ΠΟΛΥΤΕΧΝΕΙΟ ΚΡΗΤΗΣ

ΣΧΟΛΗ ΜΗΧΑΝΙΚΩΝ ΠΑΡΑΓΩΓΗΣ ΚΑΙ ΔΙΟΙΚΗΣΗΣ

Θέμα Διπλωματικής Εργασίας: Επίλυση του Ανοιχτού Προβλήματος Δρομολόγησης Οχημάτων με Ελαχιστοποίηση της Κατανάλωσης Καυσίμου με χρήση του Αλγορίθμου Βελτιστοποίησης Αποικίας Μυρμηγκιών.

Ονοματεπώνυμο	**ΑΜ**
Καζαντζής Αντώνιος	2017010111

Επιβλέπων Καθηγητής: Ιωάννης Μαρινάκης

Χανιά 2023

Περιεχόμενα

Ευχαριστίες

Αρχικά, θα ήθελα να ευχαριστήσω τον καθηγητή μου Ιωάννη Μαρινάκη για την βοήθεια του. Επίσης, θέλω να ευχαριστήσω τον Νικόλαο Κυριακάκη για όλη την βοήθεια και την πάρα πολύ καλή συνεργασία που είχαμε. Τέλος, θα ήθελα να ευχαριστήσω την οικογένεια μου και τους φίλους μου για την στήριξη τους όλα αυτά τα χρόνια της φοιτητικής μου ζωής.

1) Εισαγωγή

Στην σημερινή εποχή ο τομέας της εφοδιαστικής αλυσίδας παίζει μεγάλο ρόλο στην εξέλιξη μιας εταιρείας και σε συνδυασμό με την πληθώρα των περιβαλλοντικών προβλημάτων της εποχής, η παρούσα διπλωματική εργασία επιλύει μία παραλλαγή του Ανοιχτού Προβλήματος Δρομολόγησης Οχημάτων (Open Vehicle Routing Problem – OVRP) με στόχο την Ελαχιστοποίηση της Κατανάλωσης Καυσίμων. Ένα πολύ σημαντικό πρόβλημα που επιβαρύνει το περιβάλλον είναι η ρύπανση του ατμοσφαιρικού αέρα και ένας κύριος λόγος που την δημιουργεί είναι τα οχήματα. Ένας από τους τομείς που χρησιμοποιεί πολλούς τύπους οχημάτων είναι και η εφοδιαστική αλυσίδα. Έτσι ένας τρόπος μείωσης αυτής της ρύπανσης είναι η ελαχιστοποίηση της κατανάλωσης των καυσίμων των οχημάτων που χρησιμοποιούνται από τις εταιρείες της εφοδιαστικής αλυσίδας. Παράλληλα, με την αύξηση της τιμής των καυσίμων η μείωση της κατανάλωσης των οχημάτων αποτελεί σημαντικό όφελος για μία εταιρεία καθώς θα συντελέσει σημαντικό παράγοντα στην οικονομική της ανάπτυξη αλλά και σταθερότητα. Στο πρόβλημα που επιλύεται στην παρούσα διπλωματική εργασία η εταιρεία δεν έχει ιδιόκτητο στόλο οχημάτων οπότε νοικιάζει οχήματα ώστε να παραδώσει τα προϊόντα στους πελάτες της. Αφού ολοκληρωθούν οι παραδώσεις τα οχήματα δεν επιστρέφουν στην αποθήκη. Η αντικειμενική συνάρτηση του προβλήματος λαμβάνει υπόψη το φορτίο, την απόσταση και τον ρυθμό κατανάλωσης καυσίμου για τον υπολογισμό της συνολικής κατανάλωσης καυσίμου σε κάθε διαδρομή. Για την επίλυση του προβλήματος θα χρησιμοποιηθεί ο αλγόριθμος Βελτιστοποίησης Αποικίας Μυρμηγκιών (Ant Colony Optimization – ACO) και για την βελτίωση των λύσεων πραγματοποιείται μία διαδικασία τοπικής αναζήτησης, όπου χρησιμοποιούνται οι τελεστές τοπικής αναζήτησης 2-Opt, 1 – 1 Exchange και 1 – 0 Relocate. Το πρόβλημα επιλύεται μέσω αλγορίθμου που κατασκευάστηκε στο περιβάλλον της Matlab.

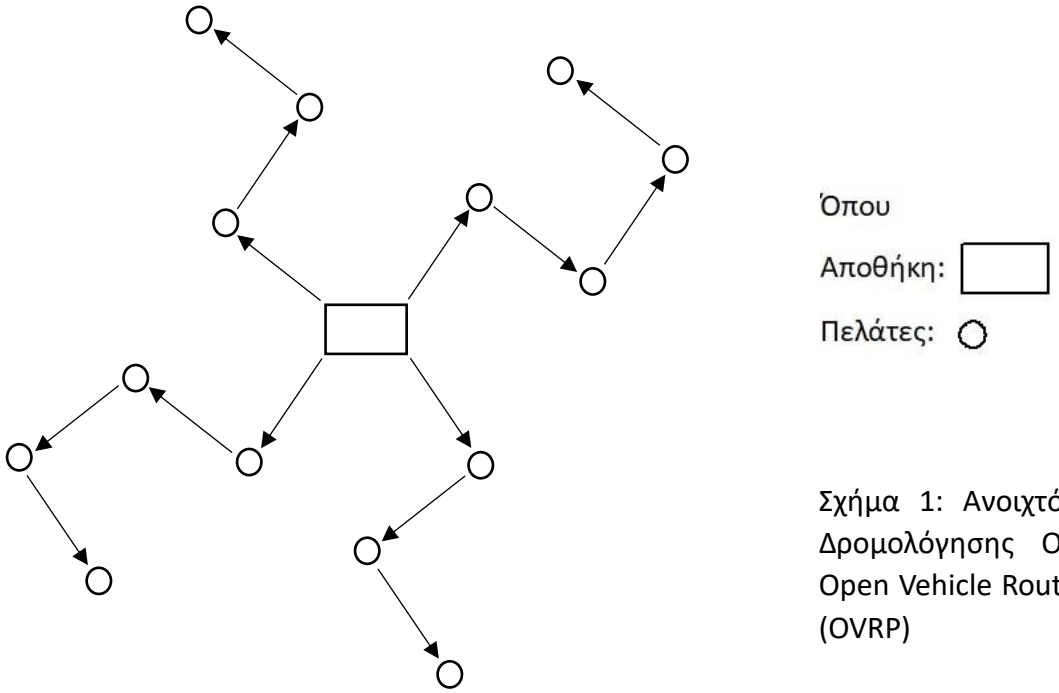
2) Προβλήματα Δρομολόγησης Οχημάτων – Vehicle Routing Problems (VRPs)

2.1) Απλά Προβλήματα Δρομολόγησης Οχημάτων

2.1.1) Ανοιχτό Πρόβλημα Δρομολόγησης Οχημάτων – Open Vehicle Routing Problem (OVRP)

Το Ανοιχτό Πρόβλημα Δρομολόγησης Οχημάτων – Open Vehicle Routing Problem (OVRP) το εμπνεύστηκε ο Schrage το 1981 και είναι ένα πρόβλημα δρομολόγησης οχημάτων όπου τα οχήματα αφού ολοκληρώσουν την εξυπηρέτηση όλων των πελατών δεν επιστρέφουν στην αποθήκη. Σε αυτό το πρόβλημα η εταιρεία δεν έχει ιδιόκτητο στόλο οχημάτων, οπότε συνεργάζεται με εταιρείες που ενοικιάζουν οχήματα ώστε να τα χρησιμοποιήσει για να εξυπηρετήσει τους πελάτες της. Μόλις τα οχήματα ολοκληρώσουν τις διαδρομές που τους έχουν ανατεθεί, δεν επιστρέφουν στην αποθήκη αλλά επιστρέφουν στην εταιρεία από την οποία ενοικιάσθηκαν. Το OVRP έχει ως στόχο να δημιουργηθούν διαδρομές, μία για κάθε φορτηγό, όπου θα βελτιστοποιείται το συνολικό κόστος που καλείται η εταιρεία να πληρώσει για να πραγματοποιήσει την εξυπηρέτηση όλων των πελατών της. Αυτό σημαίνει ότι η εταιρεία πρέπει να λάβει υπόψιν της και το πλήθος των οχημάτων που θα ενοικιάσει, ώστε να ελαχιστοποιήσει το κόστος ενοικίασης.

Παρακάτω απεικονίζεται η μορφή του Ανοιχτού Προβλήματος Δρομολόγησης Οχημάτων – Open Vehicle Routing Problem (OVRP).



Σχήμα 1: Ανοιχτό Πρόβλημα Δρομολόγησης Οχημάτων – Open Vehicle Routing Problem (OVRP)

Παρακάτω παρατίθεται η αντικειμενική συνάρτηση του προβλήματος και οι περιορισμοί του.

$$\min \sum_{i=1}^{n} \sum_{j=1}^{n} \sum_{k=1}^{K} c_{ij} x_{ij}^{k} \ (1)$$

υπό

$$\sum_{i=1}^{n} \sum_{k=1}^{K} x_{ij}^{k} = 1, j = 2, \ldots, n (2)$$

$$\sum_{j=2}^{n} \sum_{k=1}^{K} x_{ij}^{k} = 1, i = 2, \ldots, n (3)$$

$$\sum_{i=1}^{n} x_{ii_1}^{k} - \sum_{j=2}^{n} x_{i_1 j}^{k} = 0, \begin{matrix} k=1, \ldots, K \\ i_1 = 1, \ldots, n \end{matrix} (4)$$

$$\sum_{j=2}^{n} q_i \sum_{i=1}^{n} x_{ij}^{k} \leq Q_k, k = 1, \ldots, K (5)$$

$$\sum_{j=2}^{n} t_i^k \sum_{i=1}^{n} x_{ij}^{k} + \sum_{j=2}^{n} \sum_{i=1}^{n} t_{ij}^{k} x_{ij}^{k} \leq Tm_k, k = 1, \ldots, K \ (6)$$

$$\sum_{j=2}^{n} x_{1j}^{k} \leq 1, \ \ k = 1, \ldots, K \ (7)$$

$$\sum_{i=2}^{n} x_{i1}^{k} = 0, \ k = 1, \ldots, K \ (8)$$

$$X \in S \ (9)$$

$$x_{ij}^{k} = 0 \text{ ή } 1, \text{για όλα τα } i, j, k \ (10)$$

Όπου

ij: το τόξο που συνδέει τον πελάτη i με τον πελάτη j

n: σύνολο κόμβων

k: όχημα

K: σύνολο οχημάτων

c_{ij}: το κόστος του τόξου ij

$$x_{ij}^{k} = \begin{cases} 1, \text{ αν το τόξο ij υπάρχει} \\ 0, \text{ αλλιώς} \end{cases}$$

x_{ij}^{k}: η περίπτωση να υπάρχει το τόξο ij και να το επισκέπτεται το όχημα k

q_i: η ζήτηση του πελάτη i

Q_k: η συνολική χωρητικότητα του οχήματος k

t_i^k: ο χρόνος που χρειάζεται το όχημα k για να εξυπηρετήσει τον πελάτη i

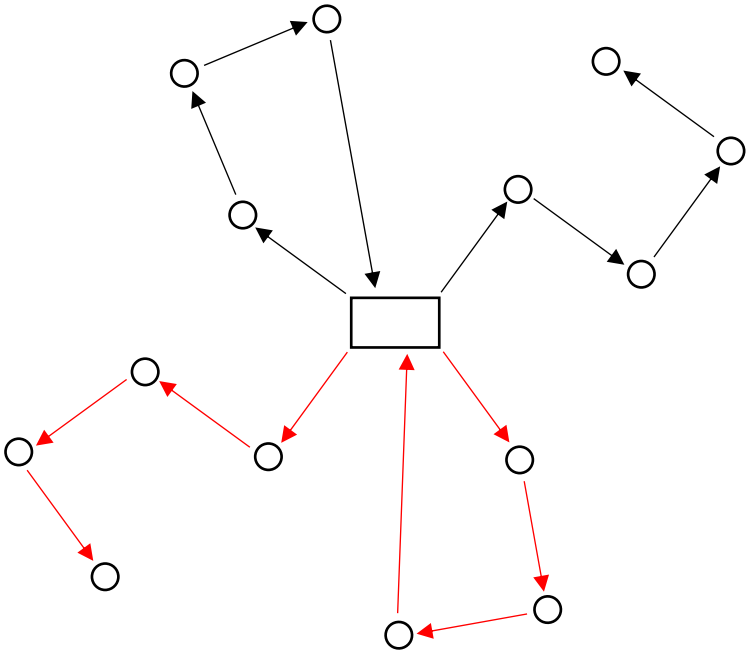
t_{ij}^k: ο χρόνος που χρειάζεται το όχημα k για να επισκεφθεί το τόξο ij

Tm_k: ο μέγιστος χρόνος που μπορεί το όχημα k να βρίσκεται μέσα σε μία διαδρομή

Η σχέση (1) είναι η αντικειμενική συνάρτηση του προβλήματος όπου ελαχιστοποιεί την συνολική απόσταση. Οι περιορισμοί (2) και (3) περιγράφουν ότι κάθε πελάτης, δηλαδή κάθε κόμβος, εξυπηρετείται από ένα όχημα μόνο. Ο περιορισμός (4) δείχνει ότι ένα όχημα όταν μπει σε έναν κόμβο πρέπει οπωσδήποτε να βγει από αυτόν. Οι περιορισμοί (5) και (6) εκφράζουν τους περιορισμούς που έχει κάθε όχημα στην χωρητικότητα και στον χρόνο παραμονής του στην διαδρομή αντίστοιχα. Οι περιορισμοί (7) και (8) παρουσιάζουν ότι δεν παραβιάζεται η διαθεσιμότητα των οχημάτων.

2.1.2) Ανοιχτό Κλειστό Πρόβλημα Δρομολόγησης Οχημάτων – Close Open Vehicle Routing Problem

Το συγκεκριμένο πρόβλημα δρομολόγησης οχημάτων αποτελεί μία παραλλαγή του OVRP, που παρουσιάστηκε παραπάνω, με την διαφορά ότι σε αυτήν την περίπτωση υπάρχει η πιθανότητα κάποιο όχημα να επιστρέψει στην αποθήκη. Πιο συγκεκριμένα, τα οχήματα στην περίπτωση που είναι νοικιασμένα μπορούν να επιστρέψουν στην αποθήκη για να φορτώσουν ξανά προϊόντα προς παράδοση από την στιγμή που δεν έχει εξαντληθεί ο μέγιστος χρόνος χρήσης τους (Σχήμα 2). Άλλη περίπτωση είναι η εταιρεία να έχει ιδιόκτητο στόλο οχημάτων αλλά να μην επαρκεί για την εξυπηρέτηση όλων των πελατών της, οπότε νοικιάζει κάποια οχήματα (Σχήμα 3). Μία ακόμη περίπτωση είναι ο συνδυασμός των δύο προηγούμενων, δηλαδή η εταιρεία να έχει ιδιόκτητο στόλο οχημάτων αλλά να μην μπορεί να εξυπηρετήσει όλους του πελάτες της, οπότε νοικιάζει οχήματα τα οποία αν δεν έχουν εξαντλήσει τον μέγιστο χρόνο χρήσης τους, επιστρέφουν στην αποθήκη για να φορτώσουν ξανά και άλλα προϊόντα (Σχήμα 4).



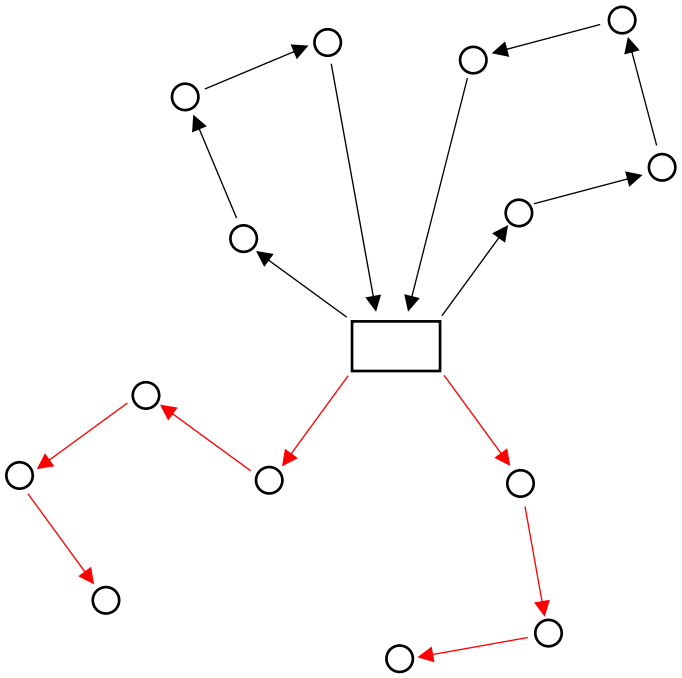
Σχήμα 2: Ανοιχτό Κλειστό Πρόβλημα Δρομολόγησης Οχημάτων – Close Open Vehicle Routing Problem

Όπου

Αποθήκη: 

Πελάτες: ○

Οι διαδρομές με το μαύρο χρώμα πραγματοποιούνται από το 1ο νοικιασμένο όχημα, ενώ οι διαδρομές με το κόκκινο χρώμα πραγματοποιούνται από το 2ο νοικιασμένο όχημα.



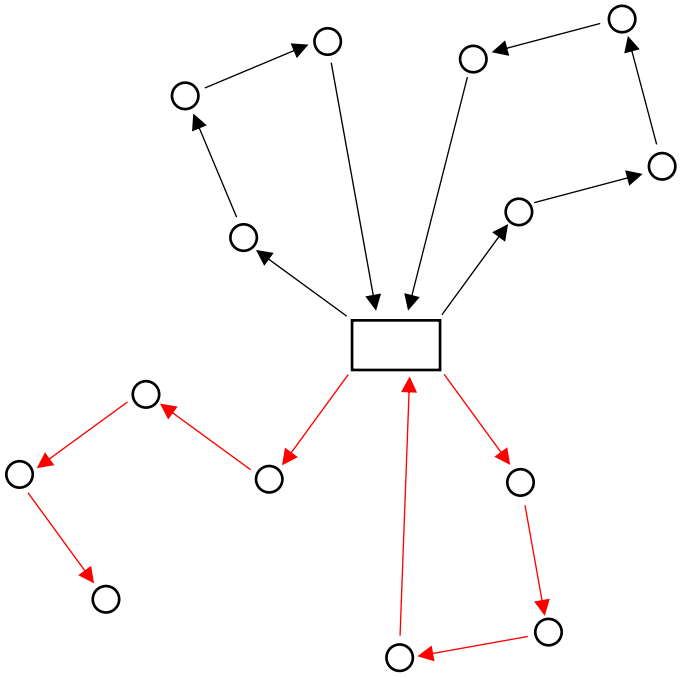
Σχήμα 3: Ανοιχτό Κλειστό Πρόβλημα Δρομολόγησης Οχημάτων – Close Open Vehicle Routing Problem

Όπου

Αποθήκη: 

Πελάτες:

Οι διαδρομές με το μαύρο χρώμα πραγματοποιούνται από το 1ο όχημα που είναι ιδιόκτητο, ενώ οι διαδρομές με το κόκκινο χρώμα πραγματοποιούνται από το 2ο όχημα και το 3ο όχημα που είναι νοικιασμένα.



Σχήμα 4: Ανοιχτό Κλειστό Πρόβλημα Δρομολόγησης Οχημάτων – Close Open Vehicle Routing Problem

Όπου

Αποθήκη: 

Πελάτες: ○

Οι διαδρομές με το μαύρο χρώμα πραγματοποιούνται από το 1ο όχημα που είναι ιδιόκτητο, ενώ οι διαδρομές με το κόκκινο χρώμα πραγματοποιούνται από το 2ο όχημα που είναι νοικιασμένο.

Η αντικειμενική συνάρτηση του προβλήματος, σχέση (11), παρουσιάζεται παρακάτω και σκοπός της είναι να ελαχιστοποιεί το συνολικό κόστος. Δηλαδή να λαμβάνει υπόψιν πόσα οχήματα θα πρέπει να ενοικιαστούν και πως θα χρησιμοποιηθούν σε συνδυασμό με τον ιδιόκτητο στόλο οχημάτων, ώστε να ελαχιστοποιηθεί το συνολικό κόστος των οχημάτων, καθώς και την ελαχιστοποίηση του συνολικού κόστους των διαδρομών.

$$\min \sum_{k=K} F_k \sum_{i \in N} x_{0i}^k + \sum_{k \in K} \sum_{i \in V} \sum_{j \in V} c_{ij} x_{ij}^k - \sum_{i \in N} c_{i0} x_{i0}^1 \ (11)$$

υπό

$$\sum_{i \in V,\ i \neq j} \sum_{k \in V} x_{ij}^k = 1,\ \forall j \in N \ (12)$$

$$\sum_{i \in V,\ i \neq j} x_{ij}^k = \sum_{i \in V,\ i \neq j} x_{ji}^k,\ \forall j \in V, k \in K \ (13)$$

$$u_i + q_j - \left(1 - x_{ij}^k\right) M \leq u_j,\ \forall i, j \in N, k \in K, i \neq j \ (14)$$

$$q_i \leq u_i \leq \sum_{k \in K} \sum_{j \in V} x_{ji}^k Q_k,\ \forall i \in N \ (15)$$

$$h_0 = 0 \ (16)$$

$$h_i + c_{ij} - \left(1 - x_{ij}^k\right) M \leq h_j,\ \forall i \in V, j \in N, k \in K, i \neq j \ (17)$$

$$h_i + c_{i0} - \left(1 - x_{i0}^0\right) M \leq H_0,\ \forall i \in N \ (18)$$

$$0 \leq h_i \leq \sum_{k \in K} \sum_{j \in V} x_{ji}^k H_k,\ \forall i \in N \ (19)$$

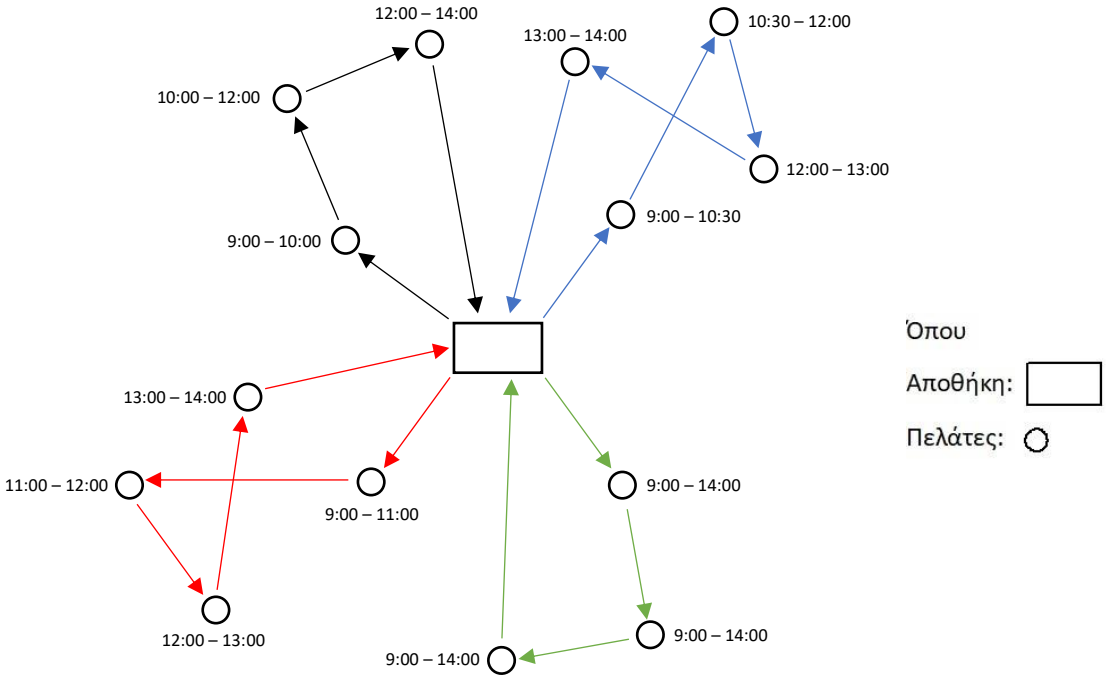
$$\sum_{i \in N} x_{0i}^0 \leq Nu \ (20)$$

$$x_{ij}^k \in \{0, 1\},\ \forall i \in V, j \in V, i \neq j \ (21)$$

Η σχέση (12) είναι ο περιορισμός που δείχνει ότι όλοι οι πελάτες θα εξυπηρετηθούν από μόνο ένα όχημα. Ο επόμενος περιορισμός, σχέση (13), εκφράζει το γεγονός ότι όταν ένα όχημα φτάσει σε έναν πελάτη, αφού τον εξυπηρετήσει, στην συνέχεια θα φύγει από αυτόν. Οι σχέσεις (14) και (15) είναι οι περιορισμοί που εξαλείφουν τους υπόκυκλους και ταυτόχρονα είναι οι περιορισμοί χωρητικότητας του οχήματος. Οι σχέσεις (16), (17), (18) και (19) αποτελούν τους περιορισμούς που δεν επιτρέπουν σε μία διαδρομή να χρειάζεται περισσότερο χρόνο από εκείνον που το όχημα μπορεί να προσφέρει, με το M να είναι ένας μεγάλος θετικός αριθμός. Η σχέση (20) είναι ο περιορισμός που δηλώνει ότι ο αριθμός των ιδιόκτητων οχημάτων πρέπει να είναι μικρότερος ή ίσος από το Nu, όπου Nu είναι ο αριθμός των ιδιόκτητων οχημάτων.

2.1.3) Πρόβλημα Δρομολόγησης Οχημάτων για την Εξυπηρέτηση Πελατών μέσα σε Δεδομένα Χρονικά Περιθώρια – Vehicle Routing Problem with Time Windows

Στο πρόβλημα δρομολόγησης οχημάτων μέσα σε Δεδομένα Χρονικά Περιθώρια η εξυπηρέτηση των πελατών πραγματοποιείται σε συγκεκριμένο χρονικό διάστημα, το οποίο το καθορίζει ο πελάτης και ονομάζεται χρονικό παράθυρο. Οι εμπνευστές αυτού του προβλήματος είναι οι Lenstra και Kan που το παρουσίασαν το 1981. Τα οχήματα, που είναι περιορισμένης χωρητικότητας, φεύγουν από την αποθήκη και επιστρέφουν πάλι σε εκείνη. Στόχος του προβλήματος είναι η εξυπηρέτηση των πελατών μέσα στα χρονικά περιθώρια που έχουν οριστεί, ελαχιστοποιώντας το μήκος των διαδρομών που θα πραγματοποιήσουν τα φορτηγά και χωρίς να παραβιάζεται ο περιορισμός της χωρητικότητας τους. Τα χρονικά παράθυρα που αναφέρθηκαν παραπάνω χωρίζονται σε δύο κατηγορίες, στα χαλαρά και στα σκληρά χρονικά παράθυρα και είναι το χρονικό διάστημα $[\alpha_i, \beta_i]$ εξυπηρέτησης του πελάτη. Στα χαλαρά χρονικά παράθυρα η εξυπηρέτηση του πελάτη μπορεί να ξεκινήσει ακόμα και όταν το όχημα φτάσει σε εκείνον σε κάποια χρονική στιγμή που βρίσκεται εκτός του χρονικού παραθύρου με κάποια ποινή. Αντίθετα, στα σκληρά χρονικά παράθυρα το όχημα δεν πρέπει να φτάσει μετά από τον αργότερο χρόνο εξυπηρέτησης και στην περίπτωση που φτάσει νωρίτερα θα πρέπει να περιμένει, σύμφωνα με το χρονικό παράθυρο που έχει οριστεί, για να ξεκινήσει η εξυπηρέτηση.



Σχήμα 5: Πρόβλημα Δρομολόγησης Οχημάτων για την Εξυπηρέτηση Πελατών μέσα σε Δεδομένα Χρονικά Περιθώρια – Vehicle Routing Problem with Time Windows

Στο σχήμα 5 η διαδρομή με το μαύρο χρώμα πραγματοποιείται από το 1ο όχημα, η διαδρομή με το μπλε χρώμα από το 2ο όχημα, η διαδρομή με το πράσινο χρώμα από το 3ο όχημα και η διαδρομή με το κόκκινο χρώμα από το 4ο όχημα. Δίπλα από κάθε πελάτη απεικονίζεται το χρονικό παράθυρο το οποίο έχει θέσει και το κάθε όχημα πραγματοποιεί την βέλτιστη διαδρομή με βάση αυτά τα χρονικά παράθυρα.

Η αντικειμενική συνάρτηση και οι περιορισμοί του προβλήματος παρουσιάζονται παρακάτω.

$$\min \sum_{i,\,j} c_{ij} \sum_{k} x_{ijk} \quad (22)$$

υπό

$$\sum_{k\in K} \sum_{j\in V} x_{ijk} = 1 \,,\ \forall i \in N \quad (23)$$

$$\sum_{i\in V-\{0\}} x_{0jk} = 1 \,,\ \forall j \in N,\ k \in K \quad (24)$$

$$\sum_{i\in V-\{j\}} x_{ijk} - \sum_{i\in V-\{j\}} x_{jik} = 0 \,,\ \forall j \in N,\ k \in K \quad (25)$$

$$\sum_{i\in V-\{n+1\}} x_{in+1k} = 1 \,,\ \forall k \in K \quad (26)$$

$$x_{ijk}(w_{ik} + s_i + t_{ij} - w_{jk}) \leq 0,\ \forall k \in K (i,j) \in A \quad (27)$$

$$\alpha_i \sum_{j\in V} x_{ijk} \leq w_{ik} \leq \beta_i \sum_{j\in V} x_{ijk} \,,\ \forall i \in N,\ k \in K \quad (28)$$

$$E \leq w_{ik} \leq L,\ \forall i \in (0,n+1),\ k \in K \quad (29)$$

$$\sum_{i\in N} d_i \sum_{j\in V} x_{ijk} \leq C \,,\ \forall k \in K \quad (30)$$

$$x_{ijk} \geq 0,\ \forall k \in K,\ (i,j) \in A \quad (31)$$

$$x_{ijk} \in \{0, 1\},\ \forall k \in K,\ (i,j) \in A \quad (32)$$

Η αντικειμενική συνάρτηση, σχέση (22), αφορά το συνολικό κόστος. Η σχέση (23) εκφράζει τον περιορισμό ότι κάθε πελάτης εξυπηρετείται από μόνο ένα όχημα. Οι επόμενοι τρεις περιορισμοί, σχέσεις (24), (25) και (26), αναφέρονται στην διαδρομή που εκτελεί το όχημα k. Οι περιορισμοί (27), (29) και (30) αφορούν τους χρονικούς περιορισμούς και τους περιορισμούς της χωρητικότητας των οχημάτων. Ο

περιορισμός (28) αναφέρεται στην χρονική μεταβλητή w_{ik}, για δεδομένο όχημα k, να ισούται με το 0 σε περίπτωση που το όχημα δεν εξυπηρετεί τους πελάτες i και j στην διαδρομή του.

2.2) Σύνθετα Προβλήματα Δρομολόγησης Οχημάτων

2.2.1) Πρόβλημα Δρομολόγησης Οχημάτων με Ελαχιστοποίηση της Κατανάλωσης Καυσίμων – Fuel Consumption Vehicle Routing Problem (FCVRP)

Το Πρόβλημα Δρομολόγησης Οχημάτων με Ελαχιστοποίηση της Κατανάλωσης Καυσίμων – Fuel Consumption Vehicle Routing Problem (FCVRP) εμπνεύστηκαν οι Xiao Y., Zhao Q., Kaku I., Xu Y. το 2012. Η μοντελοποίηση του προβλήματος πραγματοποιήθηκε χρησιμοποιώντας την διανυόμενη απόσταση, το φορτίο που έχει το όχημα και τον ρυθμό κατανάλωσης καυσίμου – Fuel Consumption Rate (FCR). Το FCR μετριέται σε liters per km (l/km) και υπολογίζεται από την παρακάτω εξίσωση:

$$\mathrm{FCR}(\mathrm{Q}_1)=\mathrm{FCR}_0+\frac{\mathrm{FCR}^*-\mathrm{FCR}_0}{\mathrm{Q}}\mathrm{Q}_1$$

Όπου

FCR_0: η τιμή του ρυθμού κατανάλωσης καυσίμου όταν το όχημα είναι άδειο

FCR^*: η τιμή του ρυθμού κατανάλωσης καυσίμου όταν το όχημα είναι γεμάτο

Q_1: το βάρος του φορτίου το οποίο έχει το όχημα

Q: το μέγιστο βάρος του φορτίου που μπορεί να έχει ένα όχημα, το οποίο ισοδυναμεί με την μέγιστη χωρητικότητα του οχήματος

Για τον υπολογισμό του FCR από τον κόμβο i στον κόμβο j χρησιμοποιείται η εξίσωση:

$$\mathrm{FCR}_{\mathrm{ij}}=\mathrm{FCR}_0+\frac{\mathrm{FCR}^*-\mathrm{FCR}_0}{\mathrm{Q}}\mathrm{y}_{\mathrm{ij}},\ \forall(\mathrm{i},\mathrm{j})\in\mathrm{A}$$

Όπου

y_{ij}: το βάρος του φορτίου που μεταφέρεται από τον κόμβο i στον κόμβο j

A: το σύνολο των τόξων

Για τον υπολογισμό της κατανάλωσης καυσίμου (FC), η οποία μετριέται σε μονάδες όγκου, από τον κόμβο i στον κόμβο j χρησιμοποιείται η παρακάτω εξίσωση:

$$FC_{ij}=FCR_{ij}d_{ij}x_{ij},\ \forall(i,j)\in A$$

Όπου

d_{ij}: η απόσταση από τον κόμβο i μέχρι τον κόμβο j

x_{ij}: η μεταβλητή που περιγράφει αν το τόξο (i, j) βρίσκεται στην διαδρομή και παίρνει την τιμή 1 αν το τόξο υπάρχει στην διαδρομή, αλλιώς παίρνει την τιμή 0

Για τον υπολογισμό του κόστους καυσίμου F_{cost} χρησιμοποιείται η παρακάτω εξίσωση:

$$Fcost_{ij}=c_0FCR_{ij}d_{ij}x_{ij},\ \forall(i,j)\in A$$

Όπου

c_0: το μοναδιαίο κόστος καυσίμου

Η αντικειμενική συνάρτηση και οι περιορισμοί του προβλήματος παρουσιάζονται παρακάτω.

$$\min\sum_{j=1}^{n}Fx_{0j}+\sum_{i=0}^{n}\sum_{j=0}^{n}c_0d_{ij}(FCR_0x_{ij}+\frac{FCR^*-FCR_0}{Q}y_{ij})\ \ (33)$$

υπό

$$\sum_{j=0}^{n}x_{ij}=1\ ,\ i=1,\ \dots,\ n\ (34)$$

$$\sum_{j=0}^{n}x_{ij}-\sum_{j=0}^{n}x_{ji}=0\ ,\ i=0,\ \dots,\ n\ (35)$$

$$\sum_{j=0,\ j\neq i}^{n}y_{ji}-\sum_{j=0,\ j\neq i}^{n}y_{ij}=D_i\ ,\ i=1,\ \dots,\ n\ (36)$$

$$y_{ij}\leq Qx_{ij},\ i,\ j=0,\ \dots,\ n\ (37)$$

$$x_{ij}\in\{0,\ 1\},\ i,\ j=1,\ \dots,\ n\ (38)$$

Η αντικειμενική συνάρτηση, σχέση (33), χωρίζεται σε δύο μέρη και σε δύο αθροίσματα αντίστοιχα. Το πρώτο μέρος της αφορά τα πάγια κόστη του οχήματος και το δεύτερο στο κόστος καυσίμου όλων των οχημάτων. Ο πρώτος περιορισμός, σχέση (34), δηλώνει ότι κάθε πελάτης θα εξυπηρετηθεί από ένα και μόνο όχημα. Η σχέση

(35) περιγράφει ότι όταν ένα όχημα επισκέπτεται κάποιον πελάτη θα πρέπει και να φύγει από αυτόν. Ο επόμενος περιορισμός, σχέση (36), αναφέρει ότι το φορτίο του οχήματος μειώνεται αφού εξυπηρετήσει κάποιον πελάτη και καλύψει την ζήτηση του. Ο ίδιος περιορισμός δεν επιτρέπει την δημιουργία υποκύκλων. Η σχέση (37) απαγορεύει σε ένα όχημα να μεταφέρει φορτίο που είναι μεγαλύτερο από την μέγιστη χωρητικότητα του. Η σχέση (38) δείχνει τις τιμές που μπορούν να πάρουν οι μεταβλητές x_{ij}.

2.2.2) Πρόβλημα Δρομολόγησης Οχημάτων με Στόχο την Αποφυγή της Μόλυνσης – The Pollution Routing Problem (PRP)

Το Πρόβλημα Δρομολόγησης Οχημάτων με Στόχο την Αποφυγή της Μόλυνσης – The Pollution Routing Problem (PRP) βασίζεται στις εκπομπές CO_2 που δημιουργούνται κατά την μετακίνηση των οχημάτων. Το πρόβλημα εμπνεύστηκαν οι Bektas και Laporte το 2011. Προκειμένου να επιτευχθεί ο στόχος του προβλήματος, η αποφυγή της μόλυνσης, θα πρέπει να ελαχιστοποιηθεί η συνάρτηση που περιλαμβάνει το κόστος καυσίμων, τις εκπομπές των ρύπων και το κόστος μίσθωσης των οδηγών. Για να ελαχιστοποιηθεί η προαναφερθείσα συνάρτηση στο πρόβλημα κατά την εξυπηρέτηση των πελατών λαμβάνεται υπόψιν η ταχύτητα των οχημάτων και ο καθορισμός της σε κάθε τμήμα της διαδρομής που ακολουθούν.

Ο ρυθμός καυσίμου (Fuel Rate) υπολογίζεται από την παρακάτω εξίσωση:

$$FR = \frac{\xi(kNV + \frac{P}{\eta})}{\kappa}$$

Όπου

ξ: ο ρυθμός καυσίμου προς τον αέρα

k: ο συντελεστής τριβής της μηχανής

N: η ταχύτητα της μηχανής

V: το εκτόπισμα της μηχανής

η: σταθερά

κ: σταθερά

P: η ισχύς της μηχανής με μονάδα μέτρησης τα kilowatt (kW)

Η ισχύς της μηχανής υπολογίζεται από την εξίσωση:

$$P = \frac{P_{track}}{\eta_{tf}} + P_{acc}$$

Όπου

η_{tf}: η ικανότητα οδήγησης του οχήματος

P_{acc}: η ισχύς της μηχανής συμπεριλαμβανομένων των απωλειών που έχει η μηχανή λόγω της λειτουργίας εξαρτημάτων του οχήματος (π.χ. κλιματισμός)

P_{track}: οι συνολικές απαιτήσεις της ελκτικής δύναμης που υπάρχει στους τροχούς με μονάδα μέτρησης τα kilowatt (kW)

Η εξίσωση της παραμέτρου P_{track} παρουσιάζεται παρακάτω:

$$P_{track} = \frac{(M\tau + Mg\sin\theta + 0.5C_d\rho Av^2 + MgC_r\cos\theta)v}{1000}$$

Όπου

M: το συνολικό βάρος του οχήματος με μονάδα μέτρησης τα kilogram (kg), M= w + f όπου w το βάρος του άδειου οχήματος και f το φορτίο του οχήματος

v: η ταχύτητα του οχήματος με μονάδα μέτρησης τα meter/second (m/s)

τ: η επιτάχυνση του οχήματος με μονάδα μέτρησης τα meter/(second)^2 (m/s^2)

θ: η γωνία του δρόμου

g: η σταθερά της βαρύτητας

C_d: η σταθερά της αεροδυναμικής αντίστασης

C_r: η σταθερά της αντίστασης κύλισης

ρ: η πυκνότητα του αέρα

A: η μετωπική επιφάνεια αέρα του οχήματος

Η κατανάλωση καυσίμου υπολογίζεται από την εξίσωση:

$$F(v) = \frac{kNV\lambda d}{v} + \frac{P\lambda\gamma d}{v}$$

Όπου

λ: σταθερά με $\lambda = \frac{\xi}{\kappa\psi}$, ψ είναι ο συντελεστής μετατροπής του καυσίμου από gr/sec σε litters/sec

γ: σταθερά με $\gamma = \frac{1}{1000\eta_{tf}\eta}$

Η εξίσωση της κατανάλωσης καυσίμου γράφεται και ως εξής:

$$F(v)=\frac{\lambda(kNV+w\gamma\alpha v+\gamma\alpha fv+\beta\gamma v^3)d}{v}$$

Όπου

α: η σταθερά του οχήματος – τόξου με $\alpha=\tau+g\sin\theta+gC_r\cos\theta$

β: σταθερά του οχήματος με $\beta=0.5C_d\rho A$

Από την παραπάνω εξίσωση προέρχεται η αντικειμενική συνάρτηση του προβλήματος, που παρουσιάζεται παρακάτω μαζί με τους περιορισμούς της.

$$\min \sum_{(i,j)\in A} kNV\lambda d_{ij}\sum_{r=1}^{R}\frac{z_{ij}^r}{v^{-r}}+\sum_{(i,j)\in A} w\gamma\lambda\alpha_{ij}d_{ij}x_{ij}+\sum_{(i,j)\in A}\gamma\lambda\alpha_{ij}d_{ij}f_{ij} + \sum_{(i,j)\in A}\beta\gamma\lambda d_{ij}\sum_{r=1}^{R}\frac{z_{ij}^r}{(v^{-r})^2}+\sum_{j\in N_0} f_d s_j \quad (39)$$

υπό

$$\sum_{j\in N} x_{0j}=m \quad (40)$$

$$\sum_{j\in N} x_{ij}=1\,,\ \forall i\in N_0 \quad (41)$$

$$\sum_{i\in N} x_{ij}=1\,,\ \forall j\in N_0 \quad (42)$$

$$\sum_{j\in N} f_{ji}-\sum_{j\in N} f_{ij}=q_i\,,\ \forall i\in N_0 \quad (43)$$

$$q_j x_{ij}\le f_{ij}\le (Q-q_i)x_{ij},\ \forall (i,j)\in A \quad (44)$$

$$y_i-y_j+t_i+\sum_{r\in R} d_{ij}\frac{z_{ij}^r}{v^{-r}}\le K_{ij}(1-x_{ij}),\ \forall i\in N, j\in N_0, i\ne j \quad (45)$$

$$\alpha_i\le y_i\le b_i,\ \forall i\in N_0 \quad (46)$$

$$y_j+t_j-s_j+\sum_{r\in R} d_{j0}\frac{z_{j0}^r}{v^{-r}}\le L(1-x_{j0}),\ \forall j\in N_0 \quad (47)$$

$$\sum_{r=1}^{R} z_{ij}^r=x_{ij}\,,\ \forall (i,j)\in A \quad (48)$$

$$x_{ij} \in \{0, 1\}, \forall (i, j) \in A \quad (49)$$

$$f_{ij} \geq 0, \forall (i, j) \in A \quad (50)$$

$$y_i \geq 0, \forall i \in N_0 \quad (51)$$

$$z_{ij}^r \in \{0, 1\}, \forall (i, j) \in A, r = 1, \ldots, R \quad (52)$$

Όπου

x_{ij}: η τιμή του είναι 1 αν το τόξο (i, j) βρίσκεται στην διαδρομή, αλλιώς είναι 0

f_{ij}: είναι η συνολική ποσότητα ροής στο τόξο (i, j)∈A

y_j: η χρονική στιγμή όπου αρχίζει η εξυπηρέτηση στον κόμβο j

s_j: ο συνολικός χρόνος που το όχημα χρειάζεται για να περάσει από τους κόμβους που πρέπει να εξυπηρετήσει, μαζί με τον κόμβο j, πριν επιστρέψει στην αποθήκη

z_{ij}^r: η τιμή του είναι 1 αν το τόξο (i, j) διασχίζεται με ταχύτητα r, αλλιώς είναι 0

Η σχέση (39) αποτελεί την αντικειμενική συνάρτηση του προβλήματος και χωρίζεται σε πέντε μέρη. Το πρώτο και το δεύτερο μέρος αφορούν το κόστος από το απόβαρο και το φορτίο που έχει το όχημα αντίστοιχα. Ο πέμπτος όρος αφορά το μισθολογικό κόστος των οδηγών. Ο πρώτος περιορισμός, σχέση (40), εγγυάται ότι κάθε όχημα θα φύγει από την αποθήκη. Οι επόμενοι δύο περιορισμοί, σχέση (41) και (42), δηλώνουν ότι ο κάθε πελάτης εξυπηρετείται μόνο μία φορά από κάποιο όχημα. Οι σχέσεις (43) και (44) καθορίζουν την διατήρηση της ροής στα τόξα. Οι σχέσεις (45), (46) και (47) αποτελούν τους χρονικούς περιορισμούς του οχήματος, όπου $K_{ij} = \max\{0, b_i + s_i + d_{ij}/l_{ij} - a_j\}$ και L ένας μεγάλος αριθμός. Η σχέση (48) ορίζει ότι υπάρχει μόνο ένα επίπεδο ταχύτητας για κάθε τόξο και ισχύει ότι $z_{ij}^r = 1$ αν $x_{ij} = 1$.

3) Αλγόριθμοι Βελτιστοποίησης

3.1) Εξελικτικοί Αλγόριθμοι

3.1.1) Γενετικοί Αλγόριθμοι – Genetic Algorithms

Οι Γενετικοί Αλγόριθμοι βοηθούν στην εύρεση μίας καλύτερης λύσης σε ένα πρόβλημα και βασίζονται στην διαδικασία εξέλιξης που υπάρχει στην φύση. Οι αλγόριθμοι αυτοί δημιουργήθηκαν από τον John H. Holland την δεκαετία του 1960. Συγκεκριμένα, κατά την διάρκεια της εξέλιξης δημιουργούνται νέοι αλλά και καλύτεροι πληθυσμοί, διαφορετικών μεταξύ τους ειδών. Οι αλγόριθμοι αυτοί αναζητούν την καλύτερη λύση αντλώντας πληροφορίες από ένα πλήθος λύσεων, τα άτομα (individuals), άρα κάθε άτομο είναι μία υποψήφια λύση. Για τα άτομα του πληθυσμού ισχύει ότι τα χαρακτηριστικά τους αναπαρίστανται από το χρωμόσωμα (chromosome) ή αλλιώς το γονιδίωμα (genome). Οι γενετικοί αλγόριθμοι αποτελούν μια στοχαστική επαναληπτική διαδικασία που κρατάει σταθερό το μέγεθος του πλήθους των λύσεων σε κάθε επανάληψη η οποία ονομάζεται γενιά (generation). Για την δημιουργία μίας νέας λύσης ο αλγόριθμος ταιριάζει δύο λύσεις μεταξύ τους και χρησιμοποιεί δύο τελεστές. Ο ένας τελεστής είναι δυαδικός και ονομάζεται διασταύρωση (crossover). Ο δεύτερος τελεστής είναι μοναδιαίος και ονομάζεται μετάλλαξη (mutation). Έτσι η δημιουργία μίας νέας και καλύτερης λύσης πραγματοποιείται όταν η διασταύρωση παίρνει δύο άτομα τα οποία ονομάζονται γονείς (parents) και ανταλλάσσει τμήματα μεταξύ των γονέων παράγοντας δύο νέα άτομα που ονομάζονται απόγονοι (offspring).

Γενικά, μερικά πλεονεκτήματα των γενετικών αλγορίθμων είναι ότι:

- επιλύουν αποτελεσματικά τα προβλήματα που έχουν συνεχείς αλλά και διακριτές μεταβλητές
- πραγματοποιούν ταυτόχρονα αναζήτηση σε πολύ μεγάλο μέρος του χώρου των λύσεων
- επιλύουν γρήγορα το πρόβλημα χωρίς να επηρεάζονται από το μέγεθος του, δηλαδή το πλήθος των μεταβλητών που έχει το πρόβλημα

3.1.2) Μιμητικοί Αλγόριθμοι – Memetic Algorithms

Οι Μιμητικοί Αλγόριθμοι ή αλλιώς Υβριδικοί Γενετικοί Αλγόριθμοι είναι συνδυασμός ενός γενετικού αλγορίθμου και ενός αλγορίθμου τοπικής αναζήτησης. Ο εμπνευστής των μιμητικών αλγορίθμων είναι ο Pablo Moscato και τους παρουσίασε το 1989. Οι αλγόριθμοι αυτοί προσφέρουν αποτελέσματα υψηλής ποιότητας, καθώς αποτελούνται από έναν γενετικό αλγόριθμο που πραγματοποιεί μία μέθοδο ολικής

αναζήτησης για την εύρεση της καλύτερης λύσης και από έναν αλγόριθμο τοπικής αναζήτησης που με την διαδικασία της τοπικής αναζήτησης βελτιστοποιεί την λύση. Αυτό που καταφέρνουν οι μιμητικοί αλγόριθμοι είναι να αυξήσουν τις ικανότητες της διασποράς της αναζήτησης σε ολόκληρο τον χώρο των λύσεων (exploration), αλλά και να ψάξει σε συγκεκριμένα σημεία στον χώρο αναζήτησης (exploitation) ώστε να πετύχει εμβάθυνση των λύσεων. Ένας τρόπος να εφαρμοστεί ένας μιμητικός αλγόριθμος είναι να χρησιμοποιηθεί ο γενετικός αλγόριθμος μέχρι να βρει μία αρκετά καλή λύση και μετά να εφαρμοστεί σε αυτή ο αλγόριθμος τοπικής αναζήτησης ώστε να βελτιστοποιηθεί. Γενικά ισχύει ότι ένας μιμητικός αλγόριθμος πάντα αποτελείται από ένα γενετικό ή εξελικτικό αλγόριθμο καθώς και έναν άλλο αλγόριθμο ή μία άλλη τεχνική, που βοηθάει τον γενετικό αλγόριθμο να βελτιστοποιήσει την λύση.

3.1.3) Αλγόριθμος της Διαφορικής Εξέλιξης – Differential Evolution

Ο Αλγόριθμος της Διαφορικής Εξέλιξης είναι ένας στοχαστικός αλγόριθμος και εμπνευστές του είναι οι Storm και Price στα μέσα της δεκαετίας του 1990. Βασικό χαρακτηριστικό αυτού του αλγορίθμου είναι το γεγονός ότι εστιάζει στις αποστάσεις που υπάρχουν μεταξύ των μελών του πληθυσμού, καθώς και στις διαφορετικές κατευθύνσεις που ένα μέλος του πληθυσμού έχει την δυνατότητα να ακολουθήσει. Για να γίνει η εφαρμογή του συγκεκριμένου αλγορίθμου πρέπει οι λύσεις να αναπαρασταθούν σε μορφή πραγματικού αριθμού και αυτό συμβαίνει για να μπορέσουν να εφαρμοστούν οι τελεστές μετάλλαξης. Με την χρήση των τελεστών μετάλλαξης παράγεται ένα δοκιμαστικό διάνυσμα για κάθε μέλος του πληθυσμού, το οποίο χρησιμοποιείται με ένα τελεστή διασταύρωσης για να δημιουργηθεί ένας απόγονος.

Το δοκιμαστικό διάνυσμα $u_i(t)$ για κάθε γονέα $x_i(t)$ αποτελείται από το διάνυσμα στόχου $x_{i1}(t)$, που είναι κάποιο άλλο μέλος του πληθυσμού, δύο άλλα μέλη του πληθυσμού $x_{i2}(t)$ και $x_{i3}(t)$ που επιλέγονται τυχαία και ισχύει ότι $i \neq i_1 \neq i_2 \neq i_3$. Το δοκιμαστικό διάνυσμα υπολογίζεται από την παρακάτω εξίσωση:

$$u_i(t) = x_{i_1}(t) + \beta\left(x_{i_2}(t) - x_{i_3}(t)\right)$$

Όπου $\beta \in (0, \infty)$ είναι ένας παράγοντας κανονικοποίησης.

Το διάνυσμα $x_{i1}(t)$ υπολογίζεται με δύο τρόπους, ο πρώτος είναι να επιλεγεί ένα τυχαίο μέλος του πληθυσμού και ο δεύτερος είναι να επιλεγεί το καλύτερο μέλος του πληθυσμού. Τα διανύσματα $x_{i2}(t)$ και $x_{i3}(t)$ επιλέγονται τυχαία.

Ο τελεστής μετάλλαξης έχει και μερικές παραλλαγές οι οποίες παρουσιάζονται παρακάτω.

Αρχικά η κάθε παραλλαγή συμβολίζεται με την μορφή DE/x/y/z. Το DE συμβολίζει ότι χρησιμοποιείται ο αλγόριθμος της διαφορικής εξέλιξης, το x περιγράφει τον τρόπο με τον οποίο επιλέχθηκε το διάνυσμα στόχος, το y αφορά το πλήθος των διαφορών των διανυσμάτων $x_{i2}(t)$ και $x_{i3}(t)$ και το z παρουσιάζει ποιος τελεστής διασταύρωσης χρησιμοποιείται.

- DE/best/1/z: Σε αυτήν την παραλλαγή του τελεστή μετάλλαξης το διάνυσμα στόχος είναι το καλύτερο μέλος του πληθυσμού και η εξίσωση του δοκιμαστικού διανύσματος είναι η εξής:

$$u_i(t) = x_{opt}(t) + \beta\left(x_{i_2}(t) - x_{i_3}(t)\right)$$

- DE/rand/n_u/z: Στην περίπτωση αυτή ο τελεστής μετάλλαξης έχει ως διάνυσμα στόχο ένα τυχαίο μέλος του πληθυσμού και το πλήθος των διαφορών των διανυσμάτων $x_{i2}(t)$ και $x_{i3}(t)$ ισούται με n_u. Η εξίσωση του δοκιμαστικού διανύσματος παρουσιάζεται παρακάτω:

$$u_i(t) = x_{i_1}(t) + \beta\sum_{l=1}^{n_u}\left(x_{i_2,l}(t) - x_{i_3,l}(t)\right)$$

- DE/best/n_u/z: Η εξίσωση του δοκιμαστικού διανύσματος σε αυτήν την παραλλαγή είναι η εξής:

$$u_i(t) = x_{opt}(t) + \beta\sum_{l=1}^{n_u}\left(x_{i_2,l}(t) - x_{i_3,l}(t)\right)$$

- DE/rand – to – best/n_u/z: Σε αυτήν την παραλλαγή το δοκιμαστικό διάνυσμα αποτελείται από ένα συνδυασμό μεταξύ ενός τυχαίου μέλους του πληθυσμού και του καλύτερου μέλους του πληθυσμού. Η εξίσωση του δοκιμαστικού διανύσματος είναι η παρακάτω:

$$u_i(t) = \gamma x_{opt}(t) + (1-\gamma)x_{i_1}(t) + \beta\sum_{l=1}^{n_u}\left(x_{i_2,l}(t) - x_{i_3,l}(t)\right)$$

  Όπου γ η παράμετρος που αναφέρεται στην επιρροή του καλύτερου και του τυχαίου διανύσματος στόχου.

- DE/current – to – best/1+n_u/z: Σε αυτήν την περίπτωση η εξίσωση του δοκιμαστικού διανύσματος είναι η παρακάτω:

$$u_i(t)=x_{i_1}(t)+\beta\left(x_{opt}(t)-x_{i_1}(t)\right)+\beta\sum_{l=1}^{n_u}\left(x_{i_{2,l}}(t)-x_{i_{3,l}}(t)\right)$$

3.2) Αλγόριθμοι Εμπνευσμένοι από τη Φύση

3.2.1) Αλγόριθμος Βελτιστοποίησης Αποικίας Μυρμηγκιών – Ant Colony Optimization (ACO)

Ο Αλγόριθμος Βελτιστοποίησης Αποικίας Μυρμηγκιών μιμείται την διαδικασία που πραγματοποιούν τα μυρμήγκια για να βρουν τροφή και οι εμπνευστές του είναι οι Dorigo, Maniezzo και Colorni, οι οποίοι τον πρωτοπαρουσίασαν το 1996. Πιο συγκεκριμένα, τα μυρμήγκια αρχίζουν να αναζητούν την τροφή τους τυχαία και σκοπός τους είναι να βρουν τη συντομότερη διαδρομή από την φωλιά τους προς την πηγή της τροφής τους και το αντίστροφο. Καθώς το κάθε ένα από τα μυρμήγκια χαράσσει το μονοπάτι του αφήνει πίσω του μία ποσότητα φερομόνης. Η ποσότητα της φερομόνης εξαρτάται από την ποιότητα και την ποσότητα της τροφής, αλλά και την απόσταση που βρέθηκε η τροφή. Αυτή η ουσία είναι ο λόγος που τα επόμενα μυρμήγκια υπάρχει περίπτωση να ακολουθήσουν το συγκεκριμένο μονοπάτι. Τα επόμενα μυρμήγκια αφήνουν κι εκείνα από μία ποσότητα φερομόνης στα μονοπάτια που επισκέπτονται. Έτσι όσο καλύτερα είναι τα τρία προαναφερθέντα μεγέθη τόσο περισσότερη ποσότητα φερομόνης θα συγκεντρωθεί σε αυτή την διαδρομή και τόσο περισσότερα μυρμήγκια θα ακολουθήσουν το συγκεκριμένο μονοπάτι. Με την πάροδο του χρόνου η ποσότητα της φερομόνης στα μονοπάτια τα οποία δεν αποτελούν μία καλή διαδρομή και δεν τα επισκέπτονται τα μυρμήγκια μειώνεται, μέχρι που εξαφανίζεται τελείως. Το αποτέλεσμα είναι όλα τα μυρμήγκια να ακολουθούν την ίδια διαδρομή, εκείνη με την περισσότερη ποσότητα φερομόνης, η οποία αποτελεί και την βέλτιστη λύση.

Να σημειωθεί ότι κάθε μυρμήγκι αποτελεί μία λύση και ο αλγόριθμος εκτελείται σε επαναλήψεις, οπότε σε κάθε επανάληψη το κάθε μυρμήγκι ξεκινά να δημιουργεί την δική του διαδρομή με βάση τις διαδρομές που έχουν δημιουργηθεί από τα προηγούμενα μυρμήγκια. Έτσι αφού ολοκληρωθούν όλες οι επαναλήψεις τα μυρμήγκια θα έχουν κατασκευάσει την βέλτιστη λύση.

Στα προβλήματα δρομολόγησης οχημάτων η φερομόνη συνδέεται με τα τόξα, δηλαδή συμβολίζει την περίπτωση μετά από έναν πελάτη i να εξυπηρετηθεί ένας πελάτης j. Η ευρετική συνάρτηση που χρησιμοποιείται είναι η παρακάτω:

$$n_{ij}=\frac{1}{c_{ij}}$$

Όπου c_{ij} είναι η απόσταση μεταξύ των πελατών i και j.

Η φερομόνη ενός τόξου υπολογίζεται αρχικά από τον παρακάτω τύπο:

$$\tau_{ij}=\frac{m}{TC}$$

Όπου

m: το πλήθος των μυρμηγκιών

TC: το κόστος μίας αρχικής λύσης που υπολογίστηκε με έναν απλούστερο αλγόριθμο

Κάθε μυρμήγκι επιλέγει τον επόμενο πελάτη με βάση ένα πιθανοτικό κανόνα και ο τύπος που χρησιμοποιείται είναι ο παρακάτω:

$$p_{ij}=\frac{[\tau_{ij}]^{\alpha}[n_{ij}]^{\beta}}{\sum_{l=1}^{M}[\tau_{il}]^{\alpha}[n_{il}]^{\beta}}$$

Όπου

M: το σύνολο των πελατών

α: η παράμετρος που καθορίζει πόσο επηρεάζει η φερομόνη που βρίσκεται πάνω στο τόξο στην επιλογή του επόμενου πελάτη

β: η παράμετρος που καθορίζει πόσο επηρεάζει η ευρετική πληροφορία, που έχει υπολογιστεί για το κάθε τόξο από την ευρετική συνάρτηση, στην επιλογή του επόμενου πελάτη

Με βάση τον παραπάνω τύπο η πιθανότητα ένα τόξο να επιλεγεί αυξάνεται όταν αυξάνεται η φερομόνη και η ευρετική πληροφορία του τόξου.

Η φερομόνη αυξάνεται με βάση τον παρακάτω τύπο:

$$\tau_{ij}=\tau_{ij}+\sum_{k=1}^{m}\Delta\tau_{ij}^{k}$$

Όπου $\Delta\tau_{ij}^{k}$ η ποσότητα φερομόνης που αφήνει κάθε μυρμήγκι k στα τόξα από τα οποία πέρασε. Το $\Delta\tau_{ij}^{k}$ υπολογίζεται από τον τύπο:

$$\Delta\tau_{ij}^{k}=\begin{cases}\frac{1}{C^{k}}, \text{αν το τόξο έχει επιλεγεί από το k}\\ 0, \text{αλλιώς}\end{cases}$$

Όπου C^{k} είναι το κόστος της διαδρομής που έχει δημιουργήσει το μυρμήγκι k.

Η φερομόνη μειώνεται από τον τύπο που παρουσιάζεται παρακάτω:

$$\tau_{ij}=(1-\rho)\tau_{ij}$$

Όπου ρ είναι ο συντελεστής εξάτμισης της φερομόνης και ισχύει ότι $0<\rho<1$.

3.2.2) Αλγόριθμος Βελτιστοποίησης Σμήνους Σωματιδίων – Particle Swarm Optimization (PSO)

Ο Αλγόριθμος Βελτιστοποίησης Σμήνους Σωματιδίων είναι ένας αλγόριθμος ολικής βελτιστοποίησης που προσομοιώνει την κοινωνική συμπεριφορά ορισμένων οργανισμών. Οι εμπνευστές του αλγορίθμου είναι οι Kennedy και Eberhart, οι οποίοι των παρουσίασαν το 1995. Χαρακτηριστικά παραδείγματα κοινωνικών συμπεριφορών είναι τα πουλιά που οργανώνονται και πετούν σε μορφή σμήνους, αλλά και τα ψάρια που κολυμπούν σε ομάδες. Το σμήνος αποτελείται από σωματίδια όπου το κάθε ένα αποτελεί και μία πιθανή λύση στο πρόβλημα. Το κάθε σωματίδιο έχει μία συγκεκριμένη θέση στο χώρο των λύσεων και η κίνηση του πραγματοποιείται με μία συγκεκριμένη ταχύτητα. Οι μεταβολές που πραγματοποιεί ένα σωματίδιο μέσα στο σμήνος εξαρτώνται από το πως λειτουργούν τα γειτονικά του σωματίδια, έτσι ο αλγόριθμος αποτελεί ένα συμβιωτικό συνεργατικό αλγόριθμο.

Αρχικά το κάθε σωματίδιο είναι τυχαία τοποθετημένο στον χώρο και η τροχιά του έχει τρεις πιθανές μεταβολές:

- να κινηθεί σε μία δική του διαδρομή
- να κινηθεί προς την βέλτιστη θέση ($pbest_{ij}$) που εντόπισε μέσα από τις επαναλήψεις
- να κινηθεί προς την θέση που βρίσκεται το βέλτιστο σωματίδιο του πληθυσμού ($gbest_j$).

Η ταχύτητα στον συγκεκριμένο αλγόριθμο συμβολίζει την κατεύθυνση και την απόσταση που το σωματίδιο πρόκειται να κινηθεί.

Οι ταχύτητες και οι θέσεις των σωματιδίων υπολογίζονται με βάση τις παρακάτω εξισώσεις:

$$v_{ij}(t+1)=\chi\left(v_{ij}(t)+c_1 rand_1\left(pbest_{ij}-x_{ij}(t)\right)+c_2 rand_2\left(gbest_j-x_{ij}(t)\right)\right)$$

$$x_{ij}(t+1)=x_{ij}(t)+v_{ij}(t+1)$$

Όπου

$\chi=\frac{2}{\left|2-c-\sqrt{c^2-4c}\right|}$, με $c= c_1 + c_2$ και $c > 4$

t: ο μετρητής των επαναλήψεων

c_1: μεταβλητή επιτάχυνσης

c_2: μεταβλητή επιτάχυνσης

$rand_1$: τυχαία μεταβλητή στο διάστημα [0, 1]

$rand_2$: τυχαία μεταβλητή στο διάστημα [0, 1]

Οι όροι στην εξίσωση της ταχύτητας συμβολίζουν τα εξής:

- Ο πρώτος όρος αντιστοιχεί στην ταχύτητα που είχε το σωματίδιο την αμέσως προηγούμενη χρονική στιγμή.
- Ο δεύτερος όρος, ή αλλιώς γνωστικός όρος (cognitive component), επιστρέφει το σωματίδιο σε προηγούμενες καταστάσεις οι οποίες ήταν ιδιαίτερα αποδοτικές.
- Ο τρίτος όρος, ή αλλιώς (social component), αφορά στον εντοπισμό του βέλτιστου μέλους του σμήνους από το σωματίδιο.

3.2.3) Αλγόριθμος Βελτιστοποίησης Ζευγαρώματος Μελισσών – Honey Bees Mating Optimization (HBMO)

Ο Αλγόριθμος Βελτιστοποίησης Ζευγαρώματος Μελισσών προσομοιώνει την διαδικασία του ζευγαρώματος της βασίλισσας των μελισσών μίας κυψέλης. Ο εμπνευστής του αλγορίθμου είναι ο Abbass και τον παρουσίασε το 2001. Γενικά ισχύει ότι σε μία κυψέλη (hive) υπάρχει μία βασίλισσα (queen), οι κηφήνες (drones) και οι εργάτριες (workers). Η βασίλισσα έχει ως κύριο σκοπό να γεννά αυγά και είναι η μόνη που μπορεί να φάει βασιλικό πολτό (royal jelly). Οι κηφήνες έχουν ως κύριο ρόλο να γονιμοποιήσουν την βασίλισσα. Οι εργάτριες πραγματοποιούν όλες τις δουλειές που είναι απαραίτητες για να μπορεί να λειτουργεί η κυψέλη, με μία από αυτές να είναι η φροντίδα των νεογνών. Το ζευγάρωμα των μελισσών πραγματοποιείται στον αέρα και μακριά από την κυψέλη, με ένα μεγάλο σμήνος από κηφήνες να ακολουθεί την βασίλισσα, η οποία έχει την δυνατότητα να ζευγαρώσει αρκετές φορές, ενώ οι κηφήνες μόνο μία. Μετά από κάθε ζευγάρωμα το σπέρμα του κηφήνα αποθηκεύεται στην σπερματοθήκη της βασίλισσας, ώστε να δημιουργηθεί το γενετικό υλικό της αποικίας και το ζευγάρωμα ολοκληρώνεται με τον θάνατο του κηφήνα. Τα αυγά που γεννάει η βασίλισσα είναι ένας συνδυασμός από το σπέρμα όλων των κηφήνων που είναι αποθηκευμένο στην σπερματοθήκη της.

Παραπάνω έγινε μία γενική περιγραφή σχετικά με την διαδικασία του ζευγαρώματος των μελισσών στην φύση και στην συνέχεια παρουσιάζεται ο αλγόριθμος που βασίζεται σε αυτή τη διαδικασία.

Αρχικά ορίζεται ο πληθυσμός των μελισσών που θα αποτελούν την αρχική κυψέλη, με καλύτερο μέλος να είναι η βασίλισσα και οι κηφήνες να απαρτίζουν τα υπόλοιπα μέλη του αρχικού πληθυσμού. Οι εργάτριες, που δεν παίζουν κάποιο ρόλο στο ζευγάρωμα, λειτουργούν ως μέθοδοι τοπικής αναζήτησης. Οι αρχικές λύσεις επιλέγονται τυχαία και ο γονότυπος που έχει η κάθε μέλισσα αποτελεί μία συγκεκριμένη λύση. Η απόδοση της λύσης χαρακτηρίζεται από μία συνάρτηση ποιότητας $f(x_{ij})$ (fitness function). Το μέγεθος της σπερματοθήκης καθορίζεται από τον

χρήστη, το οποίο αντιπροσωπεύει το μέγιστο αριθμό από ζευγαρώματα που υπάρχει η δυνατότητα να πραγματοποιηθούν σε μία μόνο πτήση. Η πτήση ζευγαρώματος ολοκληρώνεται μόλις γεμίσει η σπερματοθήκη της βασίλισσας. Επίσης, άλλη μία παράμετρος που πρέπει να οριστεί είναι ο αριθμός των νεογνών, δηλαδή των απογόνων, ο οποίος είναι ίσος, μικρότερος αλλά και μεγαλύτερος από το μέγεθος της σπερματοθήκης, αφού το νεογνό παίρνει γονίδια από το γενετικό υλικό όλων των κηφήνων που είναι αποθηκευμένο στην σπερματοθήκη της βασίλισσας. Ο γονότυπος της βασίλισσας διασταυρώνεται με τον γονότυπο των κηφήνων, που είναι αποθηκευμένος στη σπερματοθήκη της, με την χρήση ενός τελεστή διασταύρωσης ο οποίος πρέπει να έχει την δυνατότητα να χρησιμοποιεί περισσότερους από δύο γονείς.

Όσον αφορά την πτήση του ζευγαρώματος είναι ένα σύνολο από μετακινήσεις στον χώρο των λύσεων, με την βασίλισσα να μεταβαίνει από μία κατάσταση σε μία άλλη καθώς έχει μία ταχύτητα και ζευγαρώνει με κάποια πιθανότητα με ένα κηφήνα, που είναι και εκείνος στην ίδια κατάσταση. Η πιθανότητα του ζευγαρώματος υπολογίζεται από τον παρακάτω τύπο:

$$\mathrm{Prob(D)}=\mathrm{e}^{\left[\frac{-\Delta(\mathrm{f})}{\mathrm{Speed(t)}}\right]}$$

Όπου

Prob(D): η πιθανότητα το σπέρμα του κηφήνα D να εισαχθεί στην σπερματοθήκη

Δ(f): η διαφορά στην τιμή της συνάρτησης ποιότητας μεταξύ της βασίλισσας και του κηφήνα D

Speed(t): η ταχύτητα της βασίλισσας την χρονική στιγμή t

Όταν ξεκινάει η πτήση η βασίλισσα, εκτός από κάποια ταχύτητα, έχει και κάποια ενέργεια. Οι αρχικές τιμές της ταχύτητας και της ενέργειας ορίζονται τυχαία. Καθώς η βασίλισσα μετακινείται στον χώρο μειώνεται η ενέργεια και η ταχύτητα της, με αποτέλεσμα να εξασθενούν οι δυνατότητες αναζήτησης που έχει. Όταν η ενέργεια της βρίσκεται στο διάστημα από μηδέν μέχρι και το μέγεθος της σπερματοθήκης η βασίλισσα επιστρέφει στην κυψέλη. Οι τύποι που καθορίζουν την μείωση της ταχύτητας και της ενέργειας της βασίλισσας παρουσιάζονται παρακάτω:

$$\mathrm{Speed(t+1)}=\alpha^{*}\mathrm{Speed(t)}$$

$$\mathrm{energy(t+1)}=\mathrm{energy(t)}-\mathrm{step}$$

Όπου

α: παράγοντας που καθορίζει το ποσό μείωσης της ταχύτητας σε κάθε μετακίνηση, με τιμή στο διάστημα (0, 1)

step: το ποσό μείωσης της ενέργειας σε κάθε μετακίνηση

Σε κάποιες άλλες υλοποιήσεις του αλγορίθμου για να είναι ανάλογη η μείωση της ταχύτητας και της ενέργειας, ο τύπος μείωσης της ενέργειας αποκτά της εξής μορφή:

$$\text{energy}(t+1) = \alpha * \text{energy}(t)$$

Όπως προαναφέρθηκε οι εργάτριες λειτουργούν ως μέθοδοι τοπικής αναζήτησης και βελτιστοποιούν τις λύσεις που προέκυψαν από το ζευγάρωμα. Ανάλογα με τις ικανότητες της κάθε μία εργάτρια αντιστοιχεί σε μία διαφορετική διαδικασία τοπικής αναζήτησης ή σε συνδυασμό από διαδικασίες τοπικής αναζήτησης.

Στην περίπτωση που κάποιο από τα νεογνά αποτελεί καλύτερη λύση από την βασίλισσα τότε το νεογνό την αντικαθιστά και εκείνη αποχωρεί από την κυψέλη. Τα υπόλοιπα νεογνά της κυψέλης μετατρέπονται στους κηφήνες που θα βρίσκονται στην επόμενη πτήση ζευγαρώματος. Ο πληθυσμός των κηφήνων παραμένει σταθερός αφού όταν ένας κηφήνας ζευγαρώσει με την βασίλισσα στη συνέχεια πεθαίνει και έτσι διαγράφεται από τον συνολικό πληθυσμό. Αν ο αριθμός των κηφήνων που διαγράφηκαν είναι μικρότερος από τον αριθμό των απογόνων που δημιουργήθηκαν, τότε ο αριθμός των κηφήνων είναι ο ίδιος με εκείνον του αρχικού πληθυσμού.

3.2.4) Αλγόριθμος Βελτιστοποίησης Ζευγαρώματος Μπάμπουρων – Bumble Bees Mating Optimization (BBMO)

Ο Αλγόριθμος Βελτιστοποίησης Ζευγαρώματος Μπάμπουρων βασίζεται στην διαδικασία του ζευγαρώματος των μπάμπουρων με εμπνευστές τους Marinakis Y., Marinaki M. και Matsatsinis N. που τον παρουσίασαν το 2009. Σε μία αποικία μπάμπουρων υπάρχουν τρία είδη από μπάμπουρες, η βασίλισσα (queen), οι εργάτριες (workers) και οι κηφήνες (drones). Οι βασίλισσες αποτελούν τα μόνα μέλη της κυψέλης που πέφτουν σε χειμερία νάρκη με αποτέλεσμα να επιβιώνουν από την μία χρονιά στην επόμενη. Επίσης, μία βασίλισσα έχει την δυνατότητα να γεννήσει γονιμοποιημένα και μη – γονιμοποιημένα αυγά. Τα γονιμοποιημένα αυγά προκύπτουν από χρωμοσώματα της βασίλισσας και ενός ή περισσοτέρων κηφήνων, από τα οποία δημιουργούνται οι εργάτριες. Τα μη – γονιμοποιημένα αυγά περιέχουν χρωμοσώματα μόνο από την βασίλισσα και δημιουργούν τους κηφήνες. Οι πρώτες εργάτριες που γεννιούνται αντικαθιστούν την βασίλισσα στο κομμάτι της αναζήτησης της τροφής, η οποία μένει στην κυψέλη για να γεννήσει καινούρια αυγά και με μερικές από τις εργάτριες να την βοηθούν στην ανατροφή των νεογνών. Στην συνέχεια από τα αυγά που γεννάει η βασίλισσα αναδεικνύονται νέες βασίλισσες οι οποίες μαζί με τους κηφήνες φεύγουν από την κυψέλη, με αποτέλεσμα η αποικία να εκφυλίζεται. Όταν η βασίλισσα ζευγαρώσει με έναν ή περισσότερους κηφήνες το σπέρμα τους αποθηκεύεται στην σπερματοθήκη της και εκείνη ψάχνει ένα μέρος για να περάσει τον χειμώνα.

Όσον αφορά την διαδικασία του ζευγαρώματος, υπάρχουν τρία είδη. Στο πρώτο είδος ο κηφήνας διαλέγει ένα ψηλό σημείο για να κάτσει, παρατηρώντας τον χώρο και μόλις η βασίλισσα περάσει αρχίζει να την κυνηγά ώστε να ζευγαρώσουν. Στην δεύτερη διαδικασία ο κηφήνας γεμίζει με φερομόνη κάποια συγκεκριμένη διαδρομή, με αποτέλεσμα η βασίλισσα να έλκεται από την φερομόνη και να ακολουθήσει αυτή την διαδρομή. Στην τρίτη διαδικασία ο κηφήνας βρίσκεται έξω από την κυψέλη και περιμένει την βασίλισσα να βγει για να την ακολουθήσει.

Τα παραπάνω αφορούν την διαδικασία του ζευγαρώματος των μπάμπουρων που πραγματοποιείται στην φύση και στην συνέχεια παρουσιάζεται ο αλγόριθμος που προσομοιώνει αυτή την διαδικασία.

Ο κάθε μπάμπουρας αποτελεί μία λύση του προβλήματος και ο αριθμός των μπάμπουρων επιλέγεται τυχαία. Ο μπάμπουρας που η αντικειμενική του συνάρτηση έχει την καλύτερη τιμή γίνεται βασίλισσα, ενώ όσο η διαδικασία βρίσκεται στην φάση της αρχικοποίησης οι υπόλοιποι γίνονται κηφήνες. Μετά η βασίλισσα χρησιμοποιεί την δεύτερη διαδικασία ζευγαρώματος για να επιλέξει τον κηφήνα με τον οποίο θα ζευγαρώσει, ταξινομώντας τις τιμές των αντικειμενικών συναρτήσεων όλων των κηφήνων για να διαλέξει τον κηφήνα που έχει αφήσει πίσω του την περισσότερη φερομόνη. Μετά από το κάθε ζευγάρωμα ο γονότυπος του κηφήνα αποθηκεύεται στην σπερματοθήκη της βασίλισσας και η διαδικασία ολοκληρώνεται όταν συμπληρωθεί ένας συγκεκριμένος αριθμός από ζευγαρώματα. Μόλις συμπληρωθεί αυτός ο αριθμός η βασίλισσα βρίσκει ένα μέρος με σκοπό να δημιουργήσει την κυψέλη της και να περάσει τον χειμώνα, να πέσει σε χειμερία νάρκη και τον επόμενο χρόνο να γεννήσει τα αυγά της. Όλη αυτή η διαδικασία αποτελεί μία επανάληψη του αλγορίθμου. Ο χρήστης ορίζει τον αριθμό των ζευγαρωμάτων που θα πραγματοποιηθούν σε μία επανάληψη. Οι απόγονοι που είναι οι πιο ισχυροί γίνονται οι νέες βασίλισσες ενώ οι υπόλοιποι γίνονται εργάτριες. Ο αριθμός των νέων βασιλισσών είναι ίσος με τον μέγιστο αριθμό βασιλισσών, που αποτελεί μία παράμετρο του αλγορίθμου. Οι νέες βασίλισσες και οι εργάτριες παράγονται από την διασταύρωση του γονότυπου της βασίλισσας με το γονότυπο του κηφήνα, χρησιμοποιώντας έναν τελεστή διασταύρωσης. Οι κηφήνες δημιουργούνται με μετάλλαξη του γονότυπου των παλιών βασιλισσών ή με την μετάλλαξη του γονότυπου των εργατριών.

Η διαδικασία που περιγράφεται παραπάνω πραγματοποιείται με την χρήση της παρακάτω εξίσωσης:

$$nq_{ij} = nq_{ij} + \left(b_{max} - \frac{(b_{max} - b_{min}) * lsi}{lsi_{max}}\right) * (nq_{ij} - q_j) + \frac{1}{M} * \sum_{k=1}^{M} \left(b_{min} - \frac{(b_{min} - b_{max}) * lsi}{lsi_{max}}\right) * (nq_{ij} - w_{kj})$$

Όπου:

nq_{ij}: η λύση της νέας βασίλισσας i

q_j: η λύση της παλιάς βασίλισσας

w_{kj}: η λύση των εργατριών

M: ο αριθμός των εργατριών που επιλέγει η κάθε νέα βασίλισσα για να την θρέψουν, ο αριθμός αυτός είναι διαφορετικός για κάθε βασίλισσα

b_{max}: παράμετρος που ορίζει αν η νέα βασίλισσα θα τραφεί από την παλιά βασίλισσα, από τις εργάτριες ή και από τις δύο, οι τιμές τις παραμέτρου βρίσκονται στο διάστημα (0, 1)

b_{min}: παράμετρος που ορίζει αν η νέα βασίλισσα θα τραφεί από την παλιά βασίλισσα, από τις εργάτριες ή και από τις δύο, οι τιμές τις παραμέτρου βρίσκονται στο διάστημα (0, 1)

lsi: η τρέχουσα επανάληψη της τοπικής αναζήτησης

lsi_{max}: ο μέγιστος αριθμός επαναλήψεων της τοπικής αναζήτησης

3.3) Αλγόριθμοι Τοπικής Αναζήτησης

3.3.1) 2 – Opt

Ο Αλγόριθμος Τοπικής Αναζήτησης 2 – Opt πραγματοποιείται σε κάθε μία διαδρομή ξεχωριστά ενός προβλήματος δρομολόγησης οχημάτων. Ο εμπνευστής του αλγορίθμου είναι ο Croes και τον παρουσίασε το 1958. Συγκεκριμένα η διαδικασία αυτή πραγματοποιεί την διαγραφή δύο τόξων της διαδρομής και στην συνέχεια συνδέει ξανά τους κόμβους τους με διαφορετικό τρόπο. Στόχος του συγκεκριμένου αλγορίθμου είναι να βελτιστοποιήσει μία αρχική λύση, που έχει δημιουργηθεί με κάποιον άλλο αλγόριθμο, βελτιστοποιώντας τις διαδρομές της λύσης αυτής και χωρίς να παραβιάζονται οι περιορισμοί του προβλήματος.

Παρακάτω παρουσιάζεται η διαδικασία 2 – Opt εφαρμοσμένη στο Ανοιχτό Πρόβλημα Δρομολόγησης Οχημάτων (OVRP)

Πριν την διαδικασία:

1	2	3	4	5	6

Μετά την διαδικασία:

1	2	5	4	3	6

3.3.2) 1 – 1 Exchange

Ο Αλγόριθμος Τοπικής Αναζήτησης 1 – 1 Exchange εφαρμόζεται σε δύο ή περισσότερες διαδρομές σε μία λύση ενός προβλήματος δρομολόγησης οχημάτων. Ο εμπνευστής του είναι ο Waters και τον παρουσίασε το 1987. Συγκεκριμένα αυτή η μέθοδος ανταλλάσσει ένα πελάτη μίας διαδρομής με ένα πελάτη μίας άλλης διαφορετικής διαδρομής της λύσης του προβλήματος. Στόχος αυτής της διαδικασίας είναι να βελτιστοποιηθούν οι δύο διαδρομές αλλά και ολόκληρη η λύση του προβλήματος, χωρίς όμως να παραβιάζονται οι περιορισμοί του προβλήματος. Έτσι η ανταλλαγή των πελατών θα πραγματοποιηθεί μόνο όταν ισχύουν τα παραπάνω.

Ένα παράδειγμα της μεθόδου στο Ανοιχτό Πρόβλημα Δρομολόγησης Οχημάτων (OVRP) παρουσιάζεται παρακάτω.

Πριν την ανταλλαγή:

1η Διαδρομή

1	2	3	4	5	6

2η Διαδρομή

1	7	8	9	10

Μετά την ανταλλαγή:

1η Διαδρομή

1	9	3	4	5	6

2η Διαδρομή

1	7	8	2	10

3.3.3) 1 – 0 Relocate

Ο Αλγόριθμος Τοπικής Αναζήτησης 1 – 0 Relocate εφαρμόζεται μεταξύ δύο ή περισσότερων διαδρομών σε ένα πρόβλημα δρομολόγησης οχημάτων, με εμπνευστή τον Waters που τον παρουσίασε το 1987. Η διαδικασία αυτή πραγματοποιεί την αφαίρεση ενός πελάτη από μία διαδρομή και την επανατοποθέτηση του σε μία άλλη διαδρομή της λύσης του προβλήματος. Αυτή η κίνηση εφαρμόζεται μόνο αν βελτιστοποιούνται οι δύο διαδρομές αλλά και γενικότερα η λύση του προβλήματος, με βασική προϋπόθεση να μην παραβιάζονται οι περιορισμοί του προβλήματος.

Παρακάτω παρουσιάζεται ένα παράδειγμα της διαδικασίας 1 – 0 Relocate που εφαρμόζεται στο Ανοιχτό Πρόβλημα Δρομολόγησης Οχημάτων (OVRP).

Πριν την επανατοποθέτηση:

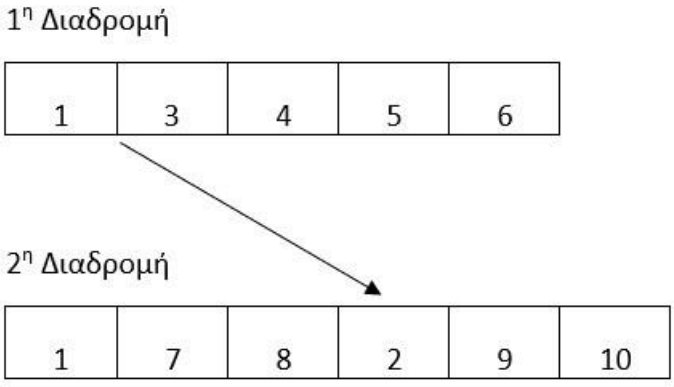
1η Διαδρομή

1	2	3	4	5	6

2η Διαδρομή

1	7	8	9	10

Μετά την επανατοποθέτηση:



4) Προτεινόμενος Αλγόριθμος

Ο αλγόριθμος που επιλύει το πρόβλημα της παρούσας διπλωματικής εργασίας, δηλαδή την παραλλαγή του Ανοιχτού Προβλήματος Δρομολόγησης Οχημάτων (Open Vehicle Routing Problem – OVRP) με στόχο την Ελαχιστοποίηση της Κατανάλωσης Καυσίμων (Fuel Consumption), όπως προαναφέρθηκε και στην εισαγωγή, έχει κατασκευαστεί στο περιβάλλον της Matlab. Επίσης, γίνεται χρήση του Αλγορίθμου Βελτιστοποίησης Αποικίας Μυρμηγκιών ο οποίος έχει παρουσιαστεί σε προηγούμενο κεφάλαιο. Για την βελτίωση των λύσεων που προκύπτουν χρησιμοποιούνται οι τελεστές τοπικής αναζήτησης 2-Opt, 1 – 1 Exchange και 1 – 0 Relocate. Ο αλγόριθμος αποτελείται από το κυρίως πρόγραμμα και από επτά συναρτήσεις.

Το κυρίως πρόγραμμα ονομάζεται main.m και θα παρουσιαστεί παρακάτω. Οι συναρτήσεις είναι οι εξής:

- Dedomena.m
- Cost.m
- Capacity.m
- Reverse_2_opt.m
- Exchange_1_1.m
- Relocate_1_0.m
- Local_Search.m

των οποίων η λειτουργία θα περιγραφεί παρακάτω.

4.1) Συναρτήσεις

- **Dedomena.m**

  Η λειτουργία της συνάρτησης είναι να εισάγει τα δεδομένα από το Excel. Τα δεδομένα αυτά είναι διαφορετικά για κάθε παράδειγμα που πρόκειται να επιλυθεί, έτσι χρησιμοποιείται η εντολή switch/case για να επιλέγει ο χρήστης κατά το κάλεσμα της συνάρτησης στο κυρίως πρόγραμμα κάθε φορά τα δεδομένα του παραδείγματος που επιθυμεί να επιλυθεί.

- **Cost.m**

  Ο σκοπός της συγκεκριμένης συνάρτησης όταν καλείται είναι να υπολογίζει το κόστος της διαδρομής που επιθυμεί ο χρήστης. Το κόστος υπολογίζεται με βάση τον τύπο της αντικειμενικής συνάρτησης του Προβλήματος Δρομολόγησης Οχημάτων με στόχο την Ελαχιστοποίηση της Κατανάλωσης

Καυσίμων – Fuel Consumption Vehicle Routing Problem (FCVRP), που παρουσιάστηκε παραπάνω.

- **Capacity.m**

  Η συνάρτηση αυτή υπολογίζει την ποσότητα των προϊόντων που απαιτεί μία διαδρομή. Την διαδρομή την επιλέγει ο χρήστης και η ποσότητα εξαρτάται από την ζήτηση των πελατών που βρίσκονται στην διαδρομή αυτή.

- **Reverse_2_opt.m**

  Η συγκεκριμένη συνάρτηση πραγματοποιεί την διαδικασία 2-Opt, που παρουσιάστηκε παραπάνω, σε μία διαδρομή ελαχιστοποιώντας το κόστος της. Στον κώδικα της συνάρτησης αυτής χρησιμοποιείται και η συνάρτηση Cost.m όπου υπολογίζει το αρχικό κόστος της διαδρομής και ξανακαλείται μετά από κάθε αλλαγή κάποιου τόξου, ώστε να υπολογίσει το κόστος της νέας διαδρομής και να ελεγχθεί αν η αλλαγή αυτή ελαχιστοποίησε το κόστος της διαδρομής. Σε περίπτωση που το κόστος ελαχιστοποιηθεί η νέα διαδρομή που δημιουργείται αποθηκεύεται και η συνάρτηση τερματίζεται όταν πραγματοποιηθούν όλες οι πιθανές αλλαγές τόξων στην διαδρομή. Η καλύτερη διαδρομή αποθηκεύεται και μαζί με το κόστος της αποτελούν την έξοδο της συνάρτησης.

- **Exchange_1_1.m**

  Η συνάρτηση αυτή πραγματοποιεί την διαδικασία 1 – 1 Exchange η οποία παρουσιάστηκε παραπάνω. Συγκεκριμένα, εφαρμόζεται σε δύο διαδρομές με σκοπό την ελαχιστοποίηση του συνολικού κόστους των δύο διαδρομών. Αφού εφαρμοστεί μία επανάληψη της διαδικασίας καλείται η συνάρτηση Cost.m και η συνάρτηση Capacity.m, για να υπολογιστεί το κόστος και των δύο διαδρομών και η ποσότητα των προϊόντων που υπάρχει και στις δύο διαδρομές. Στην συνέχεια πραγματοποιείται έλεγχος ώστε να διαπιστωθεί αν έχει μειωθεί το συνολικό κόστος και αν δεν παραβιάζεται η συνθήκη του Ανοιχτού Προβλήματος Δρομολόγησης Οχημάτων (Open Vehicle Routing Problem – OVRP) με στόχο την Ελαχιστοποίηση της Κατανάλωσης Καυσίμων (Fuel Consumption) σχετικά με την μέγιστη χωρητικότητα των οχημάτων σε μία διαδρομή. Σε περίπτωση που το συνολικό κόστος είναι μειωμένο και η συνθήκη της μέγιστης χωρητικότητας δεν παραβιάζεται, οι νέες και βελτιστοποιημένες δύο διαδρομές αποθηκεύονται. Η συνάρτηση λειτουργεί έως ότου πραγματοποιηθούν όλες οι πιθανές κινήσεις, αποθηκεύει την καλύτερη και έχει ως έξοδο τις δύο διαδρομές μαζί με το αντίστοιχο κόστος τους.

- **Relocate_1_0.m**

  Η λειτουργία της συνάρτησης είναι να πραγματοποιεί την διαδικασία 1 – 0 Relocate που παρουσιάστηκε παραπάνω. Η εφαρμογή της διαδικασίας πραγματοποιείται μεταξύ δύο διαδρομών. Κατά την διάρκεια μίας επανάληψης, αφού πραγματοποιηθεί η διαδικασία, υπολογίζεται το νέο κόστος της κάθε διαδρομής με την χρήση της συνάρτησης Cost.m, καθώς και η νέα ποσότητα προϊόντων που υπάρχει στην κάθε διαδρομή χρησιμοποιώντας την συνάρτηση Capacity.m. Στην συνέχεια ελέγχεται αν το νέο συνολικό κόστος των δύο διαδρομών είναι μικρότερο σε σύγκριση με το παλιό και αν η ποσότητα των προϊόντων σε κάθε διαδρομή δεν υπερβαίνει την μέγιστη χωρητικότητα του κάθε οχήματος, η οποία ορίζεται με βάση την συνθήκη που έχει το Ανοιχτό Πρόβλημα Δρομολόγησης Οχημάτων (Open Vehicle Routing Problem – OVRP) με στόχο την Ελαχιστοποίηση της Κατανάλωσης Καυσίμων (Fuel Consumption). Εφόσον μειωθεί το συνολικό κόστος και δεν παραβιάζεται η συνθήκη, οι δύο νέες διαδρομές μαζί με το αντίστοιχο κόστος τους αποθηκεύονται. Η συνάρτηση λειτουργεί μέχρι να πραγματοποιηθούν όλες οι πιθανές κινήσεις, αποθηκεύει την καλύτερη και έχει ως έξοδο τις δύο διαδρομές και το αντίστοιχο κόστος τους.

- **Local_Search.m**

  Η συνάρτηση αυτή χρησιμοποιεί τις συναρτήσεις Reverse_2_opt.m, Exchange_1_1.m και Relocate_1_0.m για να ελαχιστοποιήσει το κόστος δύο διαδρομών. Αρχικά καλεί από μία φορά για κάθε διαδρομή την συνάρτηση Reverse_2_opt.m, στην συνέχεια καλεί την Exchange_1_1.m που την χρησιμοποιεί για τις δύο διαδρομές που η κάθε μία έχει εξαχθεί από την συνάρτηση Reverse_2_opt.m και τέλος καλεί την συνάρτηση Relocate_1_0.m για τις δύο διαδρομές που έχουν εξαχθεί από την συνάρτηση Exchange_1_1.m. Η έξοδος της συνάρτησης Local_Search.m είναι οι δύο νέες και βελτιστοποιημένες διαδρομές με το αντίστοιχο κόστος τους.

4.2) Κυρίως Πρόγραμμα

Παρακάτω παρουσιάζεται το κυρίως πρόγραμμα main.m.

main.m

Επιλογή παραδείγματος από τον χρήστη

Εφαρμογή της συνάρτησης Dedomena και εισαγωγή δεδομένων

Αρχικοποίηση μεταβλητών του ACO

for (Επαναλήψεις κώδικα, τις ορίζει ο χρήστης**)**

Αρχικοποίηση μεταβλητών του ACO

αριθμός μυρμηγκιών= … (ορίζονται από τον χρήστη)

α= …

β= …

ρ= …

for (Επαναλήψεις= αριθμός μυρμηγκιών**)**

unvisited= [2: Αριθμός πελατών]

(όπου unvisited είναι οι πελάτες που δεν έχουν εξυπηρετηθεί)

Αρχικοποίηση της διαδρομής

while (μέχρι το unvisited να αδειάσει**)**

Επιλογή πελάτη

If (Έλεγχος αν η ζήτηση του πελάτη δεν παραβιάζει την μέγιστη χωρητικότητα της διαδρομής**)**

Προσθήκη πελάτη στην διαδρομή

Αφαίρεση του πελάτη από το unvisited

else

Προσθήκη της διαδρομής στη λύση

Εφαρμογή της συνάρτησης Cost για τον υπολογισμό του κόστους της διαδρομής

Προσθήκη του κόστους της διαδρομής στο συνολικό κόστος της λύσης

Αρχικοποίηση της νέας διαδρομής

end

end

if (κόστος λύσης ≤ κόστος καλύτερης λύσης της επανάληψης && αριθμός φορτηγών λύσης ≤ αριθμός φορτηγών καλύτερης λύσης της επανάληψης**)**

Αποθήκευση της λύσης ως την καλύτερη λύση της επανάληψης

Αποθήκευση του κόστους της λύσης ως το καλύτερο κόστος της επανάληψης

else if (αριθμός φορτηγών λύσης < αριθμός φορτηγών καλύτερης λύσης της επανάληψης**)**

Αποθήκευση της λύσης ως την καλύτερη λύση της επανάληψης

Αποθήκευση του κόστους της λύσης ως το καλύτερο κόστος της επανάληψης

end

end

end

if (αριθμός φορτηγών = μονός αριθμός**)**

Υπολογισμός κόστους του πρώτου φορτηγού της καλύτερης λύσης της επανάληψης

Εφαρμογή της συνάρτησης Reverse_2_opt στο πρώτο φορτηγό της καλύτερης λύσης της επανάληψης

Υπολογισμός του νέου κόστους της καλύτερης λύσης της επανάληψης

for (τα υπόλοιπα φορτηγά**)**

Υπολογισμός του κόστους του φορτηγού α της καλύτερης λύσης της επανάληψης με την εφαρμογή της συνάρτησης Cost

Υπολογισμός του κόστους του φορτηγού β της καλύτερης λύσης της επανάληψης με την εφαρμογή της συνάρτησης Cost

Εφαρμογή της συνάρτησης Local_Search στα φορτηγά α και β

Υπολογισμός του νέου κόστους της καλύτερης λύσης της επανάληψης

end

elseif (αριθμός φορτηγών = ζυγός αριθμός**)**

for (αριθμός φορτηγών**)**

Υπολογισμός του κόστους του φορτηγού α της καλύτερης λύσης της επανάληψης με την εφαρμογή της συνάρτησης Cost

Υπολογισμός του κόστους του φορτηγού β της καλύτερης λύσης της επανάληψης με την εφαρμογή της συνάρτησης Cost

Εφαρμογή της συνάρτησης Local_Search στα φορτηγά α και β

Υπολογισμός του νέου κόστους της καλύτερης λύσης της επανάληψης

end

end

for (αριθμός φορτηγών**)**

if (κάποιο φορτηγό = άδειο**)**

Αποθήκευση φορτηγού

end

end

if (υπάρχει άδειο φορτηγό**)**

Διαγραφή αποθηκευμένου φορτηγού

end

if (κόστος καλύτερης λύσης της επανάληψης ≤ κόστος καλύτερης λύσης όλου του προγράμματος**)**

Αποθήκευση της καλύτερης λύσης της επανάληψης ως την καλύτερη λύση όλου του προγράμματος

Αποθήκευση του κόστους της καλύτερης λύσης της επανάληψης ως το καλύτερο κόστος όλου του προγράμματος

end

Μείωση της φερομόνης στις διαδρομές

Αύξηση της φερομόνης στις διαδρομές

end

Εκτύπωση της τελικής λύσης

Εκτύπωση της τελικής λύσης στο Excel

5) Αποτελέσματα

Ο παραπάνω κώδικας επίλυσε 7 διαφορετικά παραδείγματα και σε αυτό το κεφάλαιο παρουσιάζονται τα αποτελέσματα. Τα δεδομένα των παραδειγμάτων προέρχονται από το Xiao, Y., Zhao, Q., Kaku, I., & Xu, Y. (2012), Development of a fuel consumption optimization model for the capacitated vehicle routing problem, Computers & operations research, 39(7), 1419-1431. Πιο αναλυτικά, σε όλα τα παραδείγματα το πάγιο κόστος οχήματος έχει οριστεί σε 0, το κόστος καυσίμου ισούται με 1, η τιμή του ρυθμού κατανάλωσης καυσίμου όταν το όχημα είναι άδειο FCR_0= 1 και η τιμή του ρυθμού κατανάλωσης καυσίμου όταν το όχημα είναι γεμάτο FCR^*= 2. Το κάθε παράδειγμα αποτελείται από n κόμβους, όπου ο αριθμός των πελατών ισούται με n – 1 αφού μέσα στους κόμβους βρίσκεται και η αποθήκη. Στο κάθε παράδειγμα υπάρχει συγκεκριμένη ποσότητα προϊόντων που το κάθε όχημα έχει την δυνατότητα να μεταφέρει σε κάθε διαδρομή. Παρακάτω παρουσιάζονται τα δεδομένα του κάθε παραδείγματος.

- **Παράδειγμα 1:**
  Αριθμός κόμβων: 51
  Αριθμός πελατών: 50
  Χωρητικότητα Οχήματος: 160

- **Παράδειγμα 2:**
  Αριθμός κόμβων: 76
  Αριθμός πελατών: 75
  Χωρητικότητα Οχήματος: 140

- **Παράδειγμα 3:**
  Αριθμός κόμβων: 101
  Αριθμός πελατών: 100
  Χωρητικότητα Οχήματος: 200

- **Παράδειγμα 4:**
  Αριθμός κόμβων: 151
  Αριθμός πελατών: 150
  Χωρητικότητα Οχήματος: 200

- **Παράδειγμα 5:**
  Αριθμός κόμβων: 200
  Αριθμός πελατών: 199
  Χωρητικότητα Οχήματος: 200

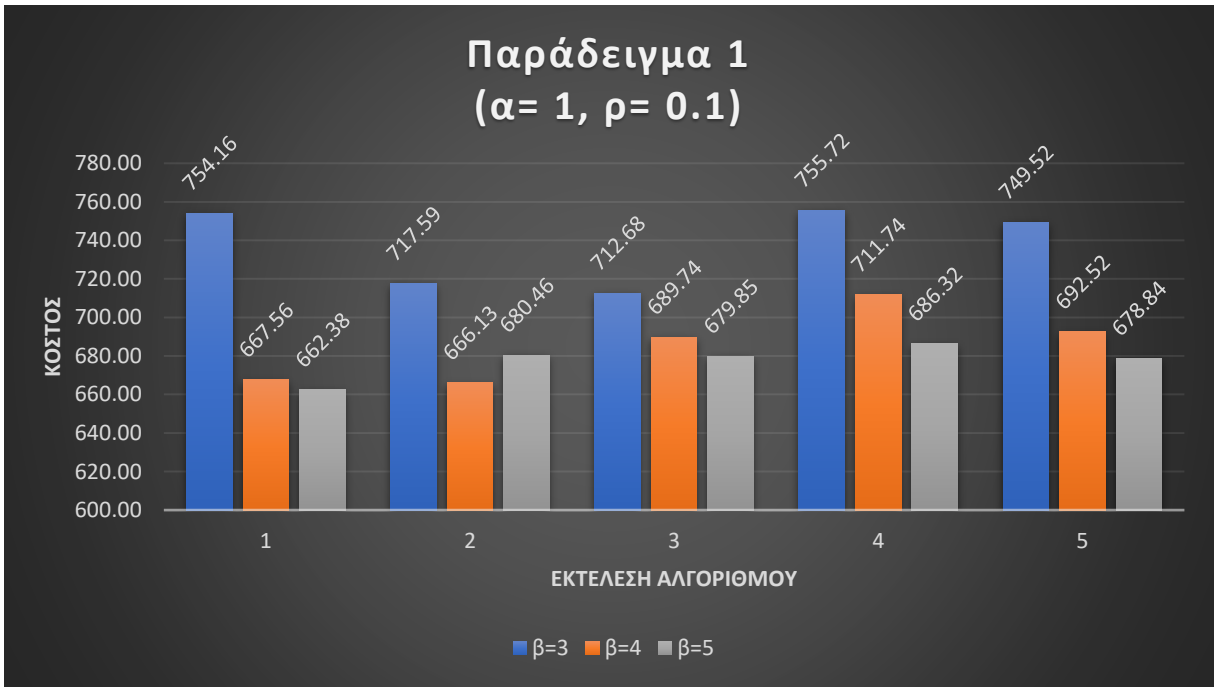
- **Παράδειγμα 11:**
  Αριθμός κόμβων: 121
  Αριθμός πελατών: 120
  Χωρητικότητα Οχήματος: 200

- **Παράδειγμα 12:**
  Αριθμός κόμβων: 101
  Αριθμός πελατών: 100
  Χωρητικότητα Οχήματος: 200

Η διαδικασία έχει ως εξής, το κάθε παράδειγμα επιλύθηκε με 6 διαφορετικούς συνδυασμούς και ο κάθε συνδυασμός του αλγορίθμου εκτελέστηκε από 5 φορές. Οι συνδυασμοί αφορούν τις παραμέτρους α, β και τον συντελεστή ρ που βρίσκονται στο ACO. Συγκεκριμένα, η παράμετρος α παραμένει σταθερή και ισούται με 1, η παράμετρος β λαμβάνει τις τιμές 3, 4, 5 και ο συντελεστής ρ λαμβάνει τις τιμές 0.1 και 0.5. Να σημειωθεί ότι οι επαναλήψεις του κώδικα ορίστηκαν σε 400 και ο αριθμός των μυρμηγκιών σε 25. Τα αποτελέσματα των 6 διαφορετικών συνδυασμών για τα 7 παραδείγματα παρουσιάζονται παρακάτω σε πίνακες και γραφήματα.

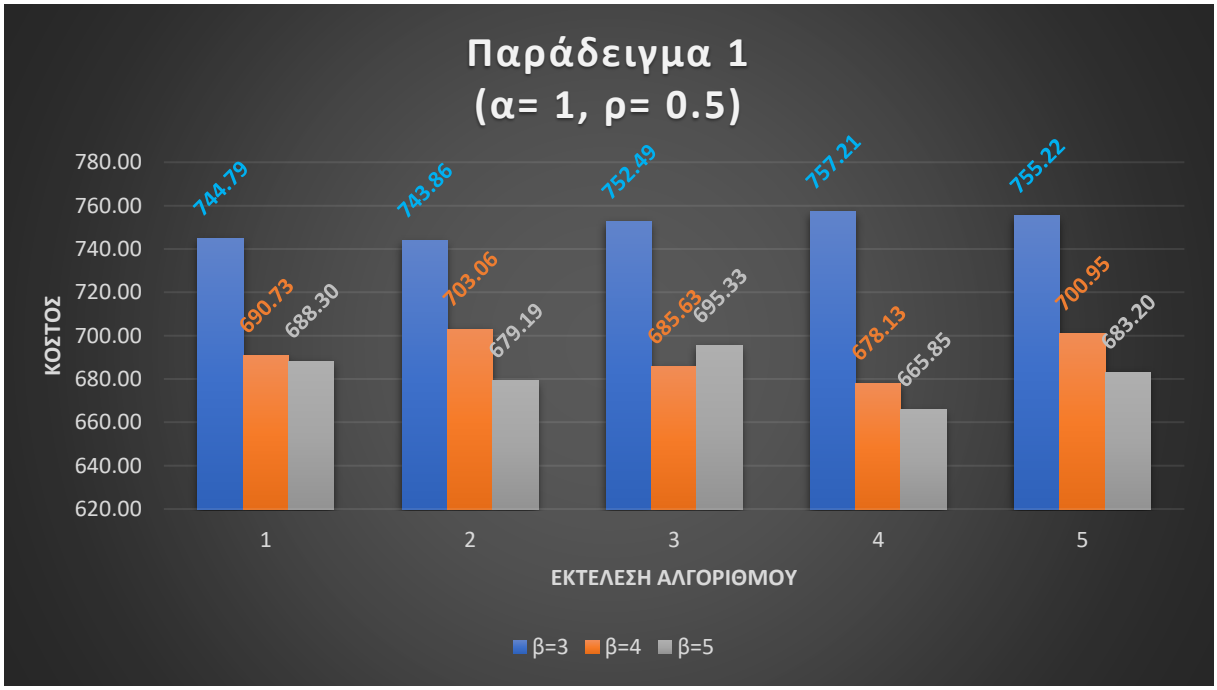
5.1) Παράδειγμα 1

	α= 1, β= 3, ρ= 0.1		α= 1, β= 4, ρ= 0.1		α= 1, β= 5, ρ= 0.1	
	Κόστος	Αριθμός Φορτηγών	Κόστος	Αριθμός Φορτηγών	Κόστος	Αριθμός Φορτηγών
1η Εκτέλεση Αλγορίθμου	754.16	5	667.56	5	**662.38**	5
2η Εκτέλεση Αλγορίθμου	717.59	5	**666.13**	5	680.46	5
3η Εκτέλεση Αλγορίθμου	**712.68**	6	689.74	5	679.85	5
4η Εκτέλεση Αλγορίθμου	755.72	5	711.74	5	686.32	5
5η Εκτέλεση Αλγορίθμου	749.52	5	692.52	5	678.84	5
Avg.	**737.93**	**5.2**	**685.54**	**5**	**677.57**	**5**



Στον παραπάνω συνδυασμό του Παραδείγματος 1 με α= 1 και ρ= 0.1 παρατηρείται ότι η χαμηλότερη τιμή του κόστους και ο μικρότερος αριθμός απαιτούμενων οχημάτων που είναι 5 δίνεται για β= 5. Αξίζει να αναφερθεί πως το χαμηλότερο κόστος για β= 3 εμφανίζεται με 6 απαιτούμενα οχήματα. Η αύξηση αυτή δεν επηρεάζει κάπου το πρόβλημα στην περίπτωση που δεν λαμβάνεται υπόψιν το πάγιο κόστος του οχήματος.

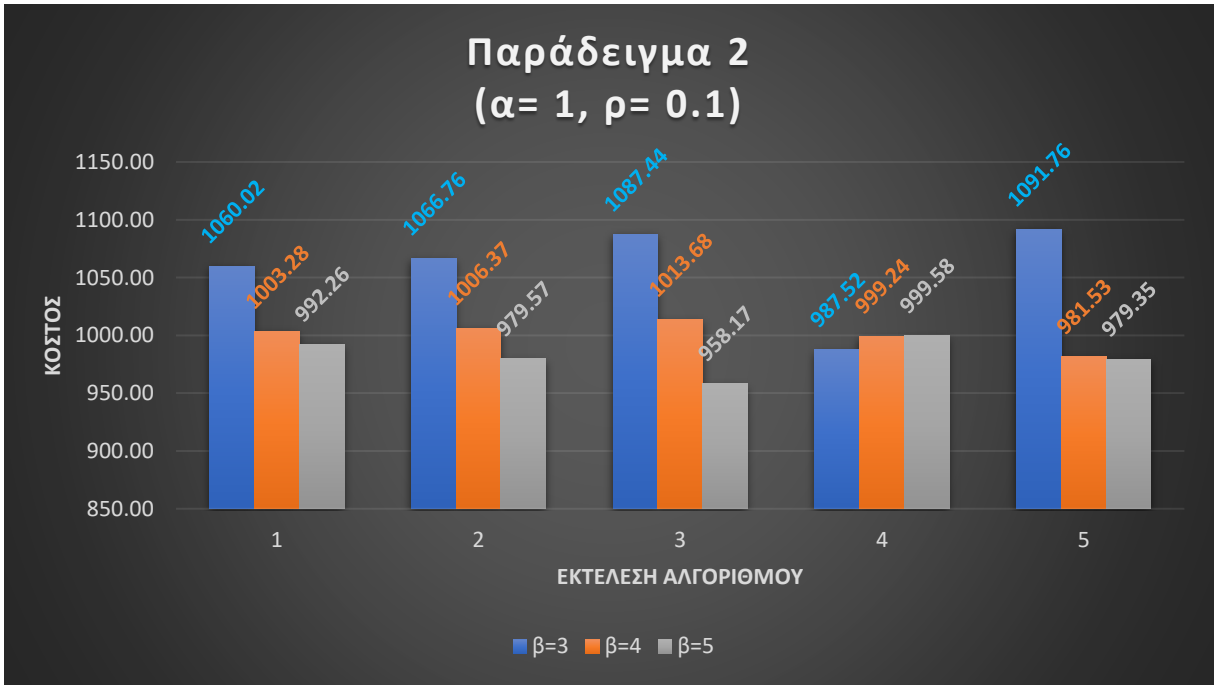
	α= 1, β= 3, ρ= 0.5		α= 1, β= 4, ρ= 0.5		α= 1, β= 5, ρ= 0.5	
	Κόστος	Αριθμός Φορτηγών	Κόστος	Αριθμός Φορτηγών	Κόστος	Αριθμός Φορτηγών
1η Εκτέλεση Αλγορίθμου	744.79	5	690.73	6	688.30	5
2η Εκτέλεση Αλγορίθμου	**743.86**	5	703.06	5	679.19	5
3η Εκτέλεση Αλγορίθμου	752.49	5	685.63	5	695.33	5
4η Εκτέλεση Αλγορίθμου	757.21	5	**678.13**	6	**665.85**	5
5η Εκτέλεση Αλγορίθμου	755.22	5	700.95	5	683.20	5
Avg.	**750.72**	**5**	**691.70**	**5.4**	**682.37**	**5**



Στον δεύτερο συνδυασμό του Παραδείγματος 1 με α= 1 και ρ= 0.5, παρατηρείται ότι η χαμηλότερη τιμή του κόστους και ο χαμηλότερος αριθμός απαιτούμενων οχημάτων, που είναι και πάλι 5, εντοπίζεται για β= 5. Παράλληλα και σε αυτόν τον συνδυασμό υπάρχει περίπτωση που εμφανίζεται χαμηλότερο κόστος με 6 οχήματα, αντί με 5 που είναι το βέλτιστο και εντοπίζεται για β= 4.

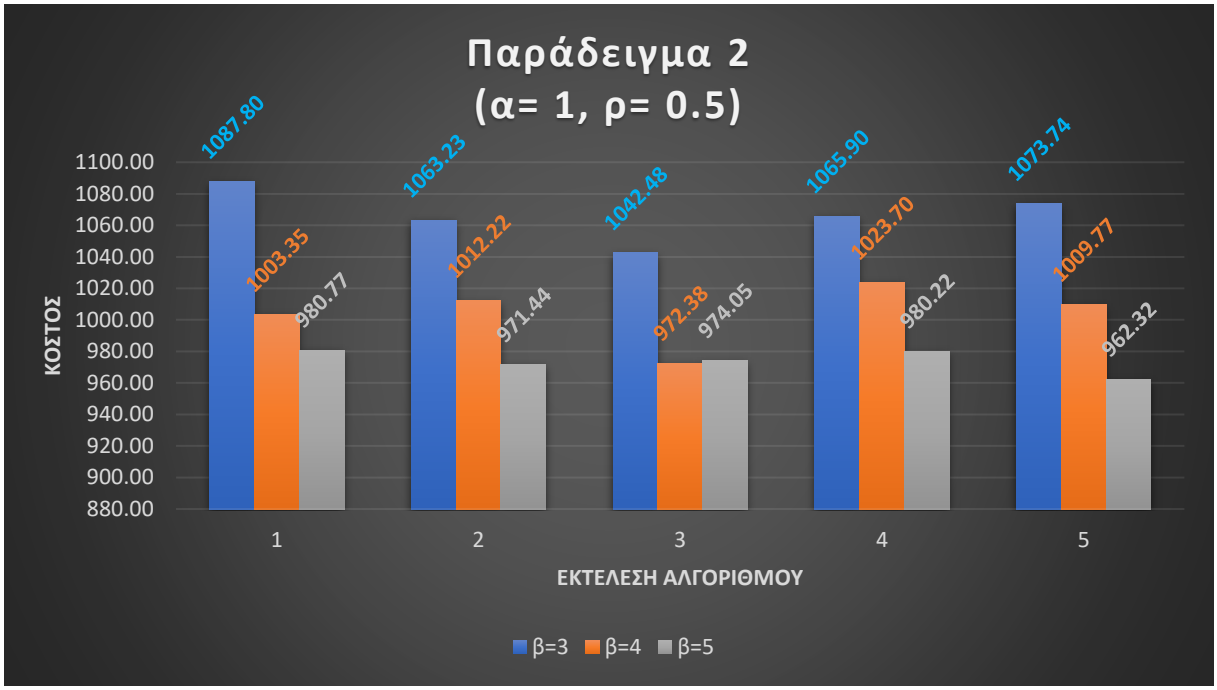
5.2) Παράδειγμα 2

	α= 1, β= 3, ρ= 0.1		α= 1, β= 4, ρ= 0.1		α= 1, β= 5, ρ= 0.1	
	Κόστος	Αριθμός Φορτηγών	Κόστος	Αριθμός Φορτηγών	Κόστος	Αριθμός Φορτηγών
1η Εκτέλεση Αλγορίθμου	1060.02	11	1003.28	11	992.26	11
2η Εκτέλεση Αλγορίθμου	1066.76	11	1006.37	11	979.57	11
3η Εκτέλεση Αλγορίθμου	1087.44	11	1013.68	11	**958.17**	11
4η Εκτέλεση Αλγορίθμου	**987.52**	11	999.24	11	999.58	11
5η Εκτέλεση Αλγορίθμου	1091.76	11	**981.53**	11	979.35	11
Avg.	**1058.70**	**11**	**1000.82**	**11**	**981.78**	**11**



Στην περίπτωση του Παραδείγματος 2, με α= 1 και ρ= 0.1, παρατηρείται ότι ο αριθμός των απαιτούμενων οχημάτων παραμένει σταθερός για όλες τις διαφορετικές τιμές που λαμβάνει το β και ισούται με 11. Το χαμηλότερο κόστος εμφανίζεται στην περίπτωση που το β= 5.

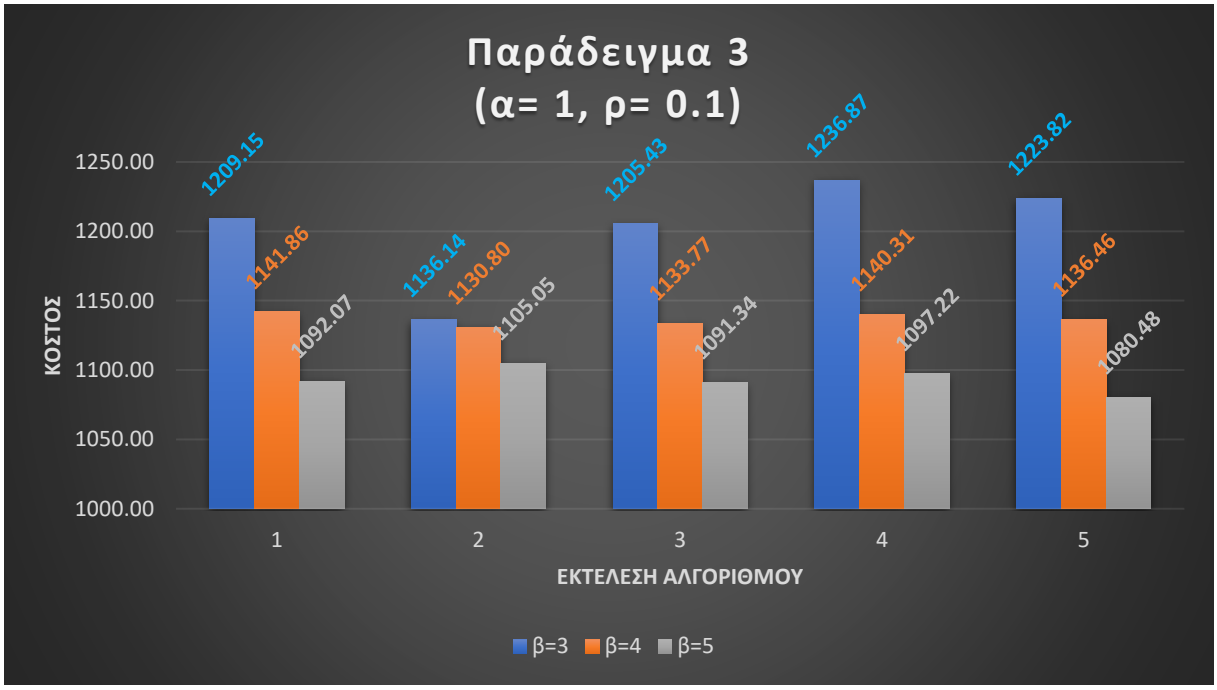
	α= 1, β= 3, ρ= 0.5		α= 1, β= 4, ρ= 0.5		α= 1, β= 5, ρ= 0.5	
	Κόστος	Αριθμός Φορτηγών	Κόστος	Αριθμός Φορτηγών	Κόστος	Αριθμός Φορτηγών
1η Εκτέλεση Αλγορίθμου	1087.80	11	1003.35	11	980.77	11
2η Εκτέλεση Αλγορίθμου	1063.23	11	1012.22	11	971.44	11
3η Εκτέλεση Αλγορίθμου	**1042.48**	11	**972.38**	11	974.05	11
4η Εκτέλεση Αλγορίθμου	1065.90	11	1023.70	11	980.22	11
5η Εκτέλεση Αλγορίθμου	1073.74	11	1009.77	11	**962.32**	11
Avg.	**1066.63**	**11**	**1004.28**	**11**	**973.76**	**11**



Τα αποτελέσματα στον δεύτερο συνδυασμό του Παραδείγματος 2, με α= 1 και ρ= 0.5, δεν διαφοροποιούνται σε σχέση με τον πρώτο όσον αφορά το ποιος συνδυασμός παραμέτρων παρουσιάζει το χαμηλότερο κόστος. Συγκεκριμένα το χαμηλότερο κόστος εμφανίζεται για β= 5, ενώ ο αριθμός των απαιτούμενων οχημάτων είναι σταθερός και ισούται με 11.

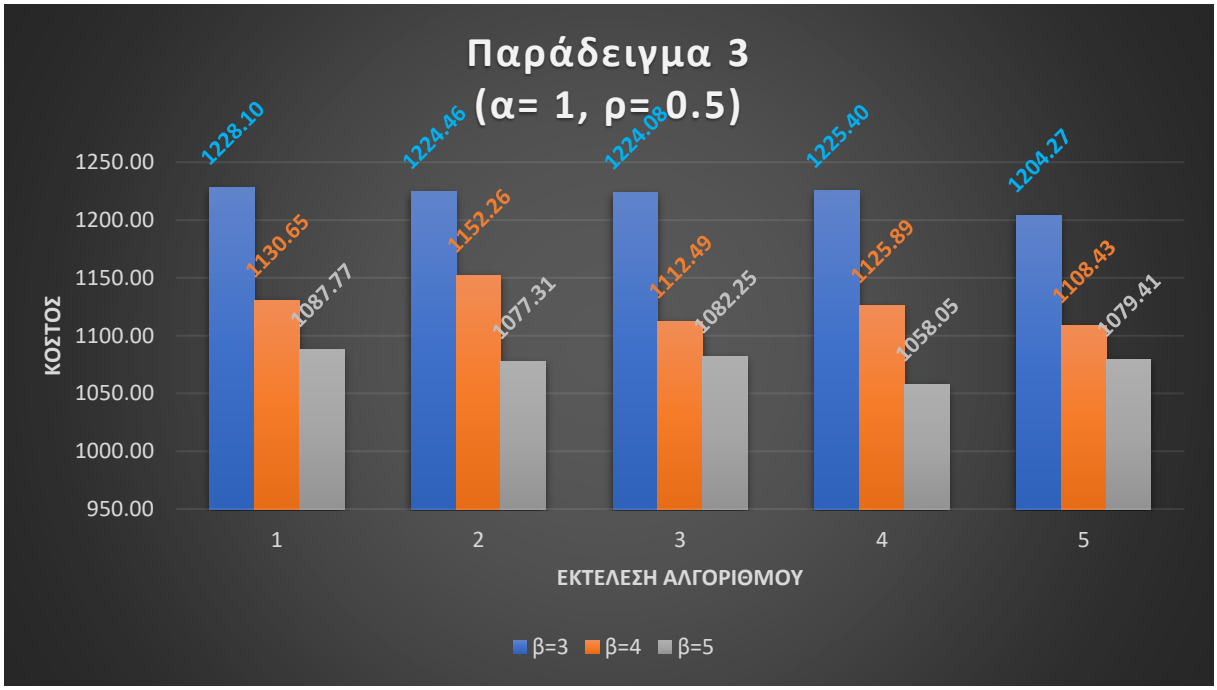
5.3) Παράδειγμα 3

	α= 1, β= 3, ρ= 0.1		α= 1, β= 4, ρ= 0.1		α= 1, β= 5, ρ= 0.1	
	Κόστος	Αριθμός Φορτηγών	Κόστος	Αριθμός Φορτηγών	Κόστος	Αριθμός Φορτηγών
1η Εκτέλεση Αλγορίθμου	1209.15	8	1141.86	8	1092.07	8
2η Εκτέλεση Αλγορίθμου	**1136.14**	8	**1130.80**	8	1105.05	8
3η Εκτέλεση Αλγορίθμου	1205.43	8	1133.77	8	1091.34	8
4η Εκτέλεση Αλγορίθμου	1236.87	8	1140.31	8	1097.22	8
5η Εκτέλεση Αλγορίθμου	1223.82	8	1136.46	8	**1080.48**	8
Avg.	**1202.28**	**8**	**1136.64**	**8**	**1093.23**	**8**



Στα αποτελέσματα του Παραδείγματος 3, με α= 1 και ρ= 0.1, παρατηρείται ότι ο αριθμός των απαιτούμενων οχημάτων παραμένει σταθερός και ισούται με 8, ενώ το χαμηλότερο κόστος εντοπίζεται για β= 5. Αξίζει να σημειωθεί ότι η διαφορά μεταξύ του χαμηλότερου κόστους για β= 3 και του αντίστοιχου για β= 4 είναι πολύ μικρή.

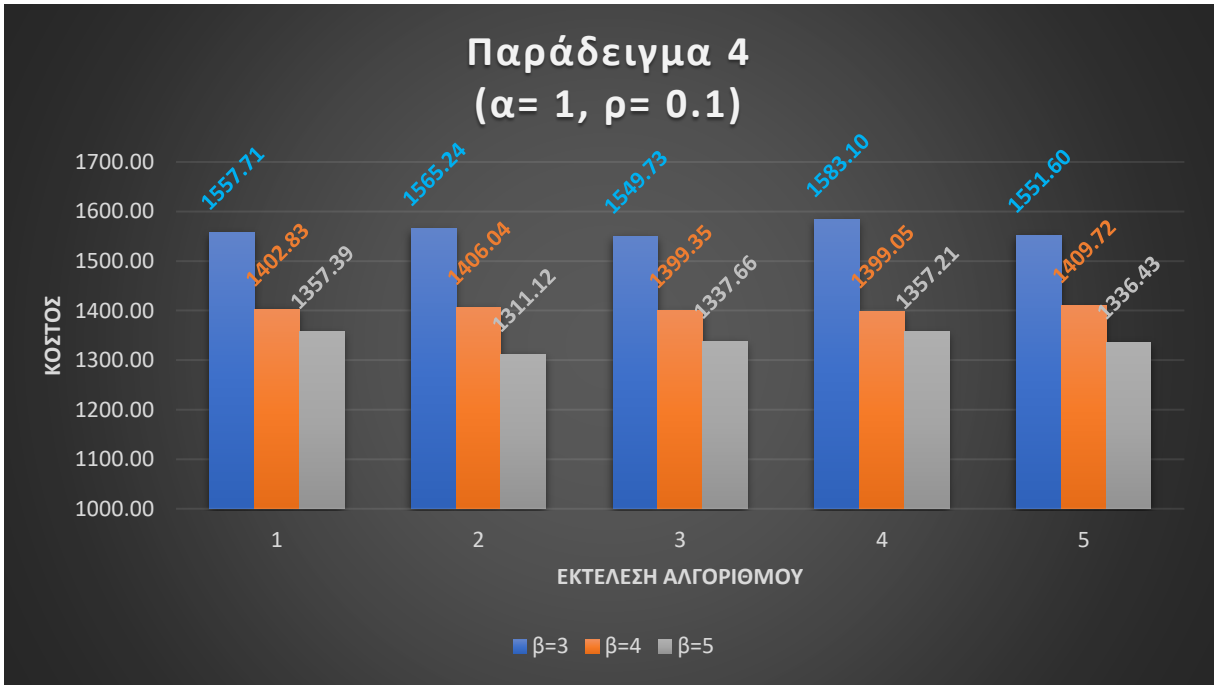
	α= 1, β= 3, ρ= 0.5		α= 1, β= 4, ρ= 0.5		α= 1, β= 5, ρ= 0.5	
	Κόστος	Αριθμός Φορτηγών	Κόστος	Αριθμός Φορτηγών	Κόστος	Αριθμός Φορτηγών
1η Εκτέλεση Αλγορίθμου	1228.10	8	1130.65	8	1087.77	8
2η Εκτέλεση Αλγορίθμου	1224.46	8	1152.26	8	1077.31	8
3η Εκτέλεση Αλγορίθμου	1224.08	8	1112.49	8	1082.25	8
4η Εκτέλεση Αλγορίθμου	1225.40	8	1125.89	8	**1058.05**	8
5η Εκτέλεση Αλγορίθμου	**1204.27**	8	**1108.43**	8	1079.41	8
Avg.	**1221.26**	**8**	**1125.94**	**8**	**1076.96**	**8**



Στην δεύτερη περίπτωση του Παραδείγματος 3, με α= 1 και ρ= 0.5, η χαμηλότερη τιμή του κόστους εντοπίζεται για β= 5 και ο αριθμός των απαιτούμενων οχημάτων παραμένει σταθερός και στους 3 συνδυασμούς, δηλαδή β= 3, β= 4 και β= 5, όπου ισούται με 8.

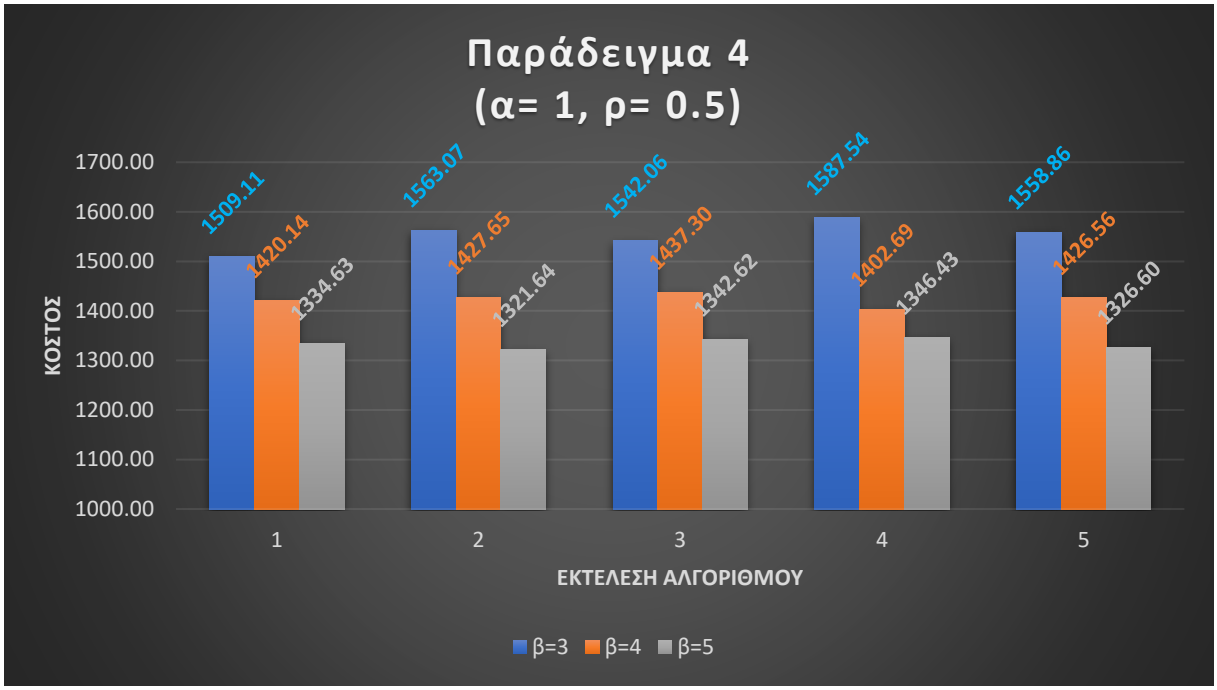
5.4) Παράδειγμα 4

	α= 1, β= 3, ρ= 0.1		α= 1, β= 4, ρ= 0.1		α= 1, β= 5, ρ= 0.1	
	Κόστος	Αριθμός Φορτηγών	Κόστος	Αριθμός Φορτηγών	Κόστος	Αριθμός Φορτηγών
1η Εκτέλεση Αλγορίθμου	1557.71	12	1402.83	12	1357.39	12
2η Εκτέλεση Αλγορίθμου	1565.24	12	1406.04	12	**1311.12**	12
3η Εκτέλεση Αλγορίθμου	**1549.73**	12	1399.35	12	1337.66	12
4η Εκτέλεση Αλγορίθμου	1583.10	12	**1399.05**	12	1357.21	12
5η Εκτέλεση Αλγορίθμου	1551.60	12	1409.72	12	1336.43	12
Avg.	**1561.48**	**12**	**1403.40**	**12**	**1339.96**	**12**



Στο Παράδειγμα 4 με α= 1 και ρ= 0.1 η χαμηλότερη τιμή του κόστους εμφανίζεται για β= 5, ο αριθμός των απαιτούμενων οχημάτων παραμένει σταθερός και στις 3 περιπτώσεις και ισούται με 12.

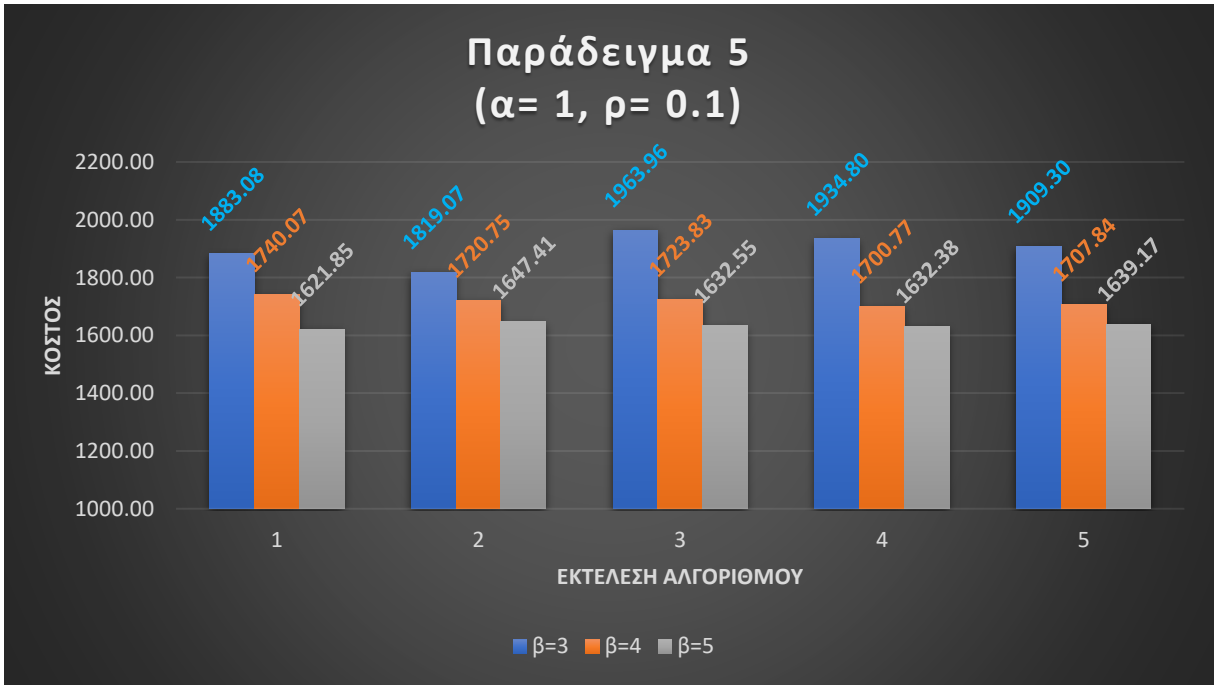
	α= 1, β= 3, ρ= 0.5		α= 1, β= 4, ρ= 0.5		α= 1, β= 5, ρ= 0.5	
	Κόστος	Αριθμός Φορτηγών	Κόστος	Αριθμός Φορτηγών	Κόστος	Αριθμός Φορτηγών
1η Εκτέλεση Αλγορίθμου	**1509.11**	12	1420.14	12	1334.63	12
2η Εκτέλεση Αλγορίθμου	1563.07	12	1427.65	12	**1321.64**	12
3η Εκτέλεση Αλγορίθμου	1542.06	12	1437.30	12	1342.62	12
4η Εκτέλεση Αλγορίθμου	1587.54	12	**1402.69**	12	1346.43	12
5η Εκτέλεση Αλγορίθμου	1558.86	12	1426.56	12	1326.60	12
Avg.	**1552.13**	**12**	**1422.87**	**12**	**1334.38**	**12**



Στην δεύτερη περίπτωση του Παραδείγματος 4 με α= 1 και ρ= 0.5 το χαμηλότερο κόστος εντοπίζεται για β= 5, ο αριθμός των απαιτούμενων οχημάτων παραμένει σταθερός και στις 3 περιπτώσεις και ισούται με 12.

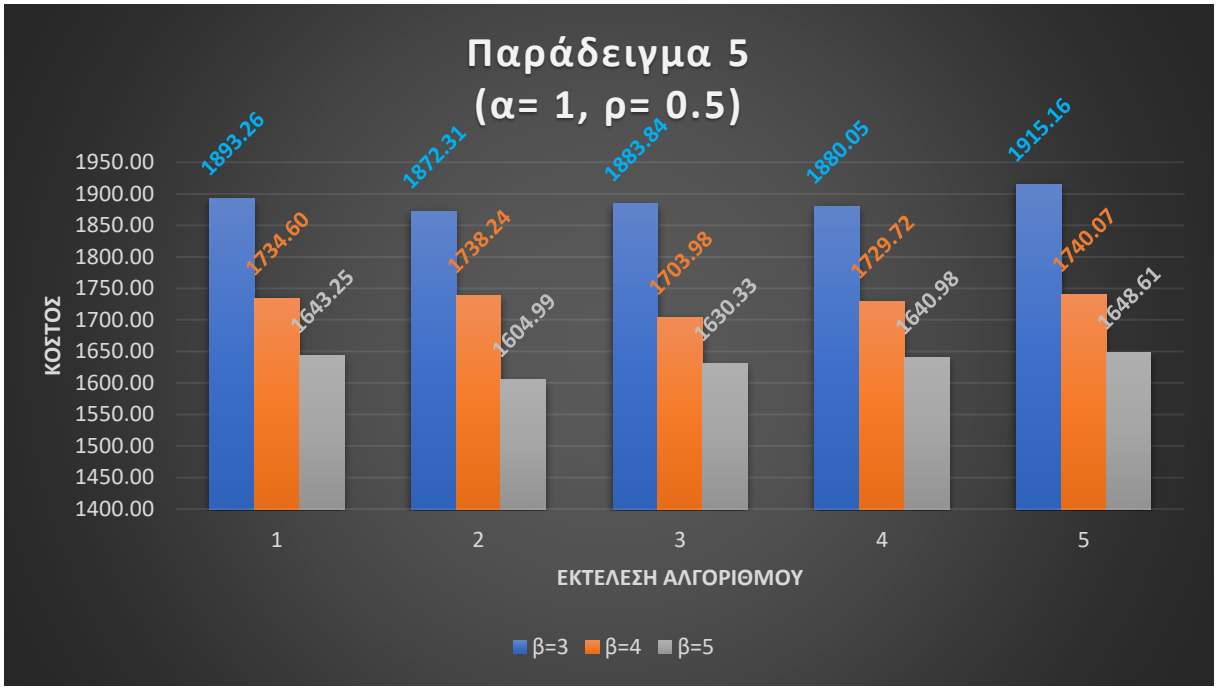
5.5) Παράδειγμα 5

	α= 1, β= 3, ρ= 0.1		α= 1, β= 4, ρ= 0.1		α= 1, β= 5, ρ= 0.1	
	Κόστος	Αριθμός Φορτηγών	Κόστος	Αριθμός Φορτηγών	Κόστος	Αριθμός Φορτηγών
1η Εκτέλεση Αλγορίθμου	1883.08	17	1740.07	17	**1621.85**	17
2η Εκτέλεση Αλγορίθμου	**1819.07**	17	1720.75	17	1647.41	17
3η Εκτέλεση Αλγορίθμου	1963.96	17	1723.83	17	1632.55	17
4η Εκτέλεση Αλγορίθμου	1934.80	17	**1700.77**	17	1632.38	17
5η Εκτέλεση Αλγορίθμου	1909.30	17	1707.84	17	1639.17	17
Avg.	**1902.04**	**17**	**1718.65**	**17**	**1634.67**	**17**



Στο Παράδειγμα 5 με α= 1 και ρ= 0.1 η χαμηλότερη τιμή του κόστους εμφανίζεται για β= 5, ο αριθμός των απαιτούμενων οχημάτων παραμένει σταθερός και στις 3 περιπτώσεις και ισούται με 17.

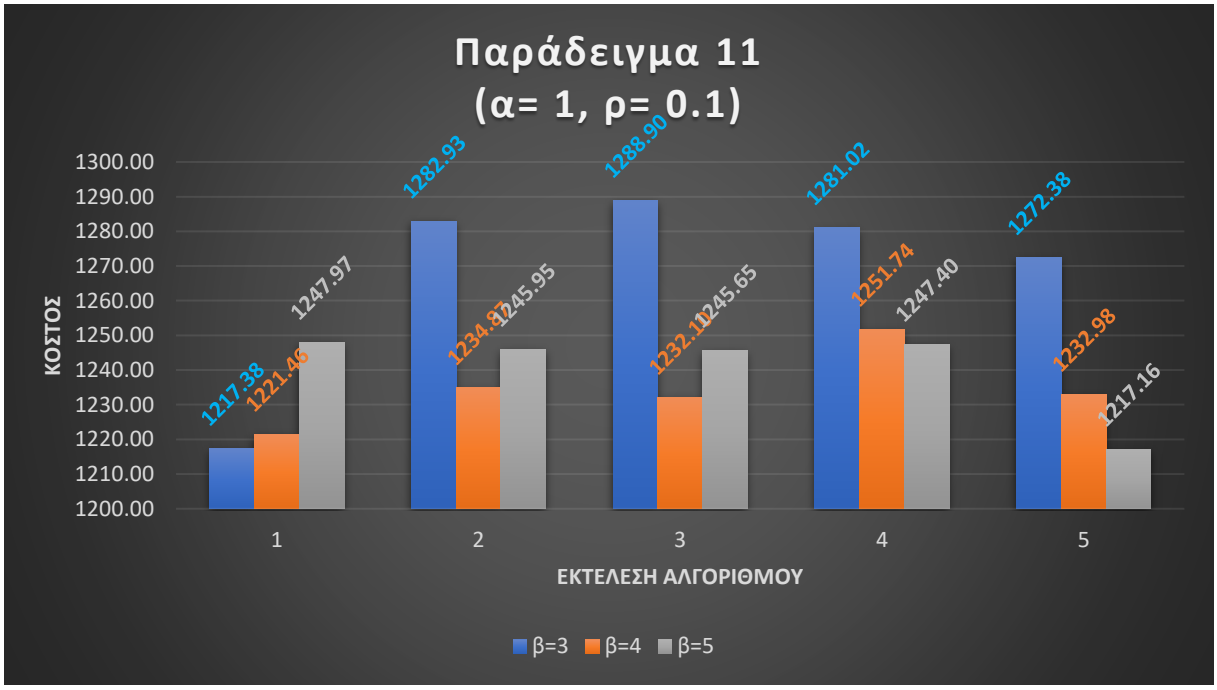
	α= 1, β= 3, ρ= 0.5		α= 1, β= 4, ρ= 0.5		α= 1, β= 5, ρ= 0.5	
	Κόστος	Αριθμός Φορτηγών	Κόστος	Αριθμός Φορτηγών	Κόστος	Αριθμός Φορτηγών
1η Εκτέλεση Αλγορίθμου	1893.26	17	1734.60	17	1643.25	17
2η Εκτέλεση Αλγορίθμου	**1872.31**	17	1738.24	17	**1604.99**	17
3η Εκτέλεση Αλγορίθμου	1883.84	17	**1703.98**	17	1630.33	17
4η Εκτέλεση Αλγορίθμου	1880.05	17	1729.72	17	1640.98	17
5η Εκτέλεση Αλγορίθμου	1915.16	17	1740.07	17	1648.61	17
Avg.	**1888.92**	**17**	**1729.32**	**17**	**1633.63**	**17**



Στην δεύτερη περίπτωση του Παραδείγματος 5, με α= 1 και ρ= 0.5, η χαμηλότερη τιμή του κόστους εντοπίζεται για β= 5 και ο αριθμός των απαιτούμενων οχημάτων παραμένει σταθερός και στους 3 συνδυασμούς, δηλαδή β= 3, β= 4 και β= 5, όπου ισούται με 17.

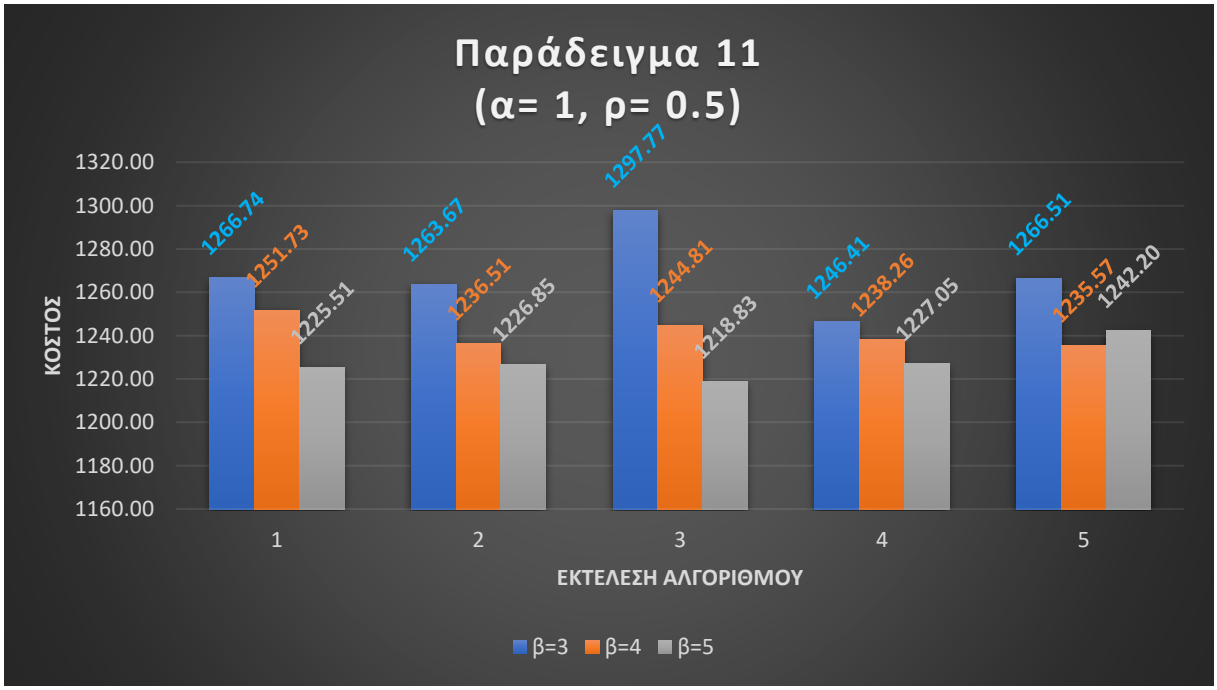
5.6) Παράδειγμα 11

	α= 1, β= 3, ρ= 0.1		α= 1, β= 4, ρ= 0.1		α= 1, β= 5, ρ= 0.1	
	Κόστος	Αριθμός Φορτηγών	Κόστος	Αριθμός Φορτηγών	Κόστος	Αριθμός Φορτηγών
1η Εκτέλεση Αλγορίθμου	**1217.38**	7	**1221.46**	7	1247.97	7
2η Εκτέλεση Αλγορίθμου	1282.93	7	1234.87	7	1245.95	7
3η Εκτέλεση Αλγορίθμου	1288.90	7	1232.10	7	1245.65	7
4η Εκτέλεση Αλγορίθμου	1281.02	7	1251.74	7	1247.40	7
5η Εκτέλεση Αλγορίθμου	1272.38	7	1232.98	7	**1217.16**	7
Avg.	**1268.52**	**7**	**1234.63**	**7**	**1240.82**	**7**



Στο Παράδειγμα 11 με α= 1 και ρ= 0.1 τα αποτελέσματα είναι αρκετά διαφορετικά σε σχέση με τα προηγούμενα παραδείγματα. Συγκεκριμένα η χαμηλότερη τιμή στο κόστος εμφανίζεται για β= 5 αλλά οι αντίστοιχες δύο τιμές στις άλλες δύο περιπτώσεις, δηλαδή για β= 3 και για β= 4, είναι πολύ κοντά. Ο αριθμός των απαιτούμενων οχημάτων είναι ο ίδιος για τις 3 περιπτώσεις και ισούται με 7.

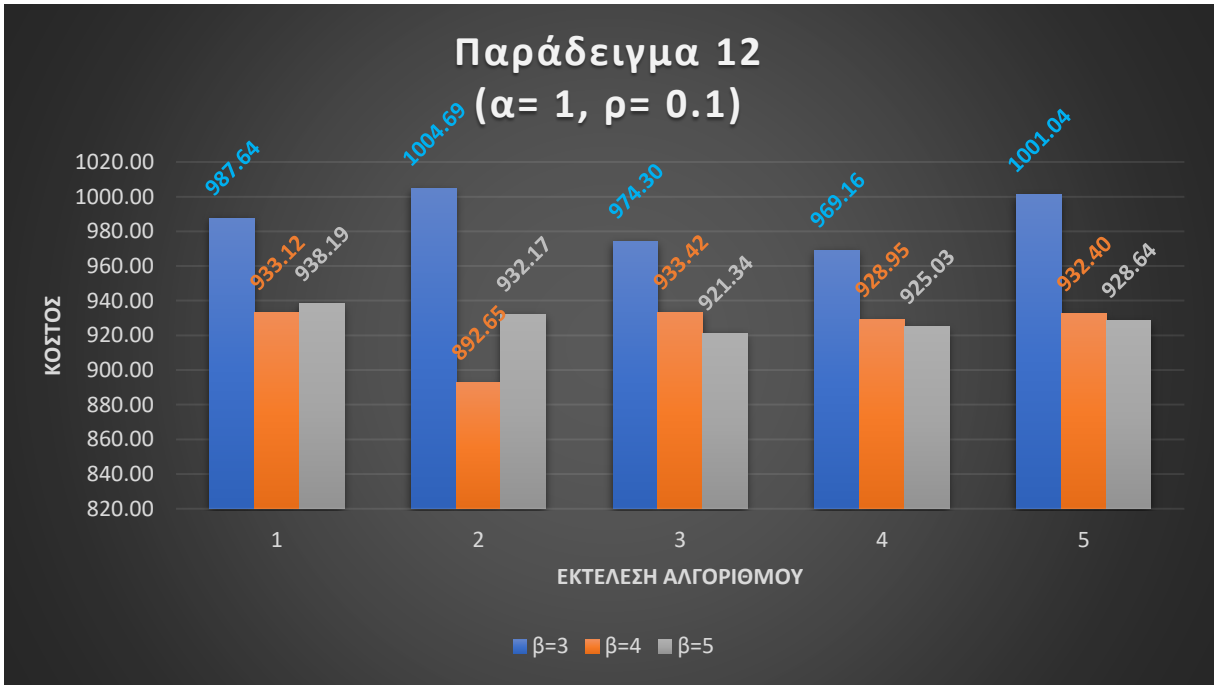
	α= 1, β= 3, ρ= 0.5		α= 1, β= 4, ρ= 0.5		α= 1, β= 5, ρ= 0.5	
	Κόστος	Αριθμός Φορτηγών	Κόστος	Αριθμός Φορτηγών	Κόστος	Αριθμός Φορτηγών
1η Εκτέλεση Αλγορίθμου	1266.74	7	1251.73	7	1225.51	7
2η Εκτέλεση Αλγορίθμου	1263.67	7	1236.51	7	1226.85	7
3η Εκτέλεση Αλγορίθμου	1297.77	7	1244.81	7	**1218.83**	7
4η Εκτέλεση Αλγορίθμου	**1246.41**	7	1238.26	7	1227.05	7
5η Εκτέλεση Αλγορίθμου	1266.51	7	**1235.57**	7	1242.20	7
Avg.	**1268.22**	**7**	**1241.37**	**7**	**1228.09**	**7**



Στην δεύτερη περίπτωση του Παραδείγματος 11, με α= 1 και ρ= 0.5, η χαμηλότερη τιμή στο κόστος εντοπίζεται για β= 5 με τις τιμές στις άλλες δύο περιπτώσεις, δηλαδή για β= 3 και για β= 4, να είναι εξίσου κοντά αλλά όχι το ίδιο κοντά σε σύγκριση με πριν. Ο αριθμός των απαιτούμενων οχημάτων είναι και πάλι ο ίδιος με πριν, δηλαδή ισούται με 7 και είναι ίδιος για τις 3 διαφορετικές τιμές του β.

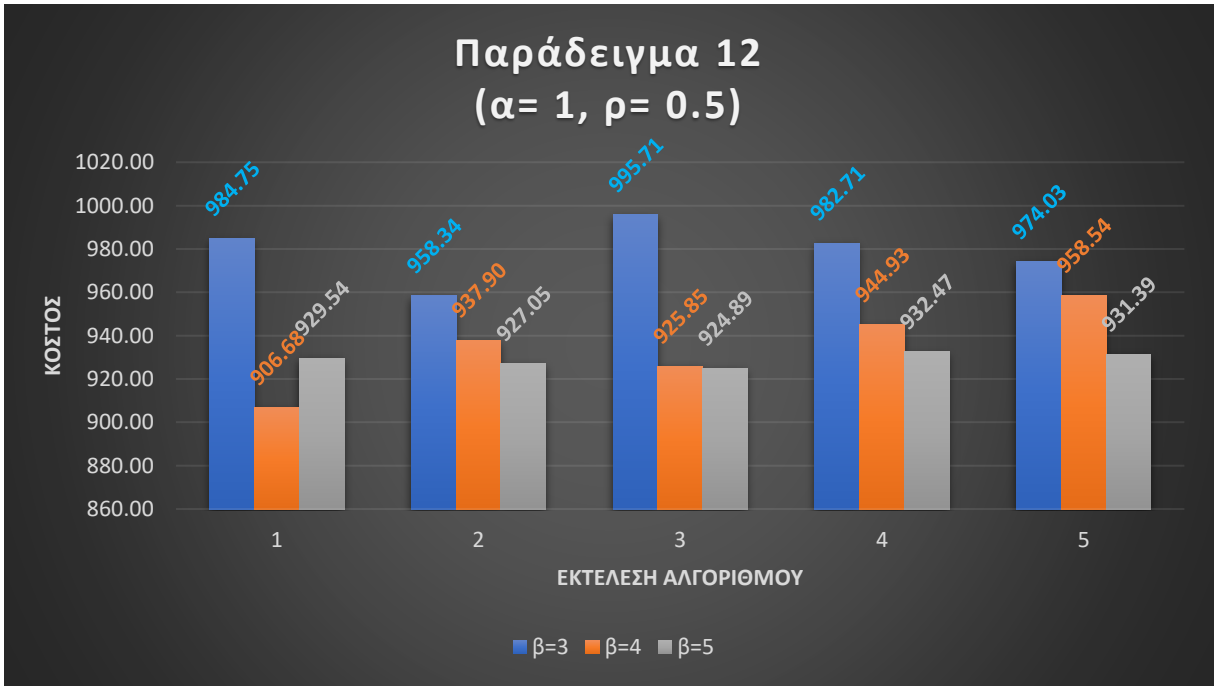
5.7) Παράδειγμα 12

	α= 1, β= 3, ρ= 0.1		α= 1, β= 4, ρ= 0.1		α= 1, β= 5, ρ= 0.1	
	Κόστος	Αριθμός Φορτηγών	Κόστος	Αριθμός Φορτηγών	Κόστος	Αριθμός Φορτηγών
1η Εκτέλεση Αλγορίθμου	987.64	10	933.12	10	938.19	10
2η Εκτέλεση Αλγορίθμου	1004.69	10	**892.65**	10	932.17	10
3η Εκτέλεση Αλγορίθμου	974.30	10	933.42	10	**921.34**	10
4η Εκτέλεση Αλγορίθμου	**969.16**	10	928.95	10	925.03	10
5η Εκτέλεση Αλγορίθμου	1001.04	10	932.40	10	928.64	10
Avg.	**987.37**	**10**	**924.11**	**10**	**929.07**	**10**



Στο Παράδειγμα 12 με α= 1 και ρ= 0.1 η χαμηλότερη τιμή στο κόστος εμφανίζεται για β= 4, με τον αριθμό των απαιτούμενων οχημάτων να είναι ο ίδιος στις 3 περιπτώσεις, δηλαδή για β= 3, για β= 4 και για β= 5 και να ισούται με 10.

	α= 1, β= 3, ρ= 0.5		α= 1, β= 4, ρ= 0.5		α= 1, β= 5, ρ= 0.5	
	Κόστος	Αριθμός Φορτηγών	Κόστος	Αριθμός Φορτηγών	Κόστος	Αριθμός Φορτηγών
1η Εκτέλεση Αλγορίθμου	984.75	10	**906.68**	10	929.54	10
2η Εκτέλεση Αλγορίθμου	**958.34**	10	937.90	10	927.05	10
3η Εκτέλεση Αλγορίθμου	995.71	10	925.85	10	**924.89**	10
4η Εκτέλεση Αλγορίθμου	982.71	10	944.93	10	932.47	10
5η Εκτέλεση Αλγορίθμου	974.03	10	958.54	10	931.39	10
Avg.	**979.11**	**10**	**934.78**	**10**	**929.07**	**10**



Στην δεύτερη περίπτωση του Παραδείγματος 12, με α= 1 και ρ= 0.5, η χαμηλότερη τιμή στο κόστος εντοπίζεται και πάλι για β= 4. Ο αριθμός των οχημάτων παραμένει σταθερός και στους 3 διαφορετικούς συνδυασμούς και ισούται με 10.

5.8) Καλύτερα Αποτελέσματα

Παράδειγμα	Συνδυασμός	Κόστος	Αριθμός Φορτηγών
1	α= 1, β= 5, ρ= 0.1	662.38	5
2	α= 1, β= 5, ρ= 0.1	958.17	11
3	α= 1, β= 5, ρ= 0.5	1058.05	8
4	α= 1, β= 5, ρ= 0.1	1311.12	12
5	α= 1, β= 5, ρ= 0.5	1604.99	17
11	α= 1, β= 5, ρ= 0.1	1217.16	7
12	α= 1, β= 4, ρ= 0.1	892.65	10

Ο παραπάνω πίνακας παρουσιάζει τα καλύτερα αποτελέσματα των 6 διαφορετικών συνδυασμών, όπου ο κάθε συνδυασμός επαναλήφθηκε από 5 φορές, για τα 7 παραδείγματα. Παρατηρείται ότι η παράμετρος β δίνει το καλύτερο αποτέλεσμα, στα 6 από τα 7 παραδείγματα, για τιμή ίση με 5, ενώ σε 1 παράδειγμα και συγκεκριμένα στο 12 το καλύτερο αποτέλεσμα προκύπτει για β= 4. Έτσι προκύπτει το συμπέρασμα ότι η παράμετρος β, που καθορίζει το πόσο η ευρετική πληροφορία επηρεάζει την επιλογή του επόμενου πελάτη, είναι προτιμότερο να λαμβάνει μεγάλες τιμές. Όσον αφορά τον συντελεστή εξάτμισης ρ, παρατηρείται ότι στα 5 Παραδείγματα και συγκεκριμένα στα 1, 2, 4, 11, 12 το καλύτερο αποτέλεσμα το δίνει η τιμή 0.1, ενώ στα Παραδείγματα 3 και 5 το καλύτερο αποτέλεσμα το δίνει η τιμή 0.5. Αυτό σημαίνει πως χαμηλή τιμή στον συντελεστή εξάτμισης ρ δίνει το καλύτερο αποτέλεσμα.

6) Συμπέρασμα

Συνοψίζοντας, η ανθρωπότητα αντιμετωπίζει και έχει να αντιμετωπίσει σοβαρά περιβαλλοντικά ζητήματα. Ένα από αυτά είναι η κλιματική αλλαγή όπου ακραία καιρικά φαινόμενα έχουν κάνει την εμφάνιση τους σε όλο τον κόσμο και πρόκειται να αυξηθεί η συχνότητα τους. Όπως προαναφέρθηκε σε προηγούμενο κεφάλαιο, ένας από τους λόγους που συνετέλεσε στην όξυνση των περιβαλλοντικών προβλημάτων είναι η ατμοσφαιρική ρύπανση, με ένα από τα αίτια που την προκαλεί να είναι οι εκπομπές ρύπων των οχημάτων. Γι' αυτό τον λόγο θα πρέπει να μειωθούν και μία λύση είναι η ελαχιστοποίηση της κατανάλωσης των καυσίμων των οχημάτων. Παράλληλα, οι τιμές των καυσίμων τα τελευταία χρόνια έχουν αυξηθεί ραγδαία, έτσι οι εταιρείες στον τομέα της εφοδιαστικής αλυσίδας έχουν άμεσο συμφέρον στο να μειωθεί η κατανάλωση των καυσίμων των οχημάτων τους. Επίσης, τα τελευταία χρόνια έχει παρατηρηθεί ότι πολλές εταιρείες, όσον αφορά τον στόλο των οχημάτων τους, έχουν κατευθυνθεί προς την ενοικίαση (leasing) οχημάτων από εταιρείες του αντίστοιχου τομέα. Έτσι, το Ανοιχτό Πρόβλημα Δρομολόγησης Οχημάτων (Open Vehicle Routing Problem – OVRP) με στόχο την Ελαχιστοποίηση της Κατανάλωσης Καυσίμων (Fuel Consumption) πρόκειται για ένα επίκαιρο πρόβλημα δρομολόγησης οχημάτων. Γενικά, τέτοιου είδους προβλήματα δρομολόγησης οχημάτων μπορούν να χρησιμοποιηθούν από εταιρείες που βρίσκονται στον τομέα της εφοδιαστικής αλυσίδας ώστε να επωφεληθούν σε οικονομικό και σε λειτουργικό επίπεδο.

Στην παρούσα διπλωματική εργασία επιλύθηκε το πρόβλημα βελτιστοποίησης που αναφέρθηκε προηγούμενα. Ο αλγόριθμος που χρησιμοποιήθηκε για την επίλυση του προαναφερθέντος προβλήματος δημιουργήθηκε στο περιβάλλον της Matlab και είναι ο Αλγόριθμος Βελτιστοποίησης Αποικίας Μυρμηγκιών. Ο αλγόριθμος επίλυσε το πρόβλημα με 6 διαφορετικούς συνδυασμούς των παραμέτρων του, τα αποτελέσματα των οποίων παρουσιάστηκαν αναλυτικά παραπάνω σε πίνακες και γραφήματα. Από τα αποτελέσματα εξάγεται το συμπέρασμα ότι η παράμετρος β που καθορίζει το πόσο επηρεάζει η ευρετική πληροφορία την επιλογή του επόμενου πελάτη είναι προτιμότερο να λαμβάνει μεγάλες τιμές. Επίσης, από τα αποτελέσματα εξάγεται και το συμπέρασμα ότι το ρ, που είναι ο συντελεστής εξάτμισης της φερομόνης, είναι προτιμότερο να λαμβάνει χαμηλές τιμές. Έτσι, ρυθμίζοντας κατά αυτόν τον τρόπο τις δύο αυτές παραμέτρους του αλγορίθμου, πραγματοποιείται η ελαχιστοποίηση του κόστους και λαμβάνεται η βέλτιστη λύση του προβλήματος.

7) Βιβλιογραφία

- Ιωάννης Μαρινάκης, Μαγδαληνή Μαρινάκη, Αθανάσιος Μυγδαλάς (2019). Προβλήματα Δρομολόγησης Οχημάτων στη Διαχείριση της Εφοδιαστικής Αλυσίδας, Μοντελοποίηση & Αλγόριθμοι Επίλυσης, Εκδόσεις Νέων Τεχνολογιών.

- Xiao, Y., Zhao, Q., Kaku, I., & Xu, Y. (2012). Development of a fuel consumption optimization model for the capacitated vehicle routing problem. Computers & operations research, 39(7), 1419-1431.

- Ramtake, D., Kumar, S., & Patle, V. K. (2016). Route optimisation byant colony optimisation technique. Procedia Computer Science, 92, 48-55.

- Asghari, M., & Al-e, S. M. J. M. (2021). Green vehicle routing problem: A state-of-the-art review. International Journal of Production Economics, 231, 107899.