ΠΟΛΥΤΕΧΝΕΙΟ ΚΡΗΤΗΣ
TECHNICAL UNIVERSITY OF CRETE

•Σχολή Μηχανικών Παραγωγής και Διοίκησης

Αλγόριθμος προσομοιωμένης ανόπτησης για την επίλυση του προβλήματος δρομολόγησης σχολικών λεωφορείων

Συγγραφέας: Βαβελίδης Γεώργιος

Επιβλέπων καθηγητής: Δρ Μαρινάκης Ιωάννης

Χανιά Κρήτης Ιούλιος 2023

Ευχαριστίες

Ευχαριστώ θερμά τον καθηγητή μου, κ. Ιωάννη Μαρινάκη, για την πολύτιμη βοήθειά του καθώς και την καθοδήγηση του καθ' όλη την διάρκεια της διπλωματικής μου εργασίας.

Αφιερώνω την εργασία οικογένειά μου και τους κοντινούς μου φίλους η υποστήριξη των οποίων ήταν για εμένα καθοριστική κατά την διάρκεια των σπουδών μου στο Πολυτεχνείο Κρήτης.

Πίνακας περιεχομένων

Περίληψη

Κεφάλαιο 1. Εισαγωγή στην Εφοδιαστική Αλυσίδα

1.1 Εισαγωγή

1.2 Εφοδιαστική αλυσίδα

Κεφάλαιο 2. Μεταφορές και Διανομή

2.1 Εισαγωγή

2.2 Προβλήματα Δρομολόγησης οχημάτων

2.2.1 Εισαγωγή στα προβλήματα δρομολόγησης οχημάτων

2.2.2 Τα βασικά χαρακτηριστικά του προβλήματος

2.2.3 Τα δεδομένα του προβλήματος

2.2.4 Οι περιορισμοί του προβλήματος

2.2.5 Το κόστος των διαδρομών

2.2.6 Στόχοι για την επίλυση του προβλήματος

2.3.Εφαρμογές του προβλήματος δρομολόγησης οχημάτων

2.3.1 Προβλήματα δρομολόγησης πλοίων

2.3.2 Πρόβλημα δρομολόγησης οχημάτων στην βιομηχανία αποκομιδής σκουπιδιών

2.3.3 Πρόβλημα δρομολόγησης οχημάτων στην βιομηχανία ποτών, τροφών και γάλακτος

2.3.4 Δρομολόγηση και διανομή στην βιομηχανία παραγωγής εφημερίδων

Κεφάλαιο 3. Προβλήματα Δρομολόγησης Σχολικών λεωφορείων

3.1 Εισαγωγή στο πρόβλημα δρομολόγησης σχολικών λεωφορείων

3.1.1 Προετοιμασία δεδομένων

3.1.2 Επιλογή στάσεων λεωφορείων

3.1.3 Σχεδιασμός διαδρομών των λεωφορείων

3.1.4 Προσαρμογή χρόνου άφιξης μαθημάτων των σχολείων

3.1.5 Προγραμματισμός των λεωφορείων

3.2 Ταξινόμηση προβλημάτων δρομολόγησης σχολικών λεωφορείων

3.2.1 Αριθμός σχολείων: Μοναδικό σχολείο ή πολλαπλά σχολεία

3.2.2 Αστική ή προαστιακή περιοχή

3.2.3 Πρωί ή απόγευμα

3.2.4 Επιβίβαση μαθητών διαφορετικών σχολείων

3.2.5 Δρομολόγηση μαθητών με ειδικές ανάγκες

3.2.6 Ομογενής ή ετερογενής στόλος οχημάτων

3.2.7 Στόχος προβλήματος

3.2.8 Περιορισμός προβλήματος

Κεφάλαιο 4. Αλγόριθμοι Βελτιστοποίησης προβλημάτων της Εφοδιαστικής αλυσίδας

4.1 Ευρετικοί Αλγόριθμοι

4.1.1 Αλγόριθμοι απληστίας

4.1.2 Προσεγγιστικοί αλγόριθμοι

4.1.3 Αλγόριθμοι τοπικής αναζήτησης

4.2 Μεθευρετικοί Αλγόριθμοι

Κεφάλαιο 5. Προσομοιωμένη Ανόπτηση

Κεφάλαιο 6. Παρουσίαση προβλήματος και υλοποίηση αλγορίθμου

Κεφάλαιο 7. Παρουσίαση και σχολιασμός αποτελεσμάτων

Βιβλιογραφία

Περίληψη

Στην εποχή της έξαρσης της νόσου COVID-19 που προκαλείται από τον κορωνοϊό, είναι συνετό κάθε υπεύθυνος πολίτης να αποφεύγει τις περιττές μετακινήσεις ή αν είναι αναπόφευκτο και πρέπει να χρησιμοποιήσει τα Μέσα Μαζικής Μεταφοράς, συνίσταται να εκθέτει τον εαυτό του και τους γύρω του σε όσο το δυνατό μικρότερο κίνδυνο, χρησιμοποιώντας τα μέτρα προστασίας που όρισε η πολιτεία αλλά και ελαχιστοποιώντας τον χρόνο που βρίσκεται σε αυτά. Ωστόσο, ορισμένες κατηγορίες πολιτών, όπως οι μαθητές, πέρα των ατομικών μέτρων προστασίας, περνούν προκαθορισμένη χρονική διάρκεια καθημερινά εντός των σχολικών λεωφορείων αυξάνοντας έτσι το ρίσκο μετάδοσης της ασθένειας. Είναι συνετό επομένως και η ίδια η δρομολόγηση των λεωφορείων να προσαρμοστεί ώστε οι μαθητές να περνούν όσο το δυνατό λιγότερη ώρα σε κάθε διαδρομή.

Στην παρούσα διπλωματική εργασία ασχολούμαστε με το πρόβλημα δρομολόγησης σχολικών λεωφορείων (school bus routing problem). Σκοπός του προβλήματος είναι ο καταμερισμός των μαθητών σε στάσεις από τις οποίες θα περάσουν τα σχολικά λεωφορεία έτσι ώστε να μειωθεί ο συνωστισμός αλλά και η εισαγωγή μέγιστου αριθμού μαθητών εντός των λεωφορείων χωρίς να παραβιάζονται τα μέτρα προστασίας. Έπειτα, στόχος είναι ο μειωμένος χρόνος ταξιδιού μέχρι την άφιξη τους στο σχολείο. Όλα τα λεωφορεία, με αφετηρία το σχολείο, προσπαθούν να καλύψουν την συνολική ζήτηση (στάσεις) χωρίς να παραβιάζονται οι περιορισμοί της χωρητικότητας των λεωφορείων, της χωρητικότητας της κάθε στάσης αλλά και η μέγιστη απόσταση την οποία επιτρέπεται να διανύσει ένας μαθητής μέχρι να φτάσει στην στάση του.

Αρχικά μέσω ενός άπληστου ευρετικού αλγόριθμου θα δημιουργηθεί μια αρχική λύση η οποία όμως δεν είναι βέλτιστη. Θα χρησιμοποιηθεί ο αλγόριθμος προσομοιώμενης ανόπτησης σε συνδυασμό με αλγορίθμους τοπικής αναζήτησης για την εύρεση των βέλτιστων διαδρομών που θα ακολουθήσουν τα λεωφορεία.

Κεφάλαιο 1

Εισαγωγή στην Εφοδιαστική Αλυσίδα

1.1 Εισαγωγή

Οι διαρκώς εκτεινόμενες απαιτήσεις των πελατών και η συνεχής αύξηση του ανταγωνισμού είναι κάποιοι από τους καθοριστικούς παράγοντες για το αν μπορεί μια Εφοδιαστική Αλυσίδα να θεωρηθεί πλήρης και ανταποκρινόμενη η οποία θα στοχεύει στην ικανοποίηση των πελάτων και την εξασφάλιση αύξησης τόσο της κερδοφορίας όσο και των μεριδίων αγοράς. Επιπλέον, η ψηφιακή επανάσταση από την δεκαετία του ’80 μέχρι σήμερα, σηματοδοτώντας την έναρξη της Εποχής της Πληροφορίας, αλλάζει τα δεδομένα διότι πλέον ο επιχειρηματικός ανταγωνισμός ξεκινά να πραγματοποιείται σε επίπεδο Εφοδιαστικής Αλυσίδας στη θέση του επιπέδου επιχειρήσεων που ήταν έως τώρα, για τον λόγο ότι το e-Buisiness και η αλματώδης ανάπτυξη στην Τεχνολογία της Πληροφορικής θα έχουν καθοριστικό λόγο στους κανόνες και στο τι απαιτεί το Επιχειρείν.

Ο όρος Εφοδιαστική Αλυσίδα, περιλαμβάνει την ροή πληροφοριών ανάμεσα στα μέλη της ίδιας αλυσίδας, όχι μόνο δηλαδή την ροή υλικών από το επίπεδο πρώτων υλών (προμηθευτές, κατασκευαστές) ως τον τελικό καταναλωτή. Η διαχείριση της εφοδιαστικής αλυσίδας γίνεται σε δύο επίπεδα:

- **Επίπεδο προγραμματισμού**: Εδώ, μέσω της ανάλυσης δεδομένων προμηθειών, αναλώσεων παραγωγής πωλήσεων και αποθεματοποίησης, πραγματοποιούνται προβλέψεις και πλάνα όπου βασίζεται ο προγραμματισμός.
- **Επίπεδο εκτέλεσης**: Σε αυτό το επίπεδο, πραγματοποιείται η εκτέλεση του προκαθορισμένου πλάνου του παραπάνω επιπέδου και η στενή παρακολούθηση της εξέλιξης του, στηριζόμενη στα

δεδομένα και τις πληροφορίες, τα οποία συλλέγονται από όλο το εύρος της Εφοδιαστικής Αλυσίδας.

Στην Εφοδιαστική Αλυσίδα επίσης, υπάγονται και οι αποφάσεις και στρατηγικές ενός οργανισμού (τόσο οι διαδικασίες σχεδιασμού όσο και υλοποίησης), στηριζόμενες σε σύγχρονα μοντέλα και τεχνικές για αποτελεσματικότερες αποφάσεις, στην τεχνολογία της πληροφορικής και των επικοινωνιών και το επιχειρησιακό περιβάλλον. Για μια τόσο περίπλοκη τελική απόφαση λοιπόν, απαραίτητη προϋπόθεση αποτελεί το άρτια καλλιεργημένο κλίμα οργάνωσης, οι κατάλληλες ανθρώπινες και διοικητικές ικανότητες και η διατήρηση ενός άψογα λειτουργικού πληροφοριακού συστήματος και των δεδομένων.

Κάθε εφαρμογή της εφοδιαστικής αλυσίδας περιλαμβάνει τα παρακάτω βήματα:

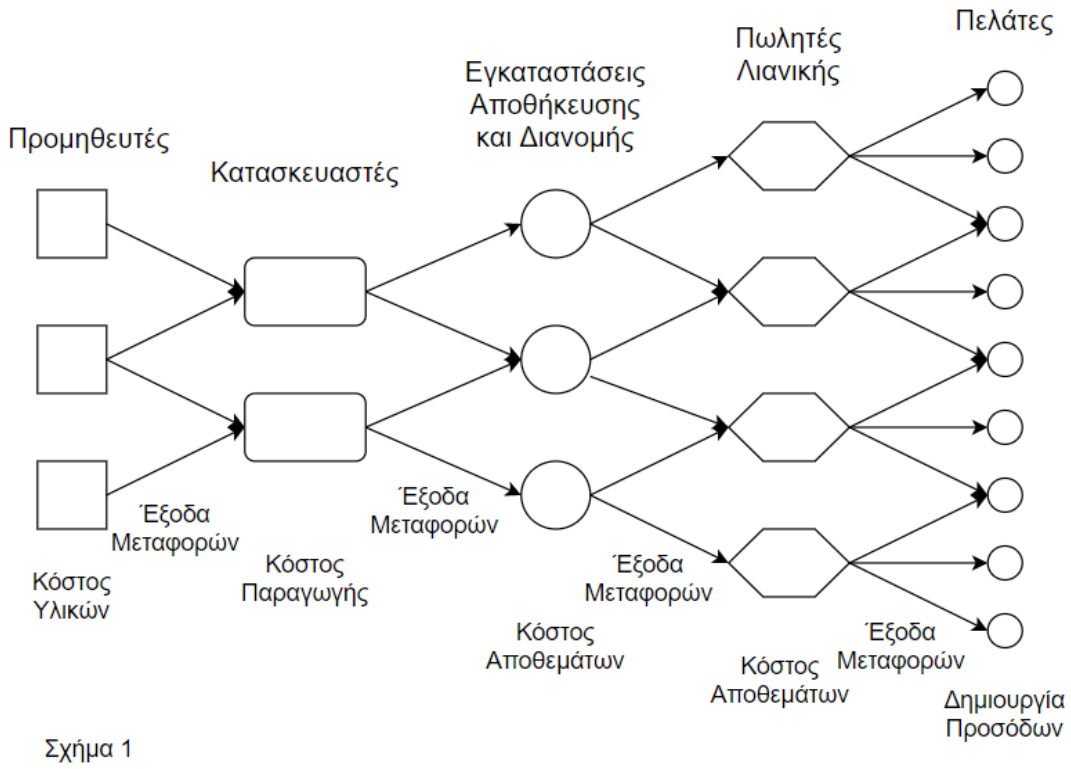
- Εντοπίζουμε το πρόβλημα και εφαρμόζουμε μεθόδους ανάλυσης για να θέσουμε τα όρια των συστατικών του, ή οτιδήποτε άλλο το επηρεάζει, και προσδιορίζουμε τους στρατηγικούς-επιχειρησιακούς στόχους και τους περιορισμούς που το διέπουν.
- Προτυποποιούμε μαθηματικά ή συστημικά για την ορθολογιστική απεικόνιση του προβλήματος.
- Αναπτύσσουμε (ή επιλέγουμε υπάρχοντες) τεχνικές επίλυσης των προβλημάτων μέσω μαθηματικού προγραμματισμού, ευρετικών αλγορίθμων ή άλλων υπολογιστικών μεθόδων ώστε να είναι εφικτή η σύγκλιση σε μία εφικτή ή/και βέλτιστη λύση.
- Υλοποιούμε τις τεχνικών επίλυσης σε κατάλληλη πλατφόρμα που εξαρτάται από την πληροφοριακή υποδομή της αντίστοιχης εταιρείας, την ύπαρξη πακέτων λογισμικού, και τις ανάγκες για αποτελεσματικότητα και ταχύτητα στην εξεύρεση λύσεων.

- Σχεδιάζουμε και συνθέτουμε υποστηρικτικά πληροφοριακά συστήματα που ενσωματώνουν τις τεχνικές επίλυσης και επιτρέπουν την διεπαφή των μάνατζερ που πρέπει να λάβουν τις αποφάσεις και των συστημάτων που περιέχουν τα δεδομένα και τις αλγοριθμικές προσεγγίσεις.

Οι αρχές της Εφοδιαστικής Αλυσίδας εφαρμόζονται σε πληθώρα προβλημάτων όπως η Χρηματοοικονομική Διοίκηση, το μάρκετινγκ, οι πωλήσεις, στην πληροφορική, τις μεταφορές, ακόμη και στους ανθρώπινους πόρους.

1.2 Εφοδιαστική Αλυσίδα

Μια εφοδιαστική Αλυσίδα, ή ένα δίκτυο εφοδιαστικής, λοιπόν, αποτελείται από όλα τα εμπλεκόμενα στάδια (έμμεσα ή άμεσα) και αποσκοπούν στην ικανοποίηση των απαιτήσεων του πελάτη. Συνεπώς κατασκευαστές, προμηθευτές, χώροι αποθήκευσης, κέντρα διανομής, πωλητές λιανικής, μεταφορείς, πελάτες όπως επίσης και πρώτες ύλες αλλά και αποθέματα που προήλθαν από την διαδικασία παραγωγής και τελικά προϊόντα που μεταφέρονται ανάμεσα σε όλους τους παραπάνω, αποτελούν την εφοδιαστική αλυσίδα. (Σχήμα 1).



Σχήμα 1

Στην εφοδιαστική αλυσίδα, όντας δυναμική, η ροή των προϊόντων, των πληροφοριών αλλά ακόμη και κεφαλαίων εμπλέκονται διαρκώς ανάμεσα στα διάφορα στάδια. Ωστόσο, δεν είναι πάντα αληθές πως κάθε διαφορετικό στάδιο της αλυσίδας είναι και πάντα παρόν σε όλες τις εφοδιαστικές αλυσίδες. Αυτό που εν τέλει καθορίζει τον κατάλληλο σχεδιασμό της εφοδιαστικής αλυσίδας, είναι πρωτίστως το τι απαιτούν οι πελάτες αλλά και δευτερεύοντος ο ρόλος που τα στάδια της διαδραματίζουν στην προσπάθεια ικανοποίησης των εν λόγω απαιτήσεων. Παρατηρώντας το παραπάνω σχήμα, βλέπουμε πως η εφοδιαστική αλυσίδα περιλαμβάνει κάθε ξεχωριστό έξοδο που δημιουργείται στο κάθε στάδιο, όπως επίσης και τα έξοδα όταν τα εκάστοτε στάδια αλληλοεπιδρούν μεταξύ τους. Άξιο αναφοράς είναι το γεγονός ότι σε κάθε εφοδιαστική αλυσίδα, ο πελάτης αποτελεί την μοναδική πηγή προσόδων. Αντιθέτως, δαπάνες δημιουργούνται από τις ροές πληροφοριών, των προϊόντων, ή των κεφαλαίων.

Τα βασικά κριτήρια για να είναι αποδοτική η εφοδιαστική αλυσίδα είναι τα παρακάτω:

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

Η εφοδιαστική αλυσίδα σκοπεύει να φτάσει την ολική αξία, , η οποία αναφέρεται στη διαφορά μεταξύ της αξίας του τελικού προϊόντος για τον πελάτη και της καταβληθέντας προσπάθειας από την εφοδιαστική αλυσίδα προς ικανοποίηση της απαίτησης του πελάτη, στο μέγιστο δυνατό. Συνεπώς, στόχος της διαχείρισης της εφοδιαστικής αλυσίδας είναι να μεγιστοποιήσει την εφικτότητα και αποτελεσματικότητα, όσων αφορά το κόστος, σε κάθε ξεχωριστό στάδιο του συστήματος και να ελαχιστοποιήσει το ολικό κόστος, που προέρχεται κυρίως από τις μεταφορές, την διανομή, κλπ. Με τον ίδιο τρόπο, λέγοντας ολικό κέρδος της εφοδιαστικής αλυσίδας αναφερόμαστε σε όλο το κέρδος που μοιράζεται σε όλο το μήκος της αλυσίδας. Συνεπώς, το να διαχειριζόμαστε τις ανάμεσα στα στάδια της εφοδιαστικής αλυσίδας προς μεγιστοποίησης του ολικού κέρδους αποτελεί βασικότατο σκοπό της διαχείρισης της εφοδιαστικής αλυσίδας. Ως χρονικό κύκλο αναφέρουμε τον απαιτούμενο ολικό χρόνο για την ολοκλήρωση της όλης της διαδικασίας με αφετηρία τις πρώτες ύλες και τερματισμό στο εμπορεύσιμο προϊόν στο πελάτη.

Κεφάλαιο 2

Μεταφορές και διανομή

2.1 Εισαγωγή

Η μεταφορά των κατάλληλων προϊόντων ή υπηρεσιών την κατάλληλη χρονική στιγμή στο κατάλληλο μέρος στην επιθυμητή κατάσταση και συγχρόνως η πραγματοποίηση μέγιστης απόδοσης για την εταιρεία αποτελούν την αποστολή της εφοδιαστικής αλυσίδας. Συνδετικό κρίκο ανάμεσα στην παραγωγή και την κατανάλωση αποτελούν οι δραστηριότητες της εφοδιαστικής αφού τοποθεσίες παραγωγής και τοποθεσίες αγοραστών ή προμηθευτών ενώνονται.

Από τις πιο κοστοβόρες δραστηριότητες της εφοδιαστικής αλυσίδας για μια επιχείρηση θεωρούνται τα αποθέματα και οι μεταφορές, ενώνουν την παραγωγή με την αποθήκευση και τη κατανάλωση. Οι μεταφορές απορροφούν τεράστιο μέρος του συνολικού κόστους. Συνεπώς, κύριο στοιχείο της εφοδιαστικής αλυσίδας αποτελεί το να υπάρχουν διαθέσιμοι αποτελεσματικοί τρόποι μεταφοράς. Στοιχεία πυλώνες οποιουδήποτε συστήματος μεταφορών αποτελούν οι εγκαταστάσεις, ο λειτουργικός εξοπλισμός και το ανθρώπινο δυναμικό.

Οι μεταφορές χωρίζονται στα παρακάτω μέρη:

- **Εσωτερικές μεταφορές**. Πρώτες ύλες από τις πηγές προς τα εργοστάσια, αλλά και μέρους των τελικών προϊόντων από εργοστάσιο σε εργοστάσιο της ίδιας εταιρείας αλλά και τελικών προϊόντων από τις παραγωγικές μονάδες στους αποθηκευτικούς χώρους ή στις τοποθεσίες που πραγματοποιούνται οι πωλήσεις.

- **Εξωτερικές μεταφορές.** Μεταφέρουν τα πλέον εμπορεύσιμα προϊόντα από τους αποθηκευτικούς χώρους στους πελάτες άμεσα, ή μέσω κέντρων διανομής.

Η διαδικασία σχεδιασμού των μεταφορών αντιμετωπίζει κάποια σημαντικά προβλήματα όπως:

- Επιλογή κατάλληλου του στόλου μεταφοράς (τόσο όσον αφορά την χωρητικότητα του στόλου όσο και το να επιλέγονται οχήματα διαφορετικού τύπου.
- Δρομολόγηση οχημάτων με τις βέλτιστες διαδρομές βάση της εκάστοτε δομής του δικτύου, των αποστάσεων και της χωρητικότητας των διαδρομών.
- Σχεδιασμός του δικτύου διανομής βελτιώνοντας τα υπάρχοντα δρομολόγια και τον χρονοπρογραμματισμού αυτών αλλά και επιλογή των κατάλληλων ενδιάμεσων αποθηκών.
- Την σωστή επιλογή του ανθρώπινου δυναμικού που θα πραγματοποιεί τις διανομές (ορθός καθορισμός απαιτήσεων προσωπικού).

Ένα σύστημα μεταφοράς αναπαρίσταται με δίκτυα κόμβων και τόξων, όπου οι κόμβοι, τυπικά, αντιπροσωπεύουν πόλεις, αεροδρόμια, στάσεις και αποθήκες και τα τόξα αντιπροσωπεύουν τους συνδέσμους ή τις διαδρομές ανάμεσα στους κόμβους. Τα τόξα, όπως και οι κόμβοι, συνήθως περιλαμβάνουν περιορισμούς χωρητικότητας.

Για την μεταφορά φορτίων μεταξύ δύο σημείων επιλέγεται ο κατάλληλος από τους διαθέσιμους διαφορετικούς τρόπους. Υπάρχουν πέντε βασικοί τρόποι μεταφοράς:

•**Οι σιδηροδρομικοί μεταφορείς**. Μεγάλες αποστάσεις, μεγάλες ποσότητες, μικρό κόστος. Δυνατότητα μεταφοράς υλικών οποιασδήποτε μορφής. Για αυτόν τον λόγο υφίσταται και παροχή των κατάλληλων εγκαταστάσεων, και κατάλληλου εξοπλισμού για τον χειρισμό των υλικών.

•**Οι οδικοί μεταφορείς**. Περιέχουν πολλές παραλλαγές αυτών καλύπτοντας όλες τις μεταφορικές ανάγκες. Ακόμα, δεν απορρίπτεται το ενδεχόμενο για μεταφορά από πόρτα σε πόρτα, χωρίς την απαίτηση κάποιας μετατροπής. Τέλος, τεράστια ευελιξία στην επιλογή δρομολογίων αλλά και αλλαγής κατευθύνσεων μεταφοράς φορτίων.

•**Οι θαλασσιοί μεταφορείς**. Οι θαλάσσιοι (εγχώριοι και υπερπόντιοι) μπορούν να μεταφέρουν με χαμηλό κόστος ανά μίλι πολύ μεγάλα και παντός είδους φορτία. Βασικό μειονέκτημα αποτελεί το πόσο χρονοβόρα είναι η διαδικασία. Ανάλογα το είδος του μεταφερόμενου φορτίου γίνεται και η επιλογή του κατάλληλου τύπου θαλάσσιου μεταφορέα. Επιλέγονται για να μεταφέρουν επικίνδυνα και εξειδικευμένα φορτία κυρίως για λόγους ασφαλείας αλλά και γιατί κάποιες χώρες απαγορεύουν την μεταφορά ορισμένων φορτίων μέσα από τα σύνορα τους.

•**Οι αεροπορικοί μεταφορείς**. Μεταφέρονται κυρίως επιβάτες και ελάχιστα φορτία. Με αυτόν τον τρόπο μεταφοράς συνήθως επιλέγονται φορτία υψηλής αξίας ή υλικά με μικρή διάρκεια ζωής ή επείγεται η μεταφορά τους.

•**Οι αγωγοί μεταφορών**. Μεταφέρουν υγρά φορτία και αέρια. Το ήδη εγκατεστημένο δίκτυο αποτελεί το βασικό μειονέκτημα, σε συνδυασμό με το γεγονός ότι κινούνται προς μια μόνο

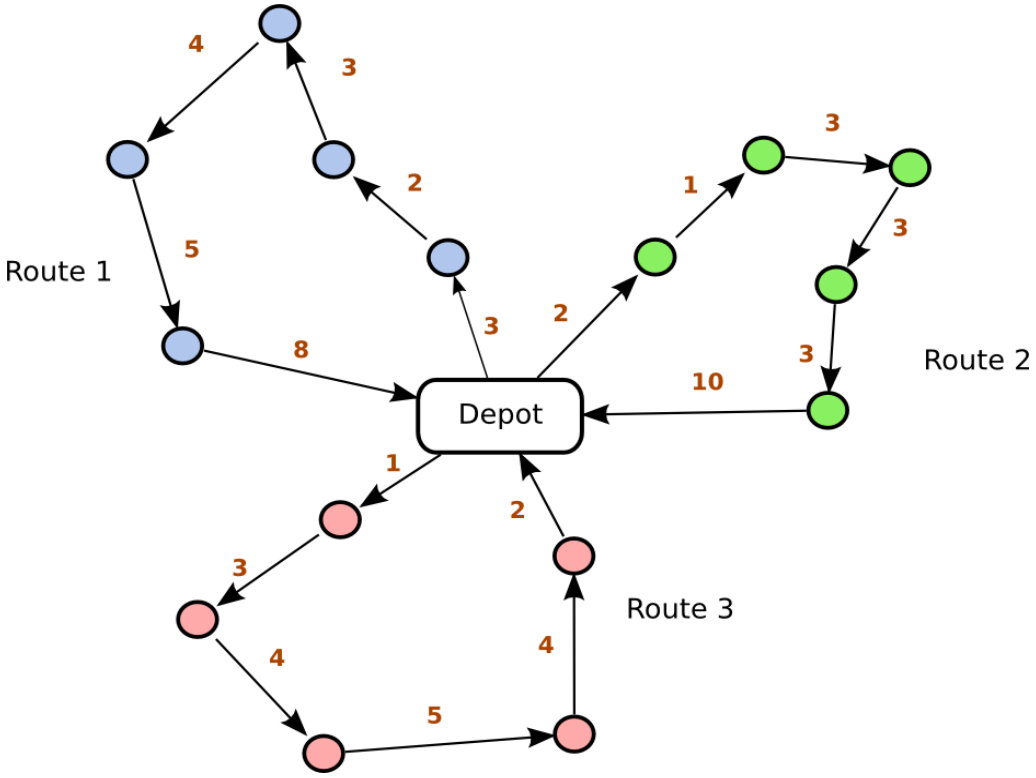
κατεύθυνση. Παρόλα ταύτα, το αρκετά χαμηλό κόστος μεταφοράς και το γεγονός ότι εξαιρείται η διαδικασία δημιουργίας συσκευασίας των προϊόντων, σε συνδυασμό με μη ανεκμετάλλευτο ή υποαπασχολούμενο εξοπλισμό, αποτελούν επιλογή μονόδρομο για ορισμένους τύπους προϊόντων.

2.2 Προβλήματα Δρομολόγησης Οχημάτων

2.2.1 Εισαγωγή στα προβλήματα δρομολόγησης οχημάτων

Τα Προβλήματα Δρομολόγησης Οχημάτων (Vehicle Routing Problems) εμπεριέχονται στην ιεραρχική διαχείριση της εφοδιαστικής αλυσίδας. Για να θεωρηθεί η διαδικασία διανομής προϊόντων αποτελεσματική βασικός οδηγός είναι η εξυπηρέτηση του πελάτη. Για ορισμένη χρονική περίοδο, ο πελάτης (ή πελάτες) ικανοποιούνται από κάποιο όχημα (Ή οχήματα) που ξεκίνησε/αν από προκαθορισμένη αποθήκη, με προκαθορισμένο αριθμό οδηγών μέσω προκαθορισμένου οδικού δικτύου. Σαν γενικός κανόνας, η σωστή επίλυση του προβλήματος δρομολόγησης οχημάτων καθορίζει ένα σύνολο διαδρομών, που ξεκινούν και καταλήγουν σε μία αποθήκη, με στόχο την ικανοποίηση του εκάστοτε πελάτη και των ξεχωριστών απαιτήσεων του και ταυτοχρόνως την αποφυγή παραβίασης των περιορισμών και ελαχιστοποίηση του κόστους διανομής.

Στην παρακάτω εικόνα παρουσιάζεται ένα σχετικά απλό πρόβλημα, που εμπεριέχει μια αποθήκη (Depot) και τρεις διαφορετικές διαδρομές σε ένα οδικό δίκτυο, η κάθε μία από τις οποίες εξυπηρετεί τους πελάτες με το αντίστοιχο χρώμα.



Το οδικό δίκτυο που χρησιμοποιείται γενικά στα προβλήματα μεταφορών περιγράφεται μέσω γραφήματος όπου τα τόξα παίρνουν την θέση των δρόμων του δικτύου και οι κόμβοι των αποθηκών και θέσεων των πελατών. Τα τόξα ενδέχεται να είναι προσανατολισμένα ή και όχι αναλόγως εάν ο δρόμος είναι διπλής κατευθύνσεως. Κάθε τόξο γενικά σχετίζεται με ένα κόστος, που αφορά το μήκος του, και κάποιο χρόνο ταξιδιού.

2.2.2 Τα βασικά χαρακτηριστικά των διαδρομών

Οι διαδρομές είθισται να πληρούν ορισμένα χαρακτηριστικά τα οποία είναι:

- Αφετηρία κάθε διαδρομής είναι η αποθήκη.
- Οι απαιτήσεις των πελατών θα πρέπει να ικανοποιούνται πλήρως από το σύνολο των διαδρομών .
- Οι περιορισμοί κάθε διαδρομής οφείλουν να τηρούνται.

- Η εύρεση διαδρομών που καλύπτει τις παραπάνω προϋποθέσεις με το χαμηλότερο δυνατό κόστος είναι ο βασικός στόχος του προβλήματος.

2.2.3 Τα δεδομένα του προβλήματος

Απαραίτητες για την λύση του προβλήματος είναι ορισμένες πληροφορίες αναφορικά με:

- **Τους πελάτες**:
- Το σημείο του γραφήματος διανομής (τοποθεσία) του πελάτη.
- Την ζήτηση του κάθε πελάτξη για την παράδοση ή συλλογή αγαθών.
- Τα χρονικά διαστήματα (time windows) μέσα στην ημέρα όπου ο πελάτης είναι διαθέσιμος για την ικανοποίηση του αιτήματος του.
- Ο χρόνος παράδοσης ή συλλογής των προϊόντων από τον πελάτη (unloading or loading times) που πιθανόν εξαρτάται από το εκάστοτε είδος του οχήματος.
- Η επιλογή του είδους οχήματος από τον στόλο των οχημάτων για την ικανοποίηση του πελάτη.

- **Τις αποθήκες:**
- Η φυσική τοποθεσία (γεωγραφικά) της αποθήκης.
- Η χωρητικότητα της αποθήκης.
- Τα διαφορετικών ειδών οχήματα που μπορεί να φιλοξενήσει στον χώρο της.
- **Τα οχήματα:**
- Η χωρητικότητα του κάθε οχήματος. Μέγιστο βάρος, όγκος ή αριθμός πελατών που χωράει το όχημα.

- Από ποια αποθήκη προέρχονται.
- Η πιθανή υποδιαίρεση των οχημάτων σε ομάδες αναλόγως τη χωρητικότητας και του είδους προϊόντων που ενδέχεται να μεταφέρει.
- Το πλήθος των διαθέσιμων οχημάτων.
- Το κόστος λειτουργίας του κάθε οχήματος

2.2.4 Οι περιορισμοί του προβλήματος

Ένας αριθμός από περιορισμούς πρέπει να ικανοποιείται. Αυτός ο αριθμός εξαρτάται από τα μεταφερόμενα προϊόντα, το επίπεδο της εξυπηρέτησης αλλά και χαρακτηριστικά των οχημάτων και εκάστοτε πελατών, από την ποιότητα του επιπέδου εξυπηρέτησης, και διάφορα χαρακτηριστικά των οχημάτων και των πελατών. Γενικά, βασικοί περιορισμοί που χρησιμοποιούνται είναι:

- Περιορισμός χωρητικότητας οχημάτων.
- Περιορισμός όσων αφορά το είδος εξυπηρέτησης.
- Περιορισμός αναφορικά με πόσα οχήματα είναι διαθέσιμα.
- Χρονικοί περιορισμοί.
- Με ποια σειρά ικανοποιούνται οι πελάτες.

2.2.5 Το κόστος των διαδρομών

Το κόστος της κάθε διαδρομής υπολογίζεται συνήθως με το άθροισμα δύο ξεχωριστών κοστών. Το ένα είναι το κόστος του ταξιδιού από τον κόμβο i στον κόμβο j, δηλαδή ο χρόνος που απαιτείται για την μεταφοράς ενός οχήματος από τον έναν πελάτη στον άλλον. Το άλλο είναι ο χρόνος που απαιτείται για την εξυπηρέτηση του πελάτη. Με άλλα λόγια, είναι ο χρόνος που χρειάζεται ένας οδηγός για να φτάσει σε έναν πελάτη και στην συνέχεια να πάει στον επόμενο πελάτη. Σε πολλές περιπτώσεις, υπολογίζεται επίσης και το κόστος λειτουργίας του οχήματος

2.2.6 Στόχοι για την επίλυση του προβλήματος

Η επίλυση προβλημάτων δρομολόγησης οχημάτων στοχεύει:

- Στην ελαχιστοποίηση του συνολικού κόστους μεταφοράς των προϊόντων .
- Στην ελαχιστοποίηση του αριθμού των οχημάτων που χρειάζονται για να ικανοποιηθούν όλοι οι πελάτες.
- Στην ικανοποίηση όλων των πελατών.

2.3 Εφαρμογές του Προβλήματος Δρομολόγησης Οχημάτων

Στην ενότητα αυτή θα παρουσιαστούν περιληπτικά ορισμένα από τα πολλά παραδείγματα εφαρμογών του προβλήματος δρομολόγησης οχημάτων στην καθημερινή ζωή.

2.3.1 Πρόβλημα δρομολόγησης Πλοίων

Σκοπός του προβλήματος αυτού είναι η εύρεση του βέλτιστου στόλου πλοίων που χρειάζονται σε μια ακτοπλοϊκή γραμμή. Επιλογή τύπου και αριθμού πλοίων και δημιουργία εβδομαδιαίων διαδρομών. Τα πλοία, μπορεί είτε να ανήκουν στην εταιρεία είτε να μισθωθούν για κάποιο χρονικό διάστημα ώστε να εξυπηρετούν τις εκάστοτε ανάγκες της εταιρείας.

2.3.2 Πρόβλημα δρομολόγησης οχημάτων στην βιομηχανία αποκομιδής σκουπιδιών

Εδώ μέσω της σωστής δρομολόγησης απορριμματοφόρων γίνεται η περισυλλογή σκουπιδιών από τις κατοικίες. Υπάρχουν τριών τύπων προβλημάτων γι’ αυτό το πρόβλημα:

- Η συλλογή των αποβλήτων από τις βιομηχανίες, όπου ο ’’πελάτης‘’ βρίσκεται πάνω στους κόμβους, καθώς οι βιομηχανίες συνήθως μαζεύουν τα απόβλητα σε κάποια αποθήκη απ’ όπου περνάει και το απορριμματοφόρο.

- Η συλλογή σκουπιδιών από τα σπίτια, το οποίο θεωρείται πρόβλημα δρομολόγησης τόξων αφού τα σπίτια βρίσκονται πάνω στο οδικό δίκτυο, άρα το όχημα περνάει από πολλά σημεία στον δρόμο.

- Χρησιμοποίηση μικρών οχημάτων, λόγω της ευελιξίας τους σε στενούς δρόμους, για την περισυλλογή και μεταφορά των απορριμμάτων σε κάποια κεντρικά σημεία όπου περιμένει κάποιο μεγάλο απορριμματοφόρο και τα μαζεύει και τα μεταφέρει στα σημεία υγειονομικής ταφής τους ή στα σημεία καταστροφής τους.

2.3.3 Πρόβλημα δρομολόγησης οχημάτων στην βιομηχανία ποτών, τροφών και γάλακτος

Σε αυτές τις βιομηχανίες λόγω της μεγάλης ποσότητας πωλήσεων τα έξοδα που συσχετίζονται με δραστηριότητες διανομής προϊόντων είναι τεράστια. Κοινός στόχος και των τριών βιομηχανιών είναι η διανομή των προϊόντων τους:

- Την σωστή χρονική στιγμή
- Στη σωστή ποσότητα
- Ελαχιστοποιώντας το κόστος διανομής και μεταφοράς αλλά και
- Μεγιστοποιώντας την ικανοποίηση των πελατών.

2.3.4 Δρομολόγηση και διανομή στην βιομηχανία παραγωγής εφημερίδων

Η βιομηχανία διανομής εφημερίδων, όπως και περιοδικών, είχε πάντοτε ένα από τα μεγαλύτερα προβλήματα διανομής. Η παραγωγή και η διανομή μιας εφημερίδας είναι ένας κύκλος που διαρκεί μια ολόκληρη

μέρα για 365 μέρες τον χρόνο. Υπάρχει ένας μεγάλος αριθμός εκδοτικών οίκων που πρέπει να τυπώσουν τις εφημερίδες μέχρι τη στιγμή που θα φύγει κάποιο φορτηγό για να τις μεταφέρει σε κάποιο κεντρικό σημείο όπου θα συλλεχθούν όλες οι ημερήσιες εφημερίδες, απ' όλους τους εκδοτικούς οίκους για να διανεμηθούν σε όλη τη χώρα όλες μαζί, γίνεται προσπάθεια δηλαδή καμία εφημερίδα να μην μείνει έξω από τη διανομή για να μην υπάρξουν παράπονα από τους αναγνώστες και έχοντας ένα χρονικό όριο που θα πρέπει να γίνει η διανομή.

Κεφάλαιο 3

Πρόβλημα Δρομολόγησης Σχολικών Λεωφορείων

3.1 Εισαγωγή στο πρόβλημα δρομολόγησης σχολικών λεωφορείων

Το πρόβλημα δρομολόγησης σχολικών λεωφορείων[3], εν συντομία SBRP (School Bus Routing Problem), έχει ως στόχο να σχεδιάσει ένα αποτελεσματικό πρόγραμμα για ένα στόλο σχολικών λεωφορείων στο οποίο κάθε λεωφορείο παραλαμβάνει μαθητές από διάφορες στάσεις και τους παραδίδει στο αντίστοιχο σχολείο ικανοποιώντας ταυτόχρονα κάποιους περιορισμούς όπως η μέγιστη χωρητικότητα του κάθε λεωφορείου, ο μέγιστος χρόνος που μπορεί να βρίσκεται ο κάθε μαθητής εντός λεωφορείου, η μέγιστη απόσταση που μπορεί να περπατήσει ο κάθε μαθητής μέχρι την στάση και το χρονικό παράθυρο που θέτει το κάθε σχολείο. Αυτή η κατηγορία προβλημάτων αποτελείται από διαφορετικά υπό-προβλήματα που περιλαμβάνουν:

- Την προετοιμασία δεδομένων
- Την επιλογή στάσεων λεωφορείων
- Τον σχεδιασμό διαδρομών των λεωφορείων
- Την προσαρμογή αναλόγως με την χρονική στιγμή που ξεκινούν τα μαθήματα στο κάθε σχολείο (“χτυπάει το κουδούνι”) και
- Τον προγραμματισμό των λεωφορείων.

3.1.1 Προετοιμασία δεδομένων

Στο στάδιο της προετοιμασίας δεδομένων, προσδιορίζεται το οδικό δίκτυο που αποτελείται από τεσσάρων ειδών δεδομένα. Τους μαθητές, τα σχολείο, τα λεωφορεία και τον πίνακα προέλευσης-προορισμού. Στα δεδομένα των μαθητών περιλαμβάνονται η διεύθυνση και το σχολείο

του κάθε μαθητή αλλά και αν ο μαθητής είναι άτομο με ειδικές ανάγκες ή όχι. Τα δεδομένα των σχολείων περιέχουν πληροφορίες περί της διεύθυνσης του κάθε σχολείου, την χρονική στιγμή έναρξης και λήξης μαθημάτων για την άφιξη των λεωφορείων και ο μέγιστος χρόνος τον οποίο μπορεί να περάσει ένας μαθητής από την στιγμή που θα εισέλθει στο λεωφορείο μέχρι να φτάσει στον προορισμό του. Όσων αφορά τα δεδομένα των λεωφορείων, εμπεριέχονται η αρχική τοποθεσία αλλά και το είδος του κάθε ενός από αυτά. Τα λεωφορεία μπορεί να διαφέρουν μεταξύ τους στην χωρητικότητα για μαθητές με ειδικές ανάγκες και απλών. Τέλος, στον πίνακα προέλευσης-προορισμού αποθηκεύονται οι συντομότερες διαδρομές (είτε βάσει χιλιομετρικής απόστασης είτε χρόνου) ανάμεσα σε ζευγάρια κόμβων (σχολείο, τοποθεσία μαθητή, τοποθεσία λεωφορείου).

3.1.2 Επιλογή στάσεων λεωφορείων

Για δεδομένο δίκτυο στο στάδιο της επιλογής των στάσεων, επιλέγονται τα ακριβή σημεία των στάσεων στα οποία αντίστοιχα αναθέτονται μαθητές. Αναλόγως αν το πρόβλημα προς επίλυση απευθύνεται σε αστικές περιοχές ή προάστεια τότε το σύνηθες φαινόμενο είναι οι μαθητές να περπατήσουν σε κάποια κοντινή στάση ή να εξυπηρετηθούν απευθείας από την οικεία τους αντίστοιχα. Πολλές φορές ωστόσο, το στάδιο επιλογής των στάσεων παραλείπεται από την επιστημονική λογοτεχνία καθώς πολλές έρευνες θεωρούν τις στάσεις ως δεδομένες. Σε ελάχιστες έρευνες εφαρμόζεται αυτό το στάδιο όπου χρησιμοποιούνται ευρετικοί αλγόριθμοι. Οι ευρετικές προσεγγιστικές λύσεις κατηγοριοποιούνται σε δύο στρατηγικές: Τοποθεσία-Κατανομή-Δρομολόγηση (LAR: Location-Allocation-Routing) και Κατανομή-Δρομολόγηση-Τοποθεσία (ARL: Allocation-Routing-Location).

- Στην LAR στρατηγική προσδιορίζονται πρώτα οι στάσεις των λεωφορείων, έπειτα κατανέμονται οι μαθητές σε αυτές καταλλήλως και τέλος σχεδιάζεται η δρομολόγηση των οχημάτων που περνάει από τις στάσεις αυτές. Παρόλα ταύτα, εφόσον οι

στάσεις και η κατανομή των μαθητών σε αυτές αποφασίζονται χωρίς να λαμβάνονται υπόψιν η επίδραση τους στον σχεδιασμό διαδρομών, αυτή η μέθοδος τείνει να παράγει υπερβολικό αριθμό διαδρομών.

▪Στην ARL στρατηγική ο κάθε μαθητής κατανέμεται σε κάποιο σύμπλεγμα ώστε να ικανοποιείται ο περιορισμός της χωρητικότητας του κάθε λεωφορείου. Ακολούθως επιλέγονται οι στάσεις των λεωφορείων και σχεδιάζεται η διαδρομή για κάθε σύμπλεγμα. Τέλος, οι μαθητές του κάθε συμπλέγματος κατανέμονται σε συγκεκριμένες στάσεις ώστε να ικανοποιούνται όλοι οι περιορισμοί του εκάστοτε προβλήματος, όπως η μέγιστη απόσταση που μπορεί να περπατήσει ένας μαθητής μέχρι να φτάσει στην στάση του, το πόσοι μαθητές μπορούν να περιμένουν σε μία στάση, ή η ελάχιστη χιλιομετρική απόσταση μεταξύ των στάσεων.

Παρακάτω ακολουθεί σχηματική αναπαράσταση των δύο στρατηγικών.

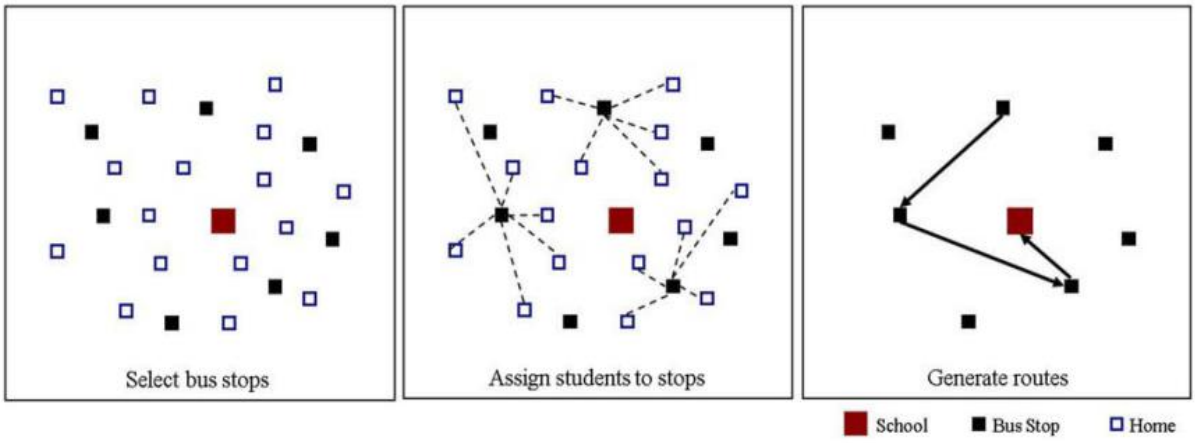


Fig. 1. Concept of LAR strategy.

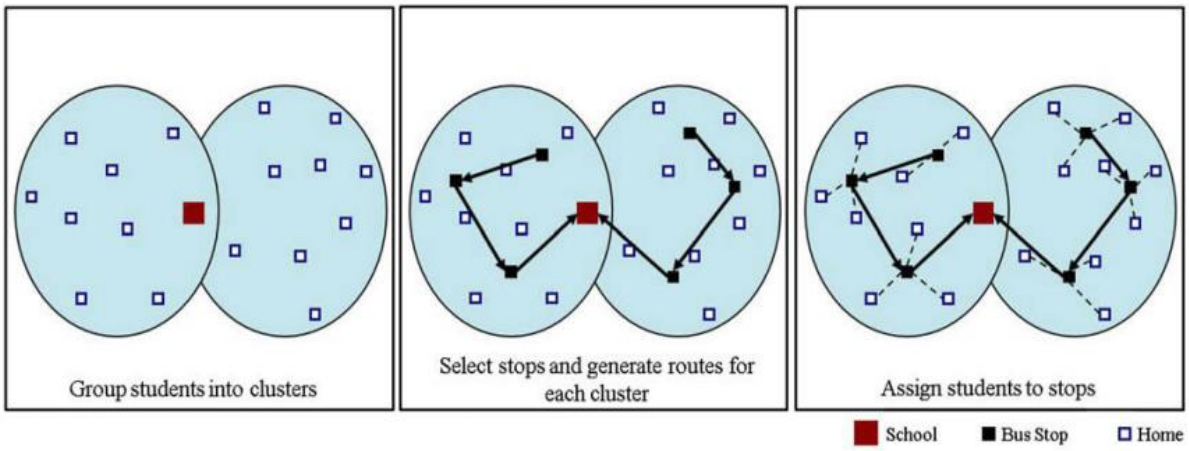


Fig. 2. Concept of ARL strategy.

3.1.3 Σχεδιασμός διαδρομών των λεωφορείων

Στο στάδιο σχεδιασμού διαδρομών καθορίζονται χρησιμοποιώντας αλγορίθμους είτε με “route-first cluster-second” είτε με “cluster-first route-second” προσέγγιση. Πρώτα, σχεδιάζεται μια αρκετά μεγάλη διαδρομή μέσω του αλγορίθμου του πλανόδιου πωλητή που δέχεται όλες τις στάσεις και την χωρίζει σε μικρότερες διαδρομές ικανοποιώντας όλους του περιορισμούς ενώ στην δεύτερη προσέγγιση χωρίζονται οι μαθητές σε συμπλέγματα έτσι ώστε κάθε σύμπλεγμα να θεωρηθεί ως μια διαδρομή ικανοποιώντας τους περιορισμούς. Αφού σχεδιαστούν οι

διαδρομές μπορούν να εφαρμοστούν ευρετικοί και μεθευρετικοί αλγόριθμοι για βελτιστοποίηση των λύσεων οι οποίοι θα αναλυθούν στην συνέχεια.

3.1.4 Προσαρμογή χρόνου έναρξης μαθημάτων των σχολείων

Όσων αφορά το στάδιο της προσαρμογής αναλόγως με την χρονική στιγμή που ξεκινούν τα μαθήματα στο κάθε σχολείο, στις περισσότερες έρευνες οι χρόνοι έναρξης και λήξης μαθημάτων θεωρούνται ως περιορισμοί. Ωστόσο υπάρχουν άλλες που θεωρούν τους χρόνους αυτούς ως μεταβλητές απόφασης και επιδιώκουν να βρουν τους βέλτιστους χρόνους ώστε να μεγιστοποιηθεί ο αριθμός των διαδοχικών διαδρομών που μπορεί να κάνει το κάθε λεωφορείο και να ελαχιστοποιηθεί ο αριθμός των λεωφορείων που χρησιμοποιούνται.

3.1.5 Προγραμματισμός των λεωφορείων

Στο τελικό στάδιο, αυτό του προγραμματισμού των λεωφορείων, καθορίζονται τον ακριβή χρόνο άφιξης και τερματισμού κάθε διαδρομής και σχηματίζεται μια αλυσίδα από διαδρομές που μπορεί να εκτελέσει το ίδιο λεωφορείο σε έναν κύκλο..

3.2 Ταξινόμηση προβλημάτων δρομολόγησης σχολικών λεωφορείων

Τα υποπροβλήματα που παρουσιάστηκαν στο κεφάλαιο 3.1. θεωρούνται[3] ξεχωριστά και διαδοχικά, ωστόσο στην πραγματικότητα δεν είναι ανεξάρτητα αλλά αλληλοσχετίζονται. Λόγω του μεγέθους και της πολυπλοκότητας του εκάστοτε προβλήματος αντιμετωπίζονται ξεχωριστά. Πιο συγκεκριμένα, ταξινομούνται ανάλογα των χαρακτηριστικών του εκάστοτε προβλήματος και βάσει την μεθόδων επίλυσης που χρησιμοποιήθηκαν.

3.2.1. Ταξινόμηση προβλημάτων βάσει των χαρακτηριστικών του προβλήματος

Παρόλο που υπάρχουν αρκετοί τρόποι ταξινόμησης, οι παρακάτω εστιάζουν στις πρακτικές πτυχές του προβλήματος.

3.2.1.1. Αριθμός σχολείων: Μοναδικό σχολείο ή πολλαπλά σχολεία

Τα λεωφορεία μπορεί να περιέχουν μαθητές από ένα και μοναδικό σχολείο είτε από δύο ή και περισσότερα.

Το πρόβλημα που αφορά μοναδικό σχολείο είναι παρόμοιο με το παραδοσιακό πρόβλημα δρομολόγησης οχημάτων (VRP) κατά το οποίο ένα όχημα πραγματοποιεί μια διαδρομή με αφετηρία την αποθήκη, και ικανοποιώντας την ζήτηση όλων των πελατών της συγκεκριμένης διαδρομής, επιστρέφει και πάλι στην αρχική του θέση (αποθήκη). Ωστόσο, τις περισσότερες φορές, στο πρόβλημα δρομολόγησης σχολικών λεωφορείων (SBRP), αφετηρία αποτελεί κάποιο άλλο σημείο και όχι το ίδιο το σχολείο (αποθήκη). Σαν αποτέλεσμα, η δομή του SBRP μοιάζει περισσότερο με αυτήν του ανοιχτού προβλήματος δρομολόγησης οχημάτων (Open VRP). Αυτό που χαρακτηρίζει το Open VRP είναι ότι το όχημα αφού περάσει από όλους τους κόμβους, δεν καταλήγει πάλι στο ίδιο αρχικό σημείο.

Η περίπτωση του προβλήματος πολλαπλών σχολείων μπορεί να επιλυθεί με δύο διαφορετικές προσεγγίσεις. Με προσέγγιση βάσει των σχολείων και με την προσέγγιση βάσει των σπιτιών.

Στην πρώτη περίπτωση, δημιουργούνται οι διαδρομές για κάθε σχολείο, τις οποίες αναλαμβάνουν να ικανοποιήσουν ένας στόλος λεωφορείων βάσει των χρονικών παραθύρων του κάθε σχολείου. Εδώ, δεν επιτρέπεται για μαθητές διαφορετικών σχολείων να βρίσκονται στο ίδιο λεωφορείο την ίδια στιγμή.

Στην περίπτωση προσέγγισης βάσει των σπιτιών τοποθετείται ένα σημείο stop στην διαδρομή κάποια χρονική στιγμή. Όταν ένα τέτοιο σημείο τοποθετηθεί στην διαδρομή, το αντίστοιχο σχολείο θα πρέπει να βρίσκεται μέσα στην διαδρομή αυτή. Εάν το συγκεκριμένο σχολείο δεν βρίσκεται στην διαδρομή αυτή, τότε ο αλγόριθμος τοποθετεί την θέση του στην βέλτιστη θέση της διαδρομής αυτή, η οποία επιλέγεται βάσει του κόστους της τοποθέτησης. Στην συγκεκριμένη περίπτωση, δεν απαγορεύεται η ύπαρξη μαθητών διαφορετικών σχολείων στο ίδιο λεωφορείο την ίδια στιγμή.

3.2.1.2. Αστική ή προαστιακή περιοχή

Η λύση που προσεγγίζει το πρόβλημα δρομολόγησης σχολικών λεωφορείων μπορεί να διαφέρει αναλόγως του αν το σχολείο βρίσκεται σε αστική περιοχή ή κάποιο προάστιο.

Στην περίπτωση αστικής περιοχής, υποθέτουμε πως οι μαθητές διανύουν μια απόσταση με τα πόδια μέχρι την κοντινότερη στάση λεωφορείου. Πολλές μελέτες υποστηρίζουν πως τις περισσότερες φορές, παραβιάζεται ο περιορισμός της χωρητικότητας των λεωφορείων πριν από τον μέγιστο χρόνο ταξιδιού ενός μαθητή, εξ αιτίας του αυξημένοι αριθμού μαθητών στις αστικές περιοχές.

Σε προαστιακές περιοχές, εν αντιθέσει με τις αστικές περιοχές, ο αριθμός των μαθητών είναι μικρός και επομένως το λεωφορείο μπορεί να παραλάβει τους μαθητές από τα σπίτια τους, καθιστώντας την διαδικασία επιλογής στάσεων μη αναγκαία. Τα προάστια διαφέρουν αρκετά από τις αστικές περιοχές καθώς: έχουν λιγότερο πυκνό πληθυσμό, μεγαλύτερες χιλιομετρικές αποστάσεις ανά διαδρομή, ελάχιστους ή και πολλές φορές κανένα μονόδρομο, λιγότερο επιβατικό κοινό σε κάθε στάση, περισσότερες στάσεις συνολικά ανά διαδρομή και λιγότερες εναλλακτικές διαδρομές από στάση σε στάση.

3.2.1.3. Πρωί ή απόγευμα

Τα προβλήματα δρομολόγησης σχολικών λεωφορείων συνήθως αντιμετωπίζουν το πρόβλημα του να μεταφέρουν τους μαθητές το πρωί από τις στάσεις στο αντίστοιχο σχολείο τους τις πρωινές ώρες αλλά και αντίστροφα το πρόβλημα του να μεταφερθούν οι μαθητές από τα σχολεία τους στις αντίστοιχες στάσεις από όπου τους παρέλαβαν τις απογευματινές ώρες. Το πρόβλημα τις πρωινές ώρες θεωρείται σαφώς πιο δύσκολο λόγω των μη ομοιόμορφων χρονικών παραθύρων των σχολείων και της αυξημένης κίνησης στους δρόμους τις ώρες αιχμής. Για αυτό και το πρωινό πρόβλημα είναι πιο δημοφιλές. Πιθανά σενάρια μπορεί να είναι οι διαδρομές του απογεύματος να ταυτίζονται με του πρωινού ή να είναι αντίστροφες για τη μείωση του χρόνου ταξιδιού.

3.2.1.4. Επιβίβαση μαθητών διαφορετικών σχολείων

Η περίπτωση αυτού του προβλήματος πέρα από τις αστικές μπορεί να προκύψει και σε προαστιακές περιοχές. Εδώ, μαθητές διαφορετικών σχολείων είναι εφικτό να βρίσκονται στο ίδιο λεωφορείο και μάλιστα την ταυτόχρονα. Με αυτόν τον τρόπο έχουμε περισσότερη ευλυγισία και οικονομία κόστους

3.2.1.5. Δρομολόγηση μαθητών με ειδικές ανάγκες

Το συγκεκριμένο πρόβλημα διαφέρει κατά πολύ σε σχέση με το την δρομολόγηση μαθητών γενικής κατηγορίας. Καταρχάς οι μαθητές με ειδικές ανάγκες πρέπει να παραλαμβάνονται και να επιστρέφουν απευθείας στα σπίτια τους και όχι σε προκαθορισμένες στάσεις λεωφορείων. Επιπλέον διαφέρει ο χρόνος μέχρι να επιβιβαστεί ο μαθητής στο λεωφορείο. Επίσης, ανάλογα της σοβαρότητας της αναπηρίας του μαθητή μπορεί να χρειάζεται ειδικός εξοπλισμός αλλά

βοήθεια στην διαδικασία επιβίβασης του, επομένως μπορεί να χρειάζεται ειδικό λεωφορείο για να τον εξυπηρετήσει.

3.2.1.6. Ομογενής ή ετερογενής στόλος οχημάτων

Το πρόβλημα που αντιμετωπίζεται με έναν ετερογενή στόλο οχημάτων υποθέτει πως κάθε όχημα έχει κάποια ιδιαίτερα χαρακτηριστικά όπως διαφορετική χωρητικότητα, διαφορετικό μέγιστο χρόνο αναμονής σε κάθε στάση και συνολικό κόστος.

3.2.1.7. Στόχος προβλήματος

Ανάλογα με τον στόχο κάθε προβλήματος υπάρχουν τρεις διαφορετικές μέθοδοι αξιολόγησης δημόσιων λειτουργιών: Αποδοτικότητα, Αποτελεσματικότητα, Δικαιοσύνη.

Η Αποδοτικότητα ορίζεται ως η αναλογία του επιπέδου λειτουργίας με το κόστος των προμηθειών που απαιτούνται για την συγκεκριμένη λειτουργία. Για σταθερό επίπεδο λειτουργίας η αποδοτικότητα μετριέται από το κόστος της. Υπάρχουν δύο συστατικά που επηρεάζουν το κόστος ενός συστήματος δρομολόγησης σχολικών λεωφορείων. Το ένα κόστος είναι το κόστος κεφαλαίου που απαιτείται για την λειτουργία ενός σχολικού λεωφορείου καθ' όλη τη διάρκεια ενός σχολικού έτους και το άλλο είναι το σταδιακό κόστος μιας μονάδας απόστασης μου πραγματοποιήθηκε. Επομένως, μια λύση που απαιτεί μικρότερο αριθμό λεωφορείων και μικρότερο συνολικό χρόνο ταξιδιού είναι προτιμητέα αν στόχος είναι η αποδοτικότητα.

Η Αποτελεσματικότητα μιας λειτουργίας μετριέται βάση της ικανοποίησης της ζήτησης. Ένα ορθώς λειτουργικό σύστημα μεταφοράς μαθητών με σχολικά λεωφορεία πρέπει να είναι διαθέσιμο σε όλους τους μαθητές. Το επίπεδο της αποτελεσματικότητας λοιπόν μπορεί να μετρηθεί από τον συνολικό χρόνο που βρίσκονται οι μαθητές εντός των

λεωφορείων και της μέγιστης απόστασης που αυτοί οι μαθητές θα χρειαστεί να διανύσουν με τα πόδια μέχρι την στάση τους.

Η Δικαιοσύνη εκτιμάει την ισότητα και την αμεροληψία της παροχής μιας υπηρεσίας. Μια λύση με ικανοποιητικό μέτρο αποδοτικότητας μπορεί να είναι φθηνή και καλύτερη αναφορικά με το συνολικό κόστος και χρόνο αλλά μπορεί να θεωρηθεί απαράδεκτη στο κριτήριο της δικαιοσύνης επειδή δεν υπάρχει αμεροληψία στην ικανοποίηση όλων των μαθητών. Το κριτήριο της Δικαιοσύνης συχνά παραμελείται στις υπηρεσίες σχολικών λεωφορείων ή ακόμα κα δημόσιων λειτουργιών. Για την βελτίωση της λοιπόν, πρέπει να υπάρχει ισορροπία ανάμεσα στην χωρητικότητα του κάθε λεωφορείου αλλά και τον συνολικό χρόνου που αυτό ταξιδεύει.

Οι περισσότερες μελέτες προβλημάτων δρομολόγησης σχολικών λεωφορείων στοχεύουν στην ελαχιστοποίηση των αριθμών των λεωφορείων και του συνολικού χρόνου που πραγματοποιείται κάθε διαδρομή. Ελάχιστες είναι οι έρευνες που λαμβάνουν υπόψιν το κριτήριο της Δικαιοσύνης όπως την εξισορρόπηση χωρητικότητας των λεωφορείων.

3.2.1.8. Περιορισμοί προβλήματος

Υπάρχουν πολλών ειδών περιορισμοί που εφαρμόζονται στα προβλήματα δρομολόγησης σχολικών λεωφορείων όπως:

- Χωρητικότητα λεωφορείου: Τοποθέτηση ανώτατου επιτρεπόμενου ορίου αριθμού μαθητών που μπορούν να επιβιβαστούν.

- Χρόνος ταξιδιού: Ο μέγιστος χρόνος που μπορεί να περάσει ένας μαθητής μέσα στο λεωφορείο.

- Απόσταση την οποία μπορούν να περπατήσουν οι μαθητές: Η μέγιστη επιτρεπόμενη χιλιομετρική απόσταση που μπορεί να διανύσει ο κάθε μαθητής από και προς το σπίτι του από την στάση λεωφορείου.

- Χρονικό παράθυρο σχολείου: Το χρονικό παράθυρο που επιτρέπει το κάθε σχολείο στους μαθητές να παρευρίσκονται εντός των εγκαταστάσεων του.

- Χωρητικότητα στάσεων: Μέγιστος επιτρεπόμενος αριθμός μαθητών που μπορούν να περιμένουν στην ίδια στάση.

- Χρόνος παραλαβής: Όριο στον νωρίτερο χρόνο που μπορούν να παραλάβουν τους μαθητές.

- Αριθμός μαθητών ανά διαδρομή: Ο ελάχιστος αριθμός μαθητών που χρειάζονται για την δημιουργία νέας διαδρομής του σχολικού λεωφορείου.

Κεφάλαιο 4

Αλγόριθμοι Βελτιστοποίησης προβλημάτων της Εφοδιαστικής Αλυσίδας

4.1 Ευρετικοί Αλγόριθμοι

Η επίλυση ενός προβλήματος συνδυαστικής βελτιστοποίησης γίνεται σταθερά και πιο δύσκολη όσο μεγαλώνει και συχνά είναι σχεδόν ανέφικτο να υπάρξει η καλύτερη λύση σε σύντομο χρονικό διάστημα. Για επίλυση τέτοιων προβλημάτων, εφαρμόζονται τεχνικές για την εύρεση τοπικού ελαχίστου αντί για ολικού. Και δεχόμαστε αυτή τη λύση αν ορισμένα κριτήρια, όπως η ποιότητα της λύσης (πόσο απέχει από την βέλτιστη), ο χρόνος και πόσο εύκολα αποκτήθηκε αυτή η λύση, ικανοποιούνται.

Οι Ευρετικοί αλγόριθμοί κατηγοριοποιούνται ως εξής:

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

4.1.1 Αλγόριθμοι απληστίας

Η λογική του άπληστου αλγορίθμου είναι πολύ απλή. Ξεκινώντας από μια μερική ανέφικτη λύση, σε κάθε βήμα προσδιορίζεται μια ή περισσότερες μεταβλητές μέχρι τον εντοπισμό μιας εφικτής λύσης, αυτή που ζητούσαμε.

Γενικά, οι αλγόριθμοι απληστίας χρησιμοποιούν την αρχή ανάπτυξης μυωπικών κατασκευαστικών αλγορίθμων. Αυτό σημαίνει ότι αλγόριθμοι κατασκευάζουν μια εφικτή λύση για δεδομένο πρόβλημα σε χρόνο πολυωνυμικό, ως προς το μέγεθος των δεδομένων.

Εφικτή λύση είναι εκείνη που ικανοποιεί όλους τους περιορισμούς του έχουν τεθεί στο πρόβλημα, επομένως, είναι πολύ εύκολο να βρεθεί μια εφικτή λύση, καθώς η βελτιστοποίησή της δεν μας απασχολεί. Το μεγαλύτερο πλεονέκτημα η παραγωγή λύσης σε μικρό χρονικό διάστημα, παρόλα ταύτα, η λύση αυτή απέχει από το ολικό ελάχιστο που επιθυμούμε.

Οι αλγόριθμοί αυτοί χρησιμοποιούν διάφορες προσεγγίσεις οι οποίες έχουν ως στόχο στην αποτελεσματικότητα των μεθόδων. Ορισμένα χαρακτηριστικά των προσεγγίσεων αυτών είναι[1]:

- **Ομαδοποίηση πρώτα – δρομολόγηση έπειτα (Cluster first – route second)**

  Πρώτα ομαδοποιούνται οι κόμβοι και εν συνεχεία δημιουργούνται οι διαδρομές

- **Δρομολόγηση πρώτα – ομαδοποίηση έπειτα (Route first – cluster second)**

  Κατασκευάζεται πρώτα μια αρκετά μεγάλη διαδρομή που ικανοποιεί την ζήτηση και μετά διαχωρίζεται σε μικρότερες διαδρομές

- **Εξοικονομήσεις / καταχώρηση (Savings / Insertion)**

  Εδώ στοχεύουμε στην ελαχιστοποίηση του κόστους. Συγκρίνονται δύο διαδρομές που μπορεί να είναι και μη εφικτές, καταχωρείται αυτή με το μικρότερο κόστος. Σταματάει με μία εφικτή λύση.

- **Βελτίωση ή ανταλλαγή (Improvement or exchange)**

  Οι διαδικασίες βελτίωσης ή ανταλλαγής παραμένουν και καταλήγουν καλύτερη δυνατή λύση. Σε κάθε βήμα μια εφικτή λύση βελτιώνεται, για να καταλήξει σε μια άλλη εφικτή λύση το συνολικό κόστος φανερά μειωμένο.. Η διαδικασία αυτή επαναλαμβάνεται έως ότου να μην υπάρχουν μειώσεις τα επιπρόσθετα κόστη.

- **Προσέγγιση μαθηματικού προγραμματισμού (Mathematical programming approach)**

  Μέσω μαθηματικών μοντέλων και προγραμματισμού, περιλαμβάνοντας αλγορίθμους οι οποίοι βασίζονται στην μορφοποίηση του κάθε προβλήματος δρομολόγησή οχημάτων.

- **Αλληλοεπιδρών βελτιστοποίηση (Interactive optimization)**

  Εδώ στηριζόμαστε στην απόφαση έμπειρων ανθρώπων καθώς μέσω της διορατικότητας τους θα βοηθήσει στην επίλυση του προβλήματος.

- **Ακριβής διαδικασία (Exact procedure)**

  Οι ακριβείς διαδικασίες για την επίλυση προβλημάτων δρομολόγησης οχημάτων περιλαμβάνουν εξειδικευμένους αλγορίθμους (branch and bound και cutting planes).

Οι κυριότεροι αλγόριθμοι απληστίας είναι[1] :

- Αλγόριθμος Πλησιέστερου Γείτονα (Nearest Neighborhood Algorithm).
- Αλγόριθμος της διαδικασίας εισαγωγής κόμβων (Nearest Insertion Algorithm).
- Αλγόριθμος εγγύτερης συγχώνευσης
- Αλγόριθμος εγγύτερης πρόσθεσης
- Ο αλγόριθμος εξοικονομήσεων των Clarke and Write (The savings Algorithm)

4.2.2 Προσεγγιστικοί Αλγόριθμοι

Οι προσεγγιστικοί αλγόριθμοι[2] λειτουργούν με τον ίδιο τρόπο με τους αλγορίθμους απληστίας απλώς χρησιμοποιούν παραπάνω πληροφορίες. Επί της ουσίας, μέσω προσεγγιστικών αλγορίθμων υπολογίζεται,

χρονικώς πολυωνυμικά, λύση πολύ κοντά στην βέλτιστη. Χρησιμοποιούνται όταν αν δεν μπορεί να εντοπιστεί λύση στο πρόβλημα με κάποια άλλη μέθοδο ή διαφορετικά αν η λύση των υπόλοιπων μεθόδων είναι χρονοβόρα και μη αποδοτική. Ακόμα, εδώ ενσωματώνεται και ο όρος της προσέγγισης ο οποίος είναι ο λόγος της απόκλισης της λύσης του προσεγγιστικού αλγορίθμου σε σχέση με την βέλτιστη δυνατή λύση που ψάχνουμε.

4.2.3 Αλγόριθμοι τοπικής αναζήτησης

Η τοπική αναζήτηση στηρίζεται εφαρμόζεται από την αρχαιότητα. Δοκιμή και σφάλμα. Η ιδέα είναι πολύ απλή και αποδοτική γι' αυτό και οι αλγόριθμοι αυτοί χρησιμοποιούνται διαχρονικά. Οι αλγόριθμοι τοπικής αναζήτησης είναι αλγόριθμοι βελτιστοποίησης μιας ήδη υπάρχουσας λύσης προϋποθέτοντας να χρησιμοποιηθούν αλγόριθμοι απληστίας ή προσεγγιστικοί αλγόριθμοι για να δημιουργηθεί αρχική λύση.

Ξεκινώντας, γίνεται επιλογή μιας γειτονιάς λύσεων από το ευρύτερο σύνολο που είναι διαθέσιμο και ύστερα χρησιμοποιείται επαναλαμβανόμενη διαδικασία, αναλόγως του αλγορίθμου, και εν τέλη παράγεται μια εφικτή λύση. Σε κάθε επανάληψη βρίσκεται μια διαφορετική λύση από την αρχική και στην συνέχεια αν συγκριθούν οι δύο λύσεις καταλήγουμε σε αυτή με το ελάχιστο κόστος. Όταν περάσουμε το σημείο τερματισμού παύουν και οι επαναλήψεις.. Σημείο τερματισμού μπορεί να θεωρηθεί ένας αριθμός επαναλήψεων ή ένας αν σε κάποιο προκαθορισμένο αριθμό επαναλήψεων η λύση έχει πάψει να βελτιώνεται άλλο.

Κα τα την διαδικασία της σχεδίασης ενός αλγορίθμου τοπικής αναζήτης παρουσιάζονται κάποια προβλήματα, τα σημαντικότερα των οποίων είναι τα παρακάτω:

- Η επιλογή της γειτονιάς.

- Η ποιότητα της αρχικής λύσης.
- Η μέθοδος βελτιστοποίησης της αρχικής λύσης

Οι κυριότεροι αλγόριθμοι τοπικής αναζήτησης είναι:

- 2-opt
- 3-opt
- Or-opt
- Swap
- 1-1 exchange
- 1-0 relocate

4.2 Μεθευρετικοί Αλγόριθμοι

Οι μεθευρετικοί αλγόριθμοι[1] αποτελούν μεθόδους επίλυσης οι οποίοι συνδυάζουν διαδικασίες τοπικής αναζήτησης και υψηλότερου επιπέδου στρατηγικές για να δημιουργήσουν μια διαδικασία μπορεί να ξεφύγει από κάποιο τοπικό ελάχιστο, εξερευνάται δηλαδή το πεδίο της λύσης με σκοπό τον εντοπισμό μιας καλύτερης λύσης.

Ονομαστικά κάποιοι από αυτούς του αλγορίθμους είναι οι παρακάτω:

- Προσομοιωμένη Ανόπτηση (Simulated Annealing)
- Περιορισμένη Αναζήτηση (Tabu Search)
- Γενετικοί και Εξελικτικοί Αλγόριθμοι (Genetic and Evolutionary Algorithms)
- Νευρωνικά Δίκτυα (Neural Nets)
- Αλγόριθμος Βελτιστοποίησης Αποικίας Μυρμηγκιών (Ant Colony Optimization)
- Αλγόριθμος Διασκορπισμένης Αναζήτησης (Scatter Search)

- Διαδικασία Άπληστης Τυχοποιημένης Προσαρμοστικής Αναζήτησης (Greedy Randomized Adaptive Search Procedure)
- Αλγόριθμος της Διαφορικής Εξέλιξης (Differential Evolution)
- Αλγόριθμος Βελτιστοποίησης Σμήνους Σωματιδίων (Particle Swarm Optimization)
- Αλγόριθμος Βελτιστοποίησης Ζευγαρώματος Μελισσών (Honey Bees Mating Optimization)

Ως υποδιαδικασίες των μεθευρετικών αλγορίθμων είναι σύνηθες να χρησιμοποιούνται πιο παραδοσιακοί ευρετικοί αλγόριθμοι. Είναι επίσης πιθανό να επιτραπούν σε κάποιο μεθευρετικό αλγόριθμο και βήματα που οδηγούν σε μη εφικτή ενδιάμεση λύση σε κάποιο βήμα του αλγορίθμου για να αποφύγουμε κάποιο τοπικό ελάχιστο και η ολική λύση που θα καταλήξουμε να είναι καλύτερη.

Επιπροσθέτως, κάτι που αξίζει να σημειωθεί, είναι το γεγονός ότι αυτοί οι αλγόριθμοι στηρίζονται σε διαδικασίες που συμβαίνουν στην φύση. Τα στοιχεία που χρησιμοποιούνται και μεταφορικά τα παρατηρούμε και στη φύση είναι τα εξής[1]:

- Χρησιμοποιείται αριθμός από επαναληπτικές δοκιμές
- Εμπεριέχουν έναν ή περισσότερους πράκτορες (νευρώνες, μόρια, χρωμοσώματα, μυρμήγκια, μέλισσες)
- Λειτουργούν στηριζόμενοι σε μηχανισμούς συνεργασίας και ανταγωνισμού
- Περιλαμβάνουν διαδικασίες αυτό-τροποποιήσεων των ευρετικών παραμέτρων ή ακόμα και αναπαράσταση του προβλήματος.

Τα χαρακτηριστικά των μεθευρετικών αλγορίθμων είναι τα ακόλουθα:

- Μοντελοποιούν ένα φαινόμενο που υφίσταται στη φύση

- Μπορούν να μεταφερθούν εύκολα σε παράλληλη μορφή
- Είναι προσαρμοστικοί Αλγόριθμοι.

Κεφάλαιο 5

Προσομοιωμένη Ανόπτηση

Ο αλγόριθμος ονομάστηκε έτσι από την αναλογία ανάμεσα στην προσομοίωση της ανόπτησης των υλικών και την στρατηγική επίλυσης προβλημάτων συνδυαστικής βελτιστοποίησης. Η ανόπτηση σημαίνει ότι ένα στερεό υλικό θερμαίνεται μέχρι το σημείο τήξεως του και εν συνεχεία ψύχεται αργά με σκοπό το υλικό να πέσει στο χαμηλότερο στάδιο ενέργειας όταν τελειώσει η ψύξη. Στην συνδυαστική βελτιστοποίηση αυτό μπορεί να αναπαρασταθεί με την κίνηση κάποιου κριτηρίου στον εφικτό χώρο αναζήτησης ούτως ώστε όταν τελειώσει ο αλγόριθμος να έχουμε μία εφικτή λύση.

Η προσομοιωμένη ανόπτηση λοιπόν χρησιμοποιεί μία γειτονιά N(s) καινούριων καταστάσεων, που μπορεί να φτάσει κάποιος από την τρέχουσα κατάσταση s. Ο αλγόριθμος λειτουργεί παίρνοντας μία αρχική κατάσταση s_0 και μία τυχαία διαταραχή στην αρχική λύση αυτή, έτσι γεννιέται μία καινούργια λύση s'. Το αποτέλεσμα της αντικειμενικής αυτής συνάρτησης είναι το δ=f(s')-f(s_0). Εάν λοιπόν το δ<0 τότε η καινούργια κατάσταση είναι αποδεκτή, εάν δεν είναι αρνητική, δηλαδή δ>=0, τότε η νέα κατάσταση είναι αποδεκτή με κάποια πιθανότητα. Η πιθανότητα αυτή προέρχεται από τους νόμους της θερμοδυναμικής. Η πιθανότητα p(δ)=e-δ/t αύξησης στην ενέργεια της ποσότητας δ στη θερμοκρασία t (κανονικά η πιθανότητα είναι p(δ)=e-δ/kt, με το k να αναφέρεται στην σταθερά Boltzmann αλλά στην συνδυαστική βελτιστοποίηση δεν χρησιμοποιείται και για αυτό τον λόγο μπορεί να αναιρεθεί). Η προσομοιωμένη ανόπτηση λοιπόν δίνει στο πρόγραμμα επίλυσής την δυνατότητα να αποδεχτεί ακόμα και κάποιες μικρές αυξήσεις στην τιμή της αντικειμενικής συνάρτησης αλλά με κάποια συγκεκριμένη πιθανότητα. Με τον τρόπο αυτό η προσομοιωμένη ανόπτηση δίνει την δυνατότητα για την αποδοχή κάποιων μικρών αυξήσεων της αντικειμενικής ελέγχοντας την πιθανότητα αποδοχής διαμέσου της θερμοκρασίας. Έτσι λοιπόν σε υψηλότερες θερμοκρασίες είναι πιο πιθανό μια λύση με χειρότερη τιμή στη συνάρτηση κόστους να

είναι αποδεκτή σαν η καινούργια κατάσταση του προβλήματος. Χρησιμοποιείται πάντα μια συνάρτηση μείωσης θερμοκρασίας όπου σε κάθε επανάληψη μειώνει την εκάστοτε θερμοκρασία. Η

συνάρτηση μαζί με την αρχική θερμοκρασία είναι τα δύο δεδομένα που καθορίζουν πότε και πόσο θα μειωθεί η θερμοκρασία, άρα και πότε θα βρεθεί τελικά το βέλτιστο τελικό αποτέλεσμα. Υπάρχουν πολλοί τρόποι περιγραφής της συνάρτησης μείωσης της θερμοκρασίας, μερικοί από αυτούς είναι οι εξής:

- Η εκθετική μείωση περιλαμβάνει μια συνάρτηση θερμοκρασίας της μορφής α(t)=txα όπου α=0.95 ή α=0.98.
- Το προσαρμοστικό πρόγραμμα ψύχρανσης κάνει δυναμική επιλογή του t.
- Η ύπαρξη μιας καθορισμένου μήκους συνάρτησης α(t)=α*t(α<1), με 0.8 ? α ? 0.99
- Η τέταρτη προσέγγιση, που όμως λόγω του υπολογιστικού φόρτου που έχει, σπάνια χρησιμοποιείται είναι η λογαριθμική προσέγγιση.

Το τυπικό κριτήριο σύγκλισης για την προσομοιωμένη ανόπτηση είναι ένα από τα ακόλουθα:

- Η θερμοκρασία έχει φτάσει σε προκαθορισμένο χαμηλό επίπεδο.
- Αριθμός από επαναλήψεις με διαρκώς μειωμένη θερμοκρασία πέρασε δίχως να αποδεχτούμε τη λύση αυτή..
- Η αναλογία των αποδεκτών κινήσεων έπεσε κάτω από μια δεδομένη τιμή.
- Τέλος, πραγματοποιήθηκε μέγιστος αριθμός προκαθορισμένων επαναλήψεων.

Στην προσομοιωμένη ανόπτηση υπάρχουν πολλοί παράμετροι που μπορούν να ρυθμιστούν και να δοκιμαστούν. Όπως ο καθορισμός της γειτονιάς όπου θα πραγματοποιηθεί η αναζήτηση, η συνάρτηση μείωσης της θερμοκρασίας και ο αριθμός των εσωτερικών επαναλήψεων. Είναι επίσης πιθανό να υπάρχει στρατηγική αύξησης της θερμοκρασίας ή να επιτρέπεται επιπλέον αναζήτηση σε ένα τοπικό ελάχιστο. Ο βασικός αλγόριθμος προσομοιωμένης ανόπτησης παρουσιάζεται στη συνέχεια.

Αλγόριθμος Προσομοιωμένη Ανόπτηση
Αρχικοποίηση
Επέλεξε μια αρχική λύση s_0
Επέλεξε μια αρχική θερμοκρασία t_0
Επέλεξε μια συνάρτηση μείωσης της θερμοκρασίας $\alpha(t)$
repeat
    repeat
        τυχαία επιλογή μιας γειτονιάς $s \in N(s_0)$
        $\delta = f(s) - f(s_0)$
        if $\delta < 0$ ***then***
            $s_0 = s$
        else
            δημιουργούμε τυχαία x, ομοιόμορφα κατανεμημένα στην ακτίνα $(0, 1)$
            if $x < e^{-\frac{\delta}{t}}$ ***then***
                $s_0 = s$
            endif
        endif
    until ο μέγιστος αριθμός επαναλήψεων ολοκληρωθεί
    $t = \alpha(t)$
until κάποιο κριτήριο τερματισμού να ικανοποιηθεί

Κεφάλαιο 6 Παρουσίαση προβλήματος και υλοποίηση αλγορίθμου

Καθορισμός προβλήματος

Στο πρόβλημα της παρούσης εργασίας καλούμαστε να επιλύσουμε το πρόβλημα δρομολόγησης σχολικών λεωφορείων με χρήση ευρετικού αλγορίθμου προσομοιωμένης ανόπτησης και τοπικής αναζήτησης σε τρία διαφορετικά σενάρια ανάλογα το είδος της περιοχής (αστική περιοχή ή προάστιο) και την χωρητικότητα των λεωφορείων.

Στο πρώτο σενάριο, χρησιμοποιούμε λεωφορεία με μικρή χωρητικότητα σε προαστιακή περιοχή όπου θεωρούμε το σπίτι κάθε μαθητή ως διαφορετική στάση.

Στο δεύτερο σενάριο χρησιμοποιούμε λεωφορεία μεγαλύτερης χωρητικότητας πάλι σε προαστιακή περιοχή όπου και πάλι θεωρούμε το σπίτι κάθε μαθητή ως διαφορετική στάση.

Στο τελευταία σενάριο χρησιμοποιούμε και πάλι λεωφορεία μεγαλύτερη χωρητικότητας αλλά αυτήν την φορά σε αστικό περιβάλλον όπου οι μαθητές κατανέμονται σε διαφορετικές τυχαία προκαθορισμένες στάσεις ανάλογα της διεύθυνσης κατοικίας τους.

Ακολουθεί παρακάτω η υλοποίηση του αλγορίθμου**.**

Εισαγωγή

Αρχικά, εισάγουμε τα δεδομένα από το έγκυρο αρχείο Excel στο οποίο περιέχονται, το “**DataTable.xlsx**”.

Για να μπορέσουμε να εισαγάγουμε τα δεδομένα του αρχείου, σε αντίστοιχες μεταβλητές πινάκων, χρησιμοποιούμε την συνάρτηση “**readtable()**”.

Ως παραμέτρους, ορίζουμε το όνομα του αρχείου που αναφέρθηκε, τον τύπο εισαγωγής "**Sheet**", καθώς και το όνομα του sheet από τα οποίο αντλούμε τα δεδομένα, το "**CMT9**". Τέλος, ορίζουμε την εμβέλεια "**Range**", καθώς και τα κελιά της εμβέλειας αυτής.

Επαναλαμβάνουμε την συνάρτηση για κάθε τιμή ή πίνακα δεδομένων που θέλουμε να πάρουμε.

```
NodesNum = readtable('DataTable.xlsx', 'Sheet', 'CMT9', 'Range', 'A2:A2');
BussCapacity = readtable('DataTable.xlsx', 'Sheet', 'CMT9', 'Range',
'B2:B2');
BussServiceTime = readtable('DataTable.xlsx', 'Sheet', 'CMT9', 'Range',
'C2:C2');
ClientServiceTime = readtable('DataTable.xlsx', 'Sheet', 'CMT9', 'Range',
'D2:D2');
NodeCoords = readtable('DataTable.xlsx', 'Sheet', 'CMT9', 'Range',
'E2:F152');
ClientDemand = readtable('DataTable.xlsx', 'Sheet', 'CMT9', 'Range',
'G2:G152');
```

Για να μπορέσουμε να επεξεργαστούμε, όμως, αυτά τα δεδομένα, θα πρέπει το table να γίνει μάτριξ της **MATLAB**, και αυτό το επιτυγχάνουμε με την συνάρτηση "**table2array()**".

Επομένως, για κάθε ένα από τα tables δεδομένων που δημιουργήσαμε, ορίζουμε την τιμή τους ως την συνάρτηση της τιμής αυτής.

```
NodesNum = table2array(NodesNum);
BussCapacity = table2array(BussCapacity);
BussServiceTime = table2array(BussServiceTime);
ClientServiceTime = table2array(ClientServiceTime);
NodeCoords = table2array(NodeCoords);
ClientDemand = table2array(ClientDemand);
```

Πίνακας κόστους

Για να μπορέσουμε να υλοποιήσουμε, αρχικά, τον άπληστο αλγόριθμο του πλησιέστερου γείτονα, είναι απαραίτητο να δημιουργήσουμε τον πίνακα κόστους "**NodeCosts**", στον οποίο θα περιέχεται το κόστος του κάθε κόμβου, βάση των αποστάσεων τους.

Δημιουργούμε μία μεταβλητή πίνακα, αρχικοποιώντας τον με μηδενικά. Έπειτα, μέσω μίας δομής "**for**" θα πραγματοποιείται για κάθε κόμβο μία

ακόμη “**for**”, ελέγχοντας, έτσι, κάθε πιθανό συνδυασμό αποστάσεων των κόμβων. Το κόστος απόστασης των κόμβων ορίζεται ως:

Το οποίο μεταφράζεται στον κώδικα ως:

```
NodeCosts = zeros(NodesNum);
for i = 1:NodesNum
       for j = 1:NodesNum
         NodeCosts(j,i) = sqrt((NodeCoords(i,1)-
NodeCoords(j,1))^2+(NodeCoords(i,2)-NodeCoords(j,2))^2);
       end
end
```

Άπληστος αλγόριθμος

Γενικά

Αρχικά, δεσμεύουμε τις απαραίτητες μεταβλητές που θα χρειαστούμε.

- Ένας μετρητής κόμβων “**NodesCounter**”, ο οποίος αρχικοποιείται με την τιμή **1**, αφού η αφετηρία είναι δεδομένη, και επομένως ελεγχμένη.
- Ένας πίνακας που θα εξυπηρετεί την σημείωση των ελεγμένων κόμβων “**NodesCheck**”, του οποίου η πρώτη τιμή (αφετηρία) παίρνει την τιμή **1**, ενώ ο υπόλοιπος πίνακας αρχικοποιείται με την τιμή **0**.
- Ένα αρχικό κόστος “**RoutesInitCost**” που αρχικοποιείται με την τιμή **0**.
- Μία μεταβλητή για τον αριθμό των λεωφορείων που θα χρειαστούν “**BussNum**”.
- Μία μεταβλητή πίνακα “**RoutesInit**”, στην οποία καταγράφονται οι κόμβοι (στήλες), ανά λεωφορείο (γραμμές).

Έτσι, έχουμε την παρακάτω αρχική δήλωση:

```
NodesCounter = 1;
NodesCheck = zeros(1,NodesNum);
NodesCheck(1) = 1;
RoutesInitCost = 0;
BussNum = 0;
RoutesInit = zeros(NodesNum);
```

Για τον αλγόριθμο του πλησιέστερου γείτονα, φροντίζουμε να εφαρμόσουμε μία δομή “**while**”, κατά την οποία δημιουργούνται διαδρομές, έως ότου να μην απομένουν άλλοι κόμβοι για έλεγχο.

```
while NodesCounter < NodesNum
      ...
end
```

Για κάθε μία διαδρομή που δημιουργείται (επανάληψη της while), η θέση κόμβου “**NodePosition**”, παίρνει την τιμή **1**, δηλαδή τον κόμβο της αφετηρίας, και ο αριθμός των λεωφορείων αυξάνεται κατά μία μονάδα.

Ακόμη, δημιουργείται μία προσωρινή μεταβλητή πίνακα “**TempRoutesInit**”, πάνω στην οποία εφαρμόζονται οι διάφορες λειτουργίες του αλγόριθμου για την δημιουργία της νέας διαδρομής. Με κάθε επανάληψη (νέα διαδρομή) οι τιμές του πίνακα μηδενίζονται, ενώ η πρώτη τιμή (αφετηρία) παίρνει την τιμή 1.

```
NodePosition = 1;
BussNum = BussNum+1;
TempRoutesInit = zeros(1,NodesNum);
TempRoutesInit(1) = 1;
TempBussServiceTime = BussServiceTime;
TempBussCapacity = BussCapacity;
InitFlag = 0;
```

Χρησιμοποιούμε και μία ακέραια μεταβλητή "**InitFlag**" που εξυπηρετεί ως έλεγχος σε μία "**while**", για την εγκυρότητα του διανύσματος και των κόμβων.

```
While InitFlag ~= 1
      ...
end
```

Ταξινόμηση διανύσματος

Για να μπορέσουμε να εφαρμόσουμε αποτελεσματικά των άπληστο αλγόριθμο, θα πρέπει να δημιουργήσουμε δύο ταξινομημένους πίνακες, ένα με το κόστος των κόμβων, και έναν με τους αριθμούς των ίδιων κόμβων. Και οι δύο αντλούν τις τιμές τους και ταξινομούνται βάση του πίνακα κόστους "**NodeCosts**", σε αύξουσα σειρά κόστους.

```
i = 2;
[MinCostRoute, MinCostNum] = sort(NodeCosts(NodePosition,:), 'ascend');
MinCost = MinCostRoute(i);
NodePosition = MinCostNum(i);
```

Έλεγχος επανάληψης κόμβου

Είναι πολύ σημαντικό να ελέγξουμε εάν έχει γίνει επίσκεψη ενός κόμβου, ώστε να μην λαμβάνονται υπόψη κόμβοι, τους οποίους έχουν αναλάβει τα λεωφορεία άλλων διαδρομών. Αυτό το επιτυγχάνουμε με μία "**while**", καθώς και μία "**if**".

Με αυτό τον τρόπο, εάν έχει πραγματοποιηθεί επίσκεψη σε έναν κόμβο (), η τιμή ελαχίστου κόστους, αλλά και αριθμού κόμβου, παίρνουν τις τιμές των αμέσως επόμενων () στους πίνακες ελαχίστων "**MinCostRoute**" και "**MinCostNum**" αντίστοιχα.

```
j = 1;
while j ~= NodesNum
       if NodesCheck(j) == NodePosition
         i = i+1;
    MinCost = MinCostRoute(i);
    NodePosition = MinCostNum(i);
    j = 0;
       end
       j = j+1;
end
```

Έλεγχος εγκυρότητας κόμβου

Για να δούμε εάν μπορεί να προστεθεί ένας κόμβος στην διαδρομή, δημιουργούμε μία μεταβλητή "**TimeFlag**" ως έλεγχο/προδιαγραφή, η οποία λαμβάνει υπόψη τον απαραίτητο χρόνο εξυπηρέτησης του λεωφορείου στον κόμβο, τον χρόνο εξυπηρέτησης των ατόμων, το ελάχιστο κόστος κόμβου που δεν έχει ελεγχθεί, καθώς και το κόστος του κόμβου που βρίσκεται.

Έπειτα, με μία "**if**" ελέγχεται εάν αυτό το "**TimeFlag**" είναι μεγαλύτερο του μηδενός, και εάν ο υπολειπόμενος χώρος στο λεωφορείο αρκεί, για να εξυπηρετήσει την ζήτηση του κόμβου.

Σε περίπτωση που οι παραπάνω προϋποθέσεις πληρούνται, ο κόμβος προστίθεται στο διάνυσμα της προσωρινής διαδρομής "**TempRoutesInit**", και ο κόμβος ορίζεται ως ελεγμένος στον πίνακα "**NodesCheck**".

Η ζήτηση του κόμβου μειώνεται από τον διαθέσιμο χώρο του λεωφορείου, το οποίο συνεχίζει προς το επόμενο σημείο για έλεγχο, ενώ στον χρόνο εξυπηρέτησης του, προστίθεται το χρονικό κόστος του κόμβου.

Ακόμη, στο συνολικό κόστος του άπληστου αλγόριθμου "**RoutesInitCost**" προστίθεται το κόστος του κόμβου, και ο μετρητής κόμβων "**NodesCounter**" αυξάνεται κατά μία μονάδα.

Στην περίπτωση που οι προϋποθέσεις δεν πληρούνται, ή αν ο μετρητής κόμβων φτάσει τον αριθμό των συνολικών κόμβων, η ακέραια μεταβλητή "**InitFlag**" παίρνει την τιμή 1, και η προσωρινή διαδρομή του λεωφορείου θεωρείται ολοκληρωμένη.

```
if ((TempBussCapacity-ClientDemand(NodePosition) >= 0) && (TimeFlag >= 0))
      RoutesInitCost = RoutesInitCost+MinCost;
  TempBussCapacity = TempBussCapacity-ClientDemand(NodePosition);
  TempBussServiceTime = TimeFlag+NodeCosts(NodePosition,1);
  NodesCounter = NodesCounter+1;
      if NodesCounter == NodesNum
            InitFlag = 1;
      end
  for i = 1:NodesNum
        if NodesCheck(i) == 0
          NodesCheck(i) = NodePosition;
      break;
            end
      end
  for i = 1:NodesNum
        if TempRoutesInit(i) == 0
          TempRoutesInit(i) = NodePosition;
          break;
            end
  end
else
      InitFlag = 1;
end
```

Συγχώνευση διαδρομής

Αφότου ολοκληρωθεί η εύρεση των κόμβων μιας διαδρομής, θέλουμε το λεωφορείο να επιστρέφει πίσω στον πρώτο κόμβο, δηλαδή ο τελευταίος κόμβος επίσκεψης του λεωφορείου να είναι η αφετηρία.

Για να το πετύχουμε αυτό, εφαρμόζουμε μία δομή “**for**”, η οποία κάνει σειριακή αναζήτηση στον προσωρινό πίνακα της νέας διαδρομής “**TempRoutesInit**”, και με μία δομή ελέγχου “**if**”, θα αλλάξει την τιμή του πρώτου μηδενικού (κενού) στοιχείου που θα συναντήσει, από 0 σε 1 (κόμβος αφετηρίας).

```
for i = 1:NodesNum
      if TempRoutesInit(i) == 0
            TempRoutesInit(i) = 1;
            break;
      end
end
```

Τέλος, η προσωρινή διαδρομή “**TempRoutesInit**”, η οποία πλέον είναι τελική, συγχωνεύεται στον πίνακα των διαδρομών “**RoutesInit**” μέσω μίας “**for**”, αλλά και μίας “**if**”, που ελέγχει τους κόμβους εισαγωγής να είναι μη μηδενικοί.

```
for i = 1:NodesNum
      if TempRoutesInit(i) ~= 0
            RoutesInit(BussNum, i) = TempRoutesInit(i);
      end
end
```

Αλγόριθμος 3OPT

Γενικά

Για την υλοποίηση του αλγορίθμου 3OPT, χρησιμοποιούμε μία δομή επανάληψης “**for**”, με τον επιθυμητό αριθμό επαναλήψεων (προσέγγιση) του αλγορίθμου. Για χάρη χρόνου στο παράδειγμα του κώδικα μας, θέτουμε τον αριθμό επαναλήψεων .

Ορίζουμε επίσης μία ακέραια μεταβλητή “**OptFlag**” με αρχική τιμή **0**. Η μεταβλητή αυτή εξυπηρετεί ως δείκτης εύρεσης βέλτιστου κόστους.

```
OptFlag = 0;
for LoopNum = 1:5000
      ...
end
```

Η εύρεση των βέλτιστων τομών του αλγορίθμου θα πρέπει να εφαρμόζεται ξεχωριστά για κάθε διαδρομή. Έτσι, έχουμε μία “**for**”, η οποία τρέχει για όσες διαδρομές (**RoutesNum**) δημιουργήθηκαν από τον άπληστο αλγόριθμο.

```
for k = 1:RoutesNum
      ...
end
```

Για κάθε μία διαδρομή, δημιουργούμε έναν προσωρινό διανυσματικό πίνακα με τους κόμβους της διαδρομής.

Για να βρούμε το πραγματικό μέγεθος του διανύσματος της διαδρομής, πραγματοποιούμε σειριακή αναζήτηση, μετρώντας τους μη-μηδενικούς

κόμβους “**ZeroCount**”. Έτσι, δημιουργούμε τον πίνακα με μέγεθος “**TempRoutesNum**”.

```
ZeroCount = 0;
for i = 1:NodesNum
       if RoutesInit(k,i) ~= 0
         ZeroCount = ZeroCount + 1;
  end
end
TempRoutesNum = zeros(1,ZeroCount);
```

Έπειτα, αντικαθιστούμε τις μηδενικές τιμές του με τους κόμβους της αρχικής διαδρομής "**RoutesInit**", και αποθηκεύουμε το μέγεθος του σε μία μεταβλητή "**TempRoutesSize**" για μεγαλύτερη ευκολία.

```
for i = 1:NodesNum
      if RoutesInit(k,i) ~= 0
        TempRoutesNum(i) = RoutesInit(k,i);
  end
end
TempRoutesSize = size(TempRoutesNum);
```

Εύρεση τομών

Αφού δημιουργήσουμε τους απαραίτητους πίνακες, δημιουργούμε μία μεταβλητή πίνακα "**SectionAll**", όπου θα αποθηκεύσουμε τις τυχαίες τομές. Ακόμη, δεσμεύουμε μία ακέραια μεταβλητή "**SectionFlag**" που εξυπηρετεί ως boolean flag με αρχική τιμή **1**.

Μέσω μίας "**while**" γίνεται τυχαία επιλογή τομών, έως ότου η επιλογή αυτή να είναι έγκυρη, δηλαδή το "**SectionFlag**" να πάρει την τιμή **0**.

Για την τυχαία επιλογή, χρησιμοποιούμε την συνάρτηση "**randi()**", ενώ για την ταξινόμηση χρησιμοποιούμε την συνάρτηση "**sort()**". Η ταξινόμηση γίνεται με αύξουσα σειρά.

```
SectionFlag = 1;
while SectionFlag ~= 0
      SectionAll = randi(TempRoutesSize(2)-1,1,3);
  SectionAll = sort(SectionAll, 'ascend');
  if ((SectionAll(2)-SectionAll(1) <= 1) || (SectionAll(3)-SectionAll(2) <=
1) || (SectionAll(1) == SectionAll(2)) || (SectionAll(2) == SectionAll(3))
|| (SectionAll(3) == SectionAll(1)))
            SectionFlag = 0;
  end
end
```

Έπειτα, δημιουργούμε δύο προσωρινούς πίνακες "**TempSection1**" και "**TempSection2**", βάση του "**SectionAll**" που βρήκαμε προηγουμένως. Αντιστρέφουμε τους πίνακες για μεγαλύτερη ευκολία στην επεξεργασία μέσω της συνάρτησης "**flipr()**".

```
TempSection1 = zeros(1, (SectionAll(2)-SectionAll(1)));
TempSection2 = zeros(1, (SectionAll(3)-SectionAll(2)));
TempSection1 = fliplr(TempSection1);
TempSection2 = fliplr(TempSection2);
```

Επιπλέον, συμπληρώνουμε τα στοιχεία των δύο πινάκων μέσω πολλαπλών δομών της “**for**”, βάση πάλι το “**SectionAll**”.

```
j = 1;
for i = (SectionAll(1)+1):SectionAll(2)
       TempSection1(j) = i;
  j = j+1;
end
j = 1;
for i = (SectionAll(2)+1):SectionAll(3)
       TempSection2(j)=i;
  j = j+1;
end
```

Εύρεση & Συγχώνευση Διανυσμάτων

Βάση των τομών που έχουν βρεθεί, δημιουργούνται οι κατάλληλοι συνδυασμοί διανυσμάτων, οι οποίοι αποθηκεύονται στον δισδιάστατο πίνακα “**TempRoutesOpt**”, μέσω πολλαπλών δομών επαναλήψεων της “**for**”.

```
for i = 1:(SectionAll(1))
       TempRoutesOpt(i) = TempRoutesNum(i);
end
j = 1;
for i = (SectionAll(1)+1):(SectionAll(2))
       TempRoutesOpt(i) = TempRoutesNum(TempSection1(j));
       j = j+1;
end
j = 1;
for i = (SectionAll(2)+1):(SectionAll(3))
       TempRoutesOpt(i) = TempRoutesNum(TempSection2(j));
  j = j + 1;
end
for i = (SectionAll(3)+1):TempRoutesSize(2)
       TempRoutesOpt(i) = TempRoutesNum(i);
end
```

Τα διανύσματα αυτά συγχωνεύονται, με σκοπό να υπολογίσουμε και να συγκρίνουμε τα κόστη των δύο αλγορίθμων.

```
for i = 1:(TempRoutesSize(2))
       if TempRoutesOpt(i) ~= 0
         RoutesOpt(k,i) = TempRoutesOpt(i);
  end
end
```

Υπολογισμός & Σύγκριση Κόστους

Δεσμεύουμε μία μεταβλητή "**RoutesOptCost**", στην οποία θα αντιγράφεται αρχικά η τιμή κόστους "**RoutesInitCost**" του άπληστου αλγορίθμου, καθώς και μία προσωρινή μεταβλητή "**TempOptCost**", όπου θα υπολογίζεται το κόστους της κάθε επανάληψης του 3OPT.

```
RoutesOptCost = RoutesInitCost;
TempOptCost = 0;
```

Το κόστος που προκύπτει από την κάθε λύση του 3OPT, υπολογίζεται σειριακά με βάση τον πίνακα γενικού κόστους των κόμβων "**NodeCosts**".

```
for i = 1:RoutesNum
       for j = 1:NodesNum-1
         if (RoutesOpt(i,j) ~= 0) && (RoutesOpt(i,j+1) ~= 0)
           TempOptCost = TempOptCost+NodeCosts(RoutesOpt(i,j),
RoutesOpt(i,j+1));
    end
  end
end
```

Ελέγχουμε με μία "**if**" εάν το κόστος των νέων διανυσμάτων είναι βέλτιστο σε σχέση με αυτό του άπληστου αλγορίθμου (), και σε περίπτωση που ισχύει, ο διανυσματικός πίνακας "**RoutesOptCost**" παίρνει τις τιμές του "**TempOptCost**", ο "**MinRoutesOpt**" γίνεται ίσος με τον "**RoutesOpt**", και η μεταβλητή αναφοράς "**OptFlag**" γίνεται ίση με **1**.

Σε κάθε περίπτωση, ο αλγόριθμος 3OPT συνεχίζει να παράγει λύσεις μέχρι να ολοκληρωθεί ο αριθμός των επαναλήψεων που έχουμε θέσει.

Έλεγχος Εγκυρότητας 3OPT

Βάση του “**OptFlag**” που ορίσαμε στον αλγόριθμο του 3OPT, ελέγχουμε εάν βρέθηκαν βέλτιστες διαδρομές ως προς το κόστος. Σε περίπτωση που έχουν βρεθεί (), ο τελικός πίνακας διαδρομών παίρνει τις τιμές του πίνακα "**MinRoutesOpt**". Σε αντίθετη περίπτωση, ο τελικός πίνακας παίρνει τις τιμές της αρχικής λύσης του άπληστου αλγόριθμου "**RoutesInit**.

```
if OptFlag == 1
       GivenRoutes = MinRoutesOpt;
else
  GivenRoutes = RoutesInit;
end
```

Διαμόρφωση Πίνακα Διαδρομών

Για την καλύτερη διαχείριση του πίνακα διαδρομών, διαγράφουμε τις περιττές θέσεις μηδενικών τιμών, οι οποίες δεν χρειάζονται.

Έτσι, έχουμε έναν τελικό πίνακα ακεραίων τιμών, οι γραμμές του οποίου αντιστοιχούν στον αριθμό των διαδρομών που πραγματοποιούνται. Η εύρεση και διαγραφή των θέσεων γίνεται με την χρήση πολλαπλών δομών "**for**".

Η κάθετη εύρεση πραγματοποιείται βρίσκοντας την μέγιστη μη-μηδενική στήλη "**MaxCol**".

```
MaxCol=2;
for i=2:length(GivenRoutes(1,:))
    for j=2:length(GivenRoutes(:,i))
        if GivenRoutes(j,i) == 1
            if i > MaxCol
                MaxCol = i;
            end
        end
    end
end
```

Αντίστοιχα, πραγματοποιείται και η οριζόντια εύρεση περιττών στοιχείων, όπου υπολογίζεται ο αριθμός της μέγιστης μη-μηδενικής γραμμής "**MaxRow**".

```
MaxRow=1;
for i=1:length(GivenRoutes(:,1))
    if GivenRoutes(i,1) == 0
        MaxRow = i;
        break;
    end
end
```

Αφού βρεθούν τα όρια των μη-μηδενικών τιμών "**MaxCol**" και "**MaxRow**" στον πίνακα διαδρομών, διαγράφουμε τις περιττές τιμές βάση των ορίων αυτών.

```
GivenRoutes(:,MaxCol+1:end) = [];
GivenRoutes(MaxRow:end,:) = [];
```

Προσομοιωμένη Ανόπτηση

Γενικά

Για να μπορέσουμε να υλοποιήσουμε μία μορφή της προσομοιωμένης ανόπτησης, θα επιλέξουμε μία αρχική προσέγγιση, βάση της οποίας θα ελεγχθούν οι διάφοροι πιθανοί συνδυασμοί. Για χάρη χρόνου στον κώδικα μας, θέτουμε προσέγγιση (αριθμό επαναλήψεων) .

Έτσι, για κάθε διαδρομή θα δημιουργείται ένα προσωρινό κόστος, καθώς και ένας προσωρινός πίνακας κόμβων, πάνω στον οποίο θα εφαρμόζεται ο αλγόριθμος της προσομοιωμένης ανόπτησης.

```
LoopNum = 50;
for i=1:length(GivenRoutes(:,1))
      ...
end
```

Εύρεση Αριθμού Κόμβου Διαδρομής

Για κάθε διαδρομή, γίνεται εύρεση των αριθμών των κόμβων που περιλαμβάνει, με τον αριθμό αυτό να αποθηκεύεται στην μεταβλητή "**MaxCol**".

Χρησιμοποιούμε μία "**for**" πάνω στο διάνυσμα , όπου εφαρμόζεται σειριακή αναζήτηση από τον δεύτερο κόμβο και μετά.

Μόλις βρεθεί κάποιος μηδενικός κόμβος, η "**MaxCol**" παίρνει την τιμή , δηλαδή την θέση του προηγούμενου μη-μηδενικού κόμβου, και η επανάληψη σπάει με μία "**break**".

```
for j=2:length(GivenRoutes(i,:))
      if GivenRoutes(i,j) == 0
        MaxCol = j-1;
    break;
      end
end
```

Επιλογή Διαδρομής & Αρχικό Κόστος

Προτού εφαρμοστεί η προσομοιωμένη ανόπτηση, αποθηκεύουμε την διαδρομή στην οποία βρισκόμαστε, σε έναν προσωρινό πίνακα κόμβων “**TempRoute**”. Ακόμη, αποθηκεύουμε το αρχικό κόστος της διαδρομής σε μία προσωρινή μεταβλητή “**TempRouteCost**”, η οποία μηδενίζεται με κάθε νέα διαδρομή.

Έπειτα χρησιμοποιούμε μία δομή επανάληψης "**for**", για να υπολογίσουμε το αρχικό κόστος, βάση του γενικού πίνακα "**NodeCosts**".

```
TempRoute = GivenRoutes(i,1:MaxCol);
TempRouteCost = 0;
for j=2:MaxCol
	TempRouteCost = TempRouteCost + NodeCosts(TempRoute(j),TempRoute(j-
1));
end
```

Εφαρμογή Αλγορίθμου & Σύγκριση Κόστους

Για την εφαρμογή της προσομοιωμένης ανόπτησης, χρησιμοποιούμε μία "**for**" με αριθμό επαναλήψεων "**LoopNum**", την προσέγγιση δηλαδή που έχουμε θέσει.

```
for j=1:LoopNum
	...
end
```

Για κάθε μία επανάληψη, δημιουργείται ένας πίνακας διαδρομών "**TempRouteRand**". Μέσα σε αυτό τον πίνακα αποθηκεύονται οι αρχικοί κόμβοι της διαδρομής (πλην της αφετηρίας στην αρχή και το τέλος). Έπειτα, γίνεται μία τυχαία ανακατανομή των κόμβων, με την αφετηρία να προστίθεται ξανά στην αρχή και το τέλος του διανύσματος.

```
TempRouteRand = GivenRoutes(i,2:MaxCol-1);
TempRouteRand = TempRouteRand(randperm(length(TempRouteRand)));
TempRouteRand = [1 TempRouteRand 1];
```

Αφού δημιουργηθεί η νέα διαδρομή, ορίζουμε και μία μεταβλητή "**TempRouteRandCost**" με αρχική τιμή **0**. Όπως και στο κόστος της αρχικής διαδρομής, έτσι και εδώ με μία "**for**", υπολογίζουμε το κόστος της νέας διαδρομής.

```
TempRouteRandCost = 0;
for k=2:MaxCol
	TempRouteRandCost = TempRouteRandCost +
NodeCosts(TempRouteRand(k),TempRouteRand(k-1));
end
```

Έπειτα πραγματοποιούμε έναν απλό έλεγχο με "**if**", και συγκρίνουμε τα κόστη των δύο διαδρομών. Εάν το κόστος της νέας διαδρομής είναι βέλτιστο, τότε η αρχική διαδρομή αντικαθίσταται από την καινούργια. Σε αντίθετη περίπτωση, μένει ως έχει.

```
if TempRouteRandCost < TempRouteCost
       SizeLeft = length(GivenRoutes(i,:))-MaxCol;
  SizeFill = zeros(1, SizeLeft);
  TempRouteRand = [TempRouteRand SizeFill];
  GivenRoutes(i,:) = TempRouteRand;
end
```

Πίνακας Τελικών Διαδρομών

Δημιουργία Διαγραμματικού Πίνακα

Για να μπορέσουμε να εμφανίσουμε διαγραμματικά τις τελικές διαδρομές που πήραμε ως αποτέλεσμα των παραπάνω αλγορίθμων, δημιουργούμε έναν δισδιάστατο πίνακα "**FinalRoutes**", στον οποίο θα έχουμε τις συντεταγμένες των κόμβων, αντί για τον αριθμό τους.

Το μέγεθος του βασίζεται στο μέγεθος του πίνακα "**GivenRoutes**", με τον αριθμό των στηλών να είναι στο τετράγωνο, αφού αποθηκεύουμε συντεταγμένες. Οι τιμές του αρχικοποιούνται με μηδέν.

```
FinalRoutes = zeros(length(GivenRoutes(1,:)), length(GivenRoutes(:,1))*2);
```

Για να γίνει η μετάφραση των κόμβων, χρησιμοποιούμε τον πίνακα "**NodeCoords**", μέσω του οποίου οι αριθμοί των κόμβων αντιστοιχίζονται σε συντεταγμένες.

```
for i=1:length(GivenRoutes(:,1))
       for j=1:length(GivenRoutes(1,:))
         TempRouteId = GivenRoutes(i,j);
    if TempRouteId ~= 0
          FinalRoutes(j,i*2-1) = NodeCoords(TempRouteId, 1);
      FinalRoutes(j,i*2) = NodeCoords(TempRouteId, 2);
    end
       end
end
```

Εμφάνιση Διαδρομών

Αφού δημιουργήσουμε τον πίνακα των συντεταγμένων, και προτού εμφανίσουμε το διάγραμμα, πραγματοποιούμε σειριακή αναζήτηση στον πίνακα, αλλάζοντας οποιαδήποτε μη-έγκυρη μηδενική τιμή βρούμε σε "**NaN**", ώστε να μην εμφανιστεί στο τελικό αποτέλεσμα.

Τέλος, με μία ακόμη "**for**", χρησιμοποιούμε την συνάρτηση "**plot()**" για να μεταφράσουμε τις συντεταγμένες του πίνακα σε διάγραμμα.

```
for i=1:length(FinalRoutes(1,:))
       for j=1:length(FinalRoutes(:,i))
         if FinalRoutes(j,i) == 0
           FinalRoutes(j,i) = NaN;
    end
  end
end

for i=1:(length(FinalRoutes(1,:))/2)
       if mod(i, 2) ~= 0
         hold on
    plot(FinalRoutes(:,i*2-1), FinalRoutes(:,i*2));
    hold off
  end
end
```

Κεφάλαιο 7 Παρουσίαση και σχολιασμός αποτελεσμάτων

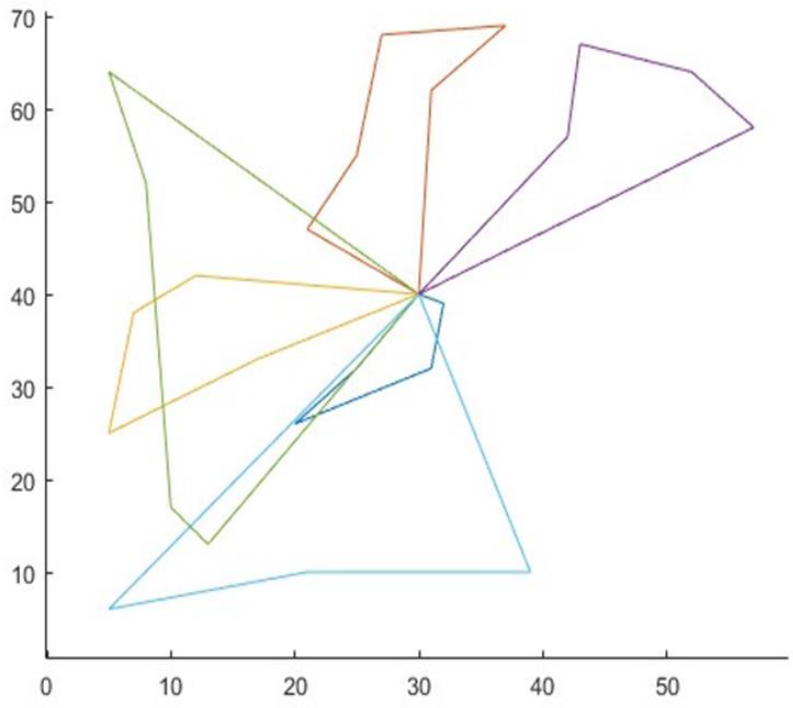
Εφαρμογή Δεδομένων / Σχόλια.

Στην δική μας δοκιμή του αλγορίθμου, έχουμε τρία παραδείγματα εκτέλεσης του αλγορίθμου, όπου τα δεδομένα αντλούνται από τρία αρχεία excel (.xlsx).

Πρώτο σενάριο

Στο πρώτο σενάριο χρησιμοποιούμε λεωφορεία μικρής χωρητικότητας σε προαστιακή περιοχή όπου θεωρούμε το σπίτι κάθε μαθητή ως διαφορετική στάση.

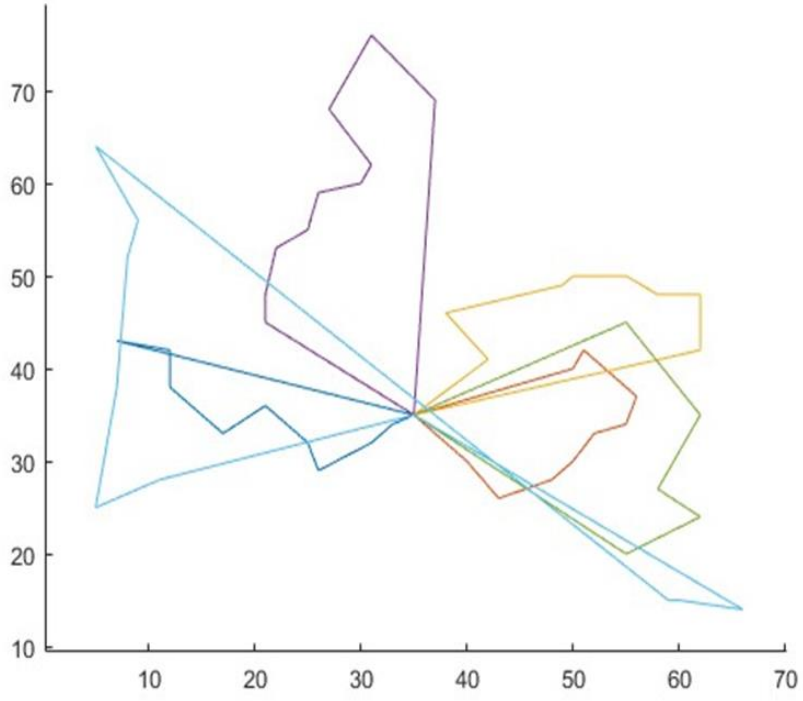
Το αρχείο δεδομένων, περιέχει το μικρότερο πλήθος συντεταγμένων, το οποίο είναι **50 κόμβοι**. Εδώ η χωρητικότητα του λεωφορείου είναι η μικρότερη, καθώς ανέρχεται στις **25 θέσεις**. Το **Max Route Time** είναι **200**, ενώ το **Service Time** είναι **10**.



Δεύτερο σενάριο

Στο δεύτερο σενάριο χρησιμοποιούμε λεωφορεία μεγαλύτερης χωρητικότητας σε αστικό περιβάλλον όπου οι μαθητές κατανέμονται σε διαφορετικές τυχαία προκαθορισμένες στάσεις ανάλογα της διεύθυνσης κατοικίας τους.

Στο δεύτερο αρχείο διαδρομών που έχουμε δημιουργήσει, το οποίο περιέχει τις συντεταγμένες για **100 κόμβους**. Ο διαθέσιμος χώρος του κάθε λεωφορείου, είναι **50 θέσεις**. Το **Max Route Time** ισούται με **260**, και το **Service Time** ισούται με **5**.

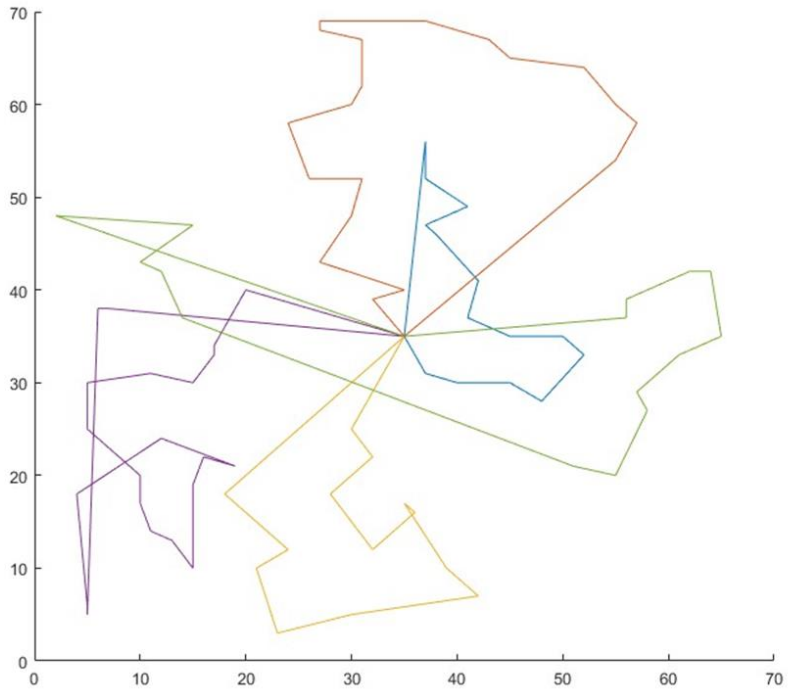


Τρίτο Σενάριο

Στο τελευταία σενάριο χρησιμοποιούμε και πάλι λεωφορεία μεγαλύτερης χωρητικότητας σε προαστιακή περιοχή όπου και πάλι θεωρούμε το σπίτι κάθε μαθητή ως διαφορετική στάση.

Το αρχείο δεδομένων περιέχει δεδομένα για **151 κόμβους**. Η χωρητικότητα (capacity) του λεωφορείου ανέρχεται στις **100 θέσεις**, ενώ τα **Max Route Time** και **Service Time** είναι **280** και **5** αντίστοιχα.

Λόγω της περιορισμένης προσέγγισης που έχουμε θέσει, για χάρη χρόνου, στους αλγορίθμους του 3OPT και της Προσομοιωμένης Ανόπτησης, η τελική λύση διαδρομών που λαμβάνουμε από το πρόγραμμα είναι η εξής (διαγραμματικά):



Το κάθε χρώμα αντιστοιχεί σε μία διαφορετική διαδρομή λεωφορείου.

Τελικό Συμπέρασμα

Από τα παραπάνω διαγράμματα είναι εμφανές πως όσο περισσότερο αυξάνεται η αναλογία διαδρομών και διαθέσιμων θέσεων, τόσο πιο ομαλές είναι οι διαδρομές που καλούνται να καλύψουν τα λεωφορεία.

Εξίσου σημαντικοί βέβαια, είναι και οι παράγοντες ζήτησης σε κάθε στάση, σε συνδυασμό με τον μέσο χρόνο εξυπηρέτησης και αναμονής των λεωφορείων.

Συμπερασματικά, το σενάριο που απέδωσε τα καλύτερα αποτελέσματα είναι το τρίτο με **50 κόμβους** και **25 θέσεις**. Το αμέσως επόμενο καλύτερο σενάριο είναι το δεύτερο με **100 κόμβους** και **50 θέσεις**, ενώ το σενάριο με τα «χειρότερα» αποτελέσματα είναι το πρώτο με **151 κόμβους** και **100 θέσεις**.

Βιβλιογραφία

[1] Ιωάννης Μαρινάκης / Αθανάσιος Μυγδαλάς. ΣΧΕΔΙΑΣΜΟΣ ΚΑΙ ΒΕΛΤΙΣΤΟΠΟΙΗΣΗ ΤΗΣ ΕΦΟΔΙΑΣΤΙΚΗΣ ΑΛΥΣΙΔΑΣ. Εκδόσεις σοφία, Θεσσαλονίκη 2008

[2] Σφηναρολάκης Γ. (2021) Αλγόριθμος Προσομοιωμένης Ανόπτησης για το Δυναμικό Πρόβλημα Δρομολόγησης Οχημάτων (DVRP). Διπλωματική Εργασία, Επιβλέπων καθηγητής: Μαρινάκης Ιωάννης, Τμήμα Μηχανικών Παραγωγής και Διοίκησης, Πολυτεχνείο Κρήτης

[3] The school bus routing problem: A review. Junhyuk Park, Byung-In Kim (2009)