αν έχει μεταβληθεί το κόστος. Το κόστος παρέμεινε σταθερό το οποίο υποδηλώνει τη σταθερότητα του αλγορίθμου.

Η παρούσα εργασία αποτελεί τη βάση για να συνεχίσουμε τη μελέτη τέτοιου είδους προβλημάτων σε πραγματικό χρόνο ώστε να μελετηθεί σε βάθος και να βρεθεί ένας βέλτιστος τρόπος αντιμετώπισης έκτακτων καταστάσεων.

5 Βιβλιογραφία

1. Mangan, J., C. Lalwani, and C.L. Lalwani, *Global logistics and supply chain management*. 2016: John Wiley & Sons.
2. Μουρκούσης, Γ., *Μεθοδολογίες ανάπτυξης πληροφοριακών συστημάτων υποστήριξης λήψης αποφάσεων για την δρομολόγηση στόλου οχημάτων και την αξιολόγηση εντολών μεταφοράς φορτίου στον τομέα των logistics*. 2009.
3. Cirovic´, G.C., Pamukar, D. , and D. Bozanic , *Green logistic vehicle routing problem: Routing light delivery vehicles in urban areas using a neuro-fuzzy model.* Expert Systems with Applications, 41, 9, 4245-4258, 2014.
4. Lambert, D.M., M.C. Cooper, and J.D. Pagh, *Supply chain management: implementation issues and research opportunities.* The international journal of logistics management, 1998. **9**(2): p. 1-20.
5. Mentzer, J.T., et al., *Defining supply chain management.* Journal of Business logistics, 2001. **22**(2): p. 1-25.
6. Tan, K.C., *A framework of supply chain management literature.* European Journal of Purchasing & Supply Management, 2001. **7**(1): p. 39-48.
7. Pillac, V., et al., *A review of dynamic vehicle routing problems.* European Journal of Operational Research, 2013. **225**(1): p. 1-11.
8. Reed, M., A. Yiannakou, and R. Evering, *An ant colony algorithm for the multi-compartment vehicle routing problem.* Applied Soft Computing, 2014. **15**: p. 169-176.
9. Marinakis, Y., A. Migdalas, and P.M. Pardalos, *A new bilevel formulation for the vehicle routing problem and a solution method using a genetic algorithm.* Journal of Global Optimization, 2007. **38**(4): p. 555-580.

10. Berbeglia, G., et al., *Static pickup and delivery problems: a classification scheme and survey.* Top, 2007. **15**(1): p. 1-31.
11. Jaillet, P. and M.R. Wagner, *Generalized online routing: New competitive ratios, resource augmentation, and asymptotic analyses.* Operations research, 2008. **56**(3): p. 745-757.
12. Ghiani, G., et al., *Real-time vehicle routing: Solution concepts, algorithms and parallel computing strategies.* European Journal of Operational Research, 2003. **151**(1): p. 1-11.
13. Gendreau, M. and J.-Y. Potvin, *Dynamic vehicle routing and dispatching*, in *Fleet management and logistics*. 1998, Springer. p. 115-126.
14. Τάτση, Β., *Ενιαία Επίλυση ενός Προβλήματος Δρομολόγησης Οχημάτων-Διαχείρισης Αποθέματος με Χρήση του Αλγορίθμου Ant Colony Optimization.*
15. Fleszar, K., I.H. Osman, and K.S. Hindi, *A variable neighbourhood search algorithm for the open vehicle routing problem.* European Journal of Operational Research, 2009. **195**(3): p. 803-809.
16. Bräysy, O. and M. Gendreau, *Vehicle routing problem with time windows, Part II: Metaheuristics.* Transportation science, 2005. **39**(1): p. 119-139.
17. Βλάχος, Α., *Μετα-ευρεστικοί αλγόριθμοι βελτιστοποίησης και εφαρμογές σε προβλήματα συνδυαστικής βελτιστοποίησης.* 2006.
18. Baker, B.M. and M. Ayechew, *A genetic algorithm for the vehicle routing problem.* Computers & Operations Research, 2003. **30**(5): p. 787-800.
19. Marinakis, Y., et al., *Ant colony and particle swarm optimization for financial classification problems.* Expert Systems with Applications, 2009. **36**(7): p. 10604-10611.
20. Khouadjia, Mostepha R., et al. "A comparative study between dynamic adapted PSO and VNS for the vehicle routing problem with dynamic requests." *Applied Soft Computing* 12.4 (2012): 1426-1439.

21. https://www.eclass.tuc.gr/modules/document/?course=MPD234