Πολυτεχνείο Κρήτης

Διατριβή Μεταπτυχιακού

Επίλυση Προβλήματων Δρομολόγησης Αποθεμάτων με τη χρήση του Αλγορίθμου Βελτιστοποίησης Αποικίας Μυρμηγκιών

Συγγραφέας:
Ανδρέας Καμπιανάκης

Επιβλέπων:
Ιωάννης Μαρινάκης

Σχολή:
Μηχανικών Παραγωγής και Διοίκησης

Περιεχόμενα

Ευχαριστίες

Θα ήθελα να ευχαριστήσω την οικογένεια μου, που με στήριξε στην πορεία μέχρι το τέλος αυτής της εργασίας. Επιπλέον, ιδιαίτερες ευχαριστίες στου δρ. Μαρινάκη, για την βοήθεια και τη γνώση που παρέδωσε όλα αυτά τα χρόνια και σε αυτή τη διατριβή. Τέλος θα ήθελα να ευχαριστήσω τους Μαρκουλάκη Βασίλη, Σαράντη Γιάννη, Ιάκωβο Μαστρογιαννάκη, Μιχάλη Τζαγκαράκη, Γιώργο Ζωγράφο και Κατερίνα Βαλουμά για την παρακίνηση που προσέφεραν σε κάθε δύσκολη στιγμή.

Περίληψη

Η διατριβή μεταπτυχιακού που εκπονήθηκε στοχεύει στην επίλυση του προβλήματος δρομολόγησης αποθεμάτων (Inventory Routing Problem, IRP) με την χρήση χρονικών περιορισμών (Time Window,TW) μέσα απο την χρήση του αλγορίθμου της Βελτιστοποίησης Αποικίας Μυρμηγκιών (Ant Colony Optimization, ACO). Το πρόβλημα IRP έχει σαν στόχο την επαναλαμβανόμενη διανομή ενός απλού προϊόντος από μια απλή εγκατάσταση σε ένα σύνολο από N πελάτες μέσα σε χρονικό ορίζοντα μήκους T ημερών. Σε αυτή την προσέγγιση, ο πωλητής έχει τον απόλυτο έλεγχο του ανεφοδιασμού των αποθεμάτων των πελατών, και έτσι θα μπορούσαν πολύ εύκολα να ελεγχθούν οι μεταφορές και να μειωθεί το κόστος διανομής. Οι αποφάσεις ανεφοδιασμού των πελατών και δρομολόγησης των οχημάτων λαμβάνονται από τον πωλητή.

Ο αλγόριθμος ACO κάνει χρήση των έννοιων της φερομόνης και της εξάτμισης, σαν ένα είδος μνήμης προηγούμενων λύσεων, ώστε να γίνει εύρεση της βέλτιστης. Για να αποφύγει ο αλγόριθμος να βρίσκει διαρκώς την ίδια λύση, κάθε νέος κόμβος προστίθεται βάση πιθανοτικού κανόνα. Όλα αυτά θα αναλυθούν στα παρακάτω κεφάλαια.

Κεφάλαιο 1

Εισαγωγή στην Εφοδιαστική Αλυσίδα

Οι ανάγκες της σύγχρονης εποχής σε επίπεδο κοινωνικό και επιχειρηματικό είναι αυξημένες. Οι ανάγκες των καταναλωτών τοσο σε προϊόντα όσο και στη διαθεσιμότητά τους, ποσοτική και ποιοτική έχει οδηγήσει σε εφαρμογή νέων μεθόδων για την ικανοποιήσή τους. Οι επιστημονικές μέθοδοι αυτοί έχουν μεγάλη εφαρμογή στην εφοδιαστική αλυσίδα, μιας και εκεί θεωρείται οτι οι επιχειρήσεις μπορούν να έχουν ανταγωνιστικό πλεονέκτημα.

Η εφοδιαστική αλυσίδα είναι κάτι παραπάνω απο απλή ροή προϊόντων απο τον κατασκευαστή προς τον καταναλωτή. Τα προϊόντα είναι ακολουθούμενα απο μια σειρά πληροφοριών και υπηρεσιών, τα οποία δημιουργούν μια εικόνα για την επιχείρηση, καθώς επίσης και μέσα απο αυτή τη ροή δημιουργουνται και αλληλεπιδράσεις μεταξύ των μελών της. Επιπλέον, η εφοδιαστική αλυσίδα είναι ένα κανάλι επικοινωνίας του εκάστοτε οργανισμού με τον έξω κόσμο, άρα καθιστά την ανάγκη της ορθής διαχείρισής της πολύ μεγάλη.

Η διαχείρισή της γίνεται σε δύο επίπεδα:

- Προγραμματιστικό: είναι το επίπεδο εκείνο κατα το οποίο πραγματώνονται αναλύσεις, βελτιστοποιήσεις και προβλέψεις βάσει δεδομένων. Μέσα απο αυτές τις αναλύσεις δημιουργείται σχέδιο δράσης και προγραμματίζονται τα μελλοντικά βήματα της επιχείρησης.
- Επιχειρησιακό: το πλάνο το οποίο σχεδιάστηκε κατά το παραπάνω επίπεδο εκτελείται κατά το συγκεκριμένο στάδιο. Η πορεία του συγκεκριμένου πλάνου καθοριζέται δυναμικά, βασιζόμενοι στις πληροφορίες και στα δεδομένα που λαμβάνονται απο όλη την γραμμή εφοδιαστικής αλυσίδα.

Η εφοδιαστική αλυσίδα αποτελείται απο κάποια οργανικά μέρη όπως οι προμηθευτές, οι αποθήκες, τα κέντρα διανομής, τα αποθέματα κ.α. Η βελτιστοποίηση του συνόλου της εφοδιαστικής αλυσίδας, αφορά την βελτίωση των αλληλεπιδράσεων των οργανικών μερών, μέσα από συγκεκριμένη μεθοδολογία. Η μεθοδολογία ποσοτικής βελτιστοποίησης των αλληλεπιδράσεων αυτών περιγράφεται ως εξής:

1. Καταγραφή υφιστάμενης κατάστασης και ανάλυση αυτής μέσα από αναλυτικές μεθόδους. Στο στάδιο αυτό αναγνωρίζεται η φύση του προβλήματος και οι περιορισμοί που υπάρχουν σε αυτό καθώς και ο στόχος που σκοπεύει η επιχείρηση να επιτύχει.
2. Δημιουργία προτύπου απεικόνισης του προβλήματος μέσα απο ποσοτικές μεθόδους. Χρήση μαθηματικών ή συστημικών μεθόδων γίνεται σε αυτό το στάδιο ανάλυσης.

3. Δημιουργία εργαλείων επίλυσης του προβλήματος, όπως αυτό διαμορφώθηκε, στο παραπάνω στάδιο, με αναδιομόρφωση υπαρχόντων διαδικασιών ή κατασκευή νέων. Στο στάδιο αυτό αναζητείται το εργαλείο με το οποίο θα δημιουργηθεί μια εφικτή ή βέλτιστη λύση.
4. Εφαρμογή των μεθόδων επίλυσης που επιλέχθηκαν στο προαναφερθέν στάδιο, μέσω υπολογιστικής πλατφόρμας. Σε αυτό το στάδιο επιλέγεται και αν θα κατασκευαστεί το λογισμικό επίλυσης, η πιθανή αγορά έτοιμου πακέτου και γίνεται εφαρμογή σύμφωνα με τις υπάρχουσες τεχνολογικές εξελίξεις και υποδομές της εταιρείας.
5. Δημιουργία διαύλου επικοινωνίας μεταξύ των πληροφοριακών συστημάτων που εμπεριέχουν τις πληροφορίες επίλυσης, με το ανθρώπινο δυναμικό της εταιρείας. Το κομμάτι αυτό παρουσιάζει ιδιαίτερη σημασία, αφού αποτελεί τη σύνδεση των δεδομένων και των λύσεων του προβλήματος με τον πραγματικό κόσμο και προσφέρει την πληροφορία που χρειάζεται για την σχεδίαση τόσο της στρατηγικής όσο και της επιχειριακής λειτουργίας της εφοδιαστικής αλυσίδας.

Κεφάλαιο 2

Εισαγωγή στους αλγορίθμους

Αφού αποτυπώθηκε η εικόνα των γραμμών ανεφοδιασμών θα αναδειχθεί η ανάγκη των βελτιστοποιητικών αλγορίθμων. Η εύρεση της βέλτιστης λύσης θα μπορούσε να χαρακτηριστεί ως το Άγιο Δισκοπότηρο της βελτιστοποίησης, ο θησαυρός στον οποίο στοχεύει να βρεί κάποιος. Έτσι, για χάρην παραδείγματος, έστω ότι γίνεται μια προσπάθεια να βρούμε το «θησαυρό» σε ένα λοφώδη τοπίο, εντός μιας συγκεκριμένης είτε χρονικής είτε χρηματικής προθεσμίας. Σε μια ακραία περίπτωση, ας υποθέσουμε ότι ο εξερευνητής είναι με δεμένα μάτια χωρίς καμία καθοδήγηση, άρα η διαδικασία αναζήτησης είναι ουσιαστικά μια καθαρή τυχαία αναζήτησης, η οποία συνήθως δεν είναι αποτελεσματική καθώς τα περιθώρια που έχουν δωθεί απο την προθεσμία δεν επαρκούν για να βρεθεί μέσα απο αυτή τη διαδικασία. Στο άλλο άκρο, έστω οτι είναι γνωστό πως ο «θησαυρός» τοποθετείται στην ψηλότερη κορυφή μιας εκ των προτέρων γνωστής περιοχής, τότε ο εξερευνητής θα σπεύσει απευθείας να ανέβει στον πιο απότομο βράχο και θα προσπαθήσει να καταλήξουν στην ψηλότερη αυτή κορυφή με το συντομότερο δυνατό τρόπο, και διαδικασία που στην επιστήμη υπολογιστών αντιστοιχεί την κλασσική τεχνική επονομαζόμενη αναρρίχηση λόφων (Hill-Climbing Technique) (Russell, 2003). Στις περισσότερες περιπτώσεις, η αναζήτηση της βέλτιστης λύσης είναι μια διαδικασία που βρίσκεται μεταξύ αυτών των δύο άκρων. Εν ολίγοις, δεν είναι πλήρως τυφλός ο εξερευνητής/επιλύτης , αλλά δεν γνωρίζει και εκ των προτέρων που να ψάξει για τη λύση. Όπως είναι και προφανές, είναι ουτοπικό και εξαντλητικό να πραγματοποιηθεί αναζήτηση σε κάθε τετραγωνικό κάποιας εξαιρετικά μεγάλης λοφώδους περιοχής, έτσι ώστε να βρεθεί ο «θησαυρός».

Το πιο πιθανό σενάριο είναι ότι θα πραγματοποιηθεί ένας τυχαίος περίπατος, ενώ ταυτόχρονα αναζητούνται και μερικά στοιχεία για το που μπορεί να βρίσκεται ο «θησαυρός». Έτσι, γίνεται η αναζήτηση αρχικώς σε κάποιο χώρο σχεδόν τυχαία, στη συνέχεια να προχωρήσουμε σε μια άλλη πιθανή θέση, μετά σε μια άλλη και ούτω καθεξής. Τέτοιος τυχαίος περίπατος είναι ένα κύριο χαρακτηριστικό των σύγχρονων αλγορίθμων αναζήτησης. Προφανώς, μπορεί να γίνει αυτή η αναζήτηση από μόνο έναν εξερευνητή, έτσι ώστε όλη η διαδρομή είναι μια αναζήτηση με βάση την πορεία που ακολούθησε, όπως πραγματοποιείται στη διαδικασία της προσομοιωμένης ανόπτησης (Kirkpatrick, Gelatt, & Vecchi, 1983). Εναλλακτικά, μπορούμε να ζητήσουμε απο μια ομάδα των εξερευνητών/επιλυτών να κάνουν την αναζήτηση και να μοιραστούν τις πληροφορίες για την περιοχή αναζήτησης, και αυτό το σενάριο χρησιμοποιεί τη λεγόμενη νοημοσύνη σμήνους, και σε αυτήν την κατηγορία αντιστοιχεί η βελτιστοποίηση σμήνους σωματιδίων, όπως και η βελτιστοποιήση αποικίας μυρμηγκιών που θα παρουσιαστεί αργότερα λεπτομερώς. Αν ο «θησαυρός» είναι πραγματικά σημαντικός αλλά και μικρός σε μέγεθος, και η περιοχή αναζήτησης είναι εξαιρετικά μεγάλη, η διαδικασία αναζήτησης θα πάρει πολύ μεγάλο χρονικό διάστημα.

Εάν δεν υπάρχει χρονικό ή χρηματικό όριο και αν η περιοχή προς αναζήτηση είναι εύκολα προσπελάσιμη, είναι θεωρητικά δυνατόν να βρεθεί ο απόλυτος «θησαυρός» (η καθολικά βέλτιστη λύση).

Προφανώς, η στρατηγική αναζήτησης μπορεί να βελτιωθεί περισσότερο. Μερικοί αναζητητές/επιλύτες είναι καλύτεροι από κάποιους άλλους. Μπορούμε να κρατήσουμε μόνο τους καλύτερους αναζητητές και να προσληφθούν νέοι, σε μια διαδικασία που είναι κάπως παρόμοια με τους γενετικούς αλγόριθμους ή τους εξελικτικούς αλγορίθμους όπου οι πράκτορες αναζήτησης βελτίωνονται σε κάθε επόμενο κύκλο αναζήτησης. Στην πραγματικότητα, όπως θα δειχθεί σε όλους σχεδόν τους σύγχρονους μεθευρετικούς αλγόριθμους, γίνεται προσπάθεια να χρησιμοποιήθούν οι καλύτερες λύσεις ή πράκτορες και να τυχαιοποιηθούν (ή να αντικατασταθούν) οι όχι και τόσο καλές λύσεις, καθώς γίνεται αξιολόγηση των ικανοτήτων/καταλληλότητας του κάθε πράκτορα (fitness), σε συνδυασμό με μια ιστορική αναδρομή των αποτελεσμάτων του όλου συστήματος (χρήση μνήμης). Με μια τέτοια ισορροπία, στοχέυουμε στο να γίνεται καλύτερη σχεδίαση και να υπάρχει βελτιωμένη απόδοση των αλγορίθμων βελτιστοποίησης.

Η γενικότερη ταξινόμηση των αλγορίθμων βελτιστοποίησης μπορεί να πραγματοποιηθεί με πολλούς τρόπους. Ένας απλός τρόπος είναι να εξετστεί η φύση του αλγόριθμου, και αυτό διαιρεί τους αλγόριθμους σε δύο κατηγορίες: ντετερμινιστικοί αλγόριθμοι, και στοχαστικοί αλγόριθμοι. Ντετερμινιστικοί, ή αιτιοκρατικοί αλγόριθμοι είναι εκείνοι που ακολουθούν μια αυστηρή και πρότερα καθορισμένη διαδικασία, και η διαδρομή των τελικών τιμών, των μεταβλητών σχεδίασης της διαδικασίας καθώς και οι διαδικασίες που την διέπουν είναι επαναλμβανόμενες και αυτούσια αναπαραγωγίσιμες. Για παράδειγμα, η αναρρίχηση λόφων είναι ένας ντετερμινιστικός αλγόριθμος, και για αν ξεκινήσουμε απο συγεκριμένο σημείο εκκίνησης, θα πραγματοποιηθεί ακριβώς η ίδια διαδικασία, θα παρουσιαστούν οι ίδιες τιμές σε κάθε βήμα σε οποιαδήποτε στιγμή κι αν εκτελεστεί το πρόγραμμα που τον εμπεριέχει.

Από την άλλ , οι στοχαστικοί αλγόριθμοι, όπως προδίδει και η ονομασία τους εμπεριέχουν πάντα κάποιο βαθμό τυχαιότητας. Οι γενετικοί αλγόριθμοι είναι ένα καλό παράδειγμα, οι λύσεις ή τα μέλη του πληθυσμού (strings) θα είναι διαφορετικά κάθε φορά που εκτελείται το πρόγραμμα του αλγορίθμου, δεδομένου ότι οι αλγόριθμοι αυτοί χρησιμοποιούν κάποιους ψευδο-τυχαίους αριθμούς, αν και στο τέλος τα αποτελέσματα δεν μπορούν να έχουν μεγάλη διαφορά, αλλά τα μονοπάτια που ακολουθεί το κάθε ατόμο δεν είναι ακριβώς επαναλαμβανόμενα.

Επιπλέον, υπάρχει ένας τρίτος τύπος αλγορίθμου που είναι ένα μείγμα, ή ένα υβρίδιο, των ντετερμινιστικών και στοχαστικών αλγορίθμων. Για παράδειγμα, η αναρρίχηση λόφων με μια τυχαία επανεκκίνηση του αλγορίθμου είναι ένα τυπικό παράδειγμα. Η βασική ιδέα είναι να χρησιμοποιηθεί ένας ντετερμινιστικός αλγόριθμος, αλλά οι εκκινήσεις του να πραγματοποιούνται απο διαφορετικά σημεία κάθε φορά, σημεία που θα επιλέγονται με τυχαίο σχετικά τρόπο. Αυτό έχει ορισμένα πλεονεκτήματα σε σχέση

με μια απλή τεχνική αναρρίχησης λόφων, η οποία μπορεί να είναι υποπέσει σε ένα τοπικό μέγιστο/ελάχιστο (ανάλογα τη φύση του προβλήματος) και λόγω της αιτιοκρατικής φύσης να μην μπορεί να ξεφύγει απο αυτό. Ωστόσο, δεδομένου ότι υπάρχει μια τυχαία συνιστώσα, μια τυχαία πλευρά, στους υβριδικούς αλγόριθμους, συχνά τους κατατάσσουν ως ένα είδος στοχαστικού αλγορίθμου στην βιβλιογραφία βελτιστοποίησης.

Η πλειοψηφία των συμβατικών ή κλασικών αλγορίθμων είναι ντετερμινιστικής φύσης. Για παράδειγμα, η γνωστή μέθοδος simplex του γραμμικού προγραμματισμού είναι καθαρά ντετερμινιστική μέθοδος εξεύρεσης καθολικά βέλτιστης λύσης. Μερικοί ντετερμινιστικοί αλγόριθμοι βελτιστοποίησης χρησιμοποιούν πληροφορίες για την κλίση, κι έτσι αποκαλούνται αλγόριθμοι βασιζόμενοι στην κλίση (gradient based algorithms). Για παράδειγμα, ο γνωστός Newton-Raphson αλγόριθμος βασιζεται στην κλίση της λύσης εκείνου του κύκλου επίλυσης, καθώς χρησιμοποιεί τις τιμές της συνάρτησης και τις παραγώγους για να προσφέρει αποτελέσματα. Ο συγκεκριμένος αλγόριθμος λειτουργεί εξαιρετικά καλά σε ομαλά, μοναδικού τοπικού ελαχίστου/μεγίστου προβλήματα. Ωστόσο, σε περίπτωση ύπαρξης κάποιας ασυνέχειας στην αντικειμενική συνάρτηση, ο συγκεκριμένος αλγόριθμος παύει να λειτουργεί καλά. Στην περίπτωση αυτή, ένας αλγόριθμος που δεν βασίζεται στην κλίση κάποιας συνάρτησης προτιμάται. Αλγόριθμοι που δεν βασίζονται στην κλίση της συνάρτησης (gradient-free algorithms) δεν χρησιμοποιούν καμιά παράγωγο σε κανένα σημείο λειτουργίας τους, αλλά στηρίζουν τη λειτουργία τους μόνο στις τιμές που λαμβάνονται κατα την εκτέλεση. Το μοτίβο αναζήτησης Hooke-Jeeves καθώς και ο simplex αλγόριθμος των Nelder-Mead είναι παραδείγματα αλγορίθμων που δεν στηρίζονται στην κλιση της αντικειμενικής συνάρτησης.

Οι στοχαστικοί αλγόριθμοι, σε γενικές γραμμές, κατατάσσονται μεταξύ δύο ειδών: τους ευρετικούς και τους μεθευρετικούς, αν και η διαφορά τους είναι μικρή. Προσπαθώντας να εξηγήσει κανείς την έννοια του ευρετικού θα μπορούσε να πει κανείς οτι σημαίνει «να βρεθεί» ή «να ανακαλύφθεί» μια λύση μέσα από μια διαδικασία δοκιμής και λάθους (trial and error). Ποιοτικές λύσεις για ένα αρκετά δύσκολο πρόβλημα βελτιστοποίησης είναι πιθανόν να βρεθούν σε ένα εύλογο χρονικό διάστημα, αλλά δεν υπάρχει καμία εγγύηση ότι θα βρεθούν και οι βέλτιστες λύσεις με τη χρήση των ευρετικών. Σε αυτό που βασίζεται ο χρήστης τέτοιων αλγορίθμων είναι ότι αυτοί οι αλγόριθμοι λειτουργούν τις περισσότερες φορές, όμως οι αλγόριθμοι αυτοί δεν λειτουργούν πάντα εξίσου καλά και ποιοτικά. Αυτό δεν είναι τόσο μεγάλο πρόβλημα, όταν δεν χρειάζονται οι καθολικά βέλτιστες ή κάποιες βέλτιστες λύσεις, αλλά κάποιες σχετικά καλές λύσεις που είναι εύκολο στο να δημιουργηθούν. Η περαιτέρω ανάπτυξη της συγκεκριμένης κατηγορίας αλγορίθμων επέφερε την κατηφορία των λεγόμενων μεθευρετικών αλγορίθμων. Εδώ το πρόθεμα μεθ- (προερχόμενο απο τη λέξη μετά) σημαίνει «πέρα από» ή «σε ανώτερο επίπεδο», και γενικά προδίδει οτι ο συγκεκριμένος τύπος αλγορίθμων αποδίδει καλύτερα από τους απλούς ευρετικούς. Επιπλέον, όλοι οι

μεθευρετικοι αλγόριθμοι χρησιμοποιούν ορισμένα επίπεδα τυχαιότητας και τοπικής αναζήτησης. Αξίζει να σημειωθεί ότι δεν έχει συμφωνηθεί ξεκάθαρος ορισμός στη βιβλιογραφία αναφορικά με τους ευρετικούς και μεθευρετικούς αλγορίθμους. Ωστόσο, η πρόσφατη τάση τείνει να αναφέρονται όλοι οι στοχαστικοί αλγόριθμοι που εμπεριέχουν την έννοια της τυχαιότητας και της τοπικής αναζήτησης ως μεθευρετικοι. Στην εργασία αυτή θα γίνει χρήση αυτής της παραδοχής. Η τυχαιότητα που εμπεριέχεται στους μεθευρετικούς αλγορίθμους παρέχει έναν καλό τρόπο για να ξεφύγει κανείς πέραν από την τοπική αναζήτηση προς την αναζήτηση ενός καθολικού βέλτιστου. Ως εκ τούτου, σχεδόν όλοι οι μεθευρετικοί αλγόριθμοι κρίνονται κατάλληλοι για να βρεθεί το καθολικό βέλτιστο ενός προβλήματος/συνάρτησης.

Οι ευρετικοί αλγόριθμοι είναι ένας τρόπος, μέσω μιας διαδικασίας δοκιμής και σφάλματος, να παραχθούν αποδεκτές λύσεις για την ένα σύνθετο πρόβλημα σε λογικά αποδεκτό χρόνο (Judea, 1984). Η πολυπλοκότητα των προς μελέτη προβλημάτων καθιστά αδύνατη την αναζήτηση κάθε δυνατής λύσης ή συνδυασμού λύσεων, και ο στόχος είναι να βρούμε καλές και εφικτές λύσεις εντός ενός αποδεκτού χρονοδιαγράμματος. Δεν υπάρχει καμία εγγύηση ότι μπορούν να βρεθούν οι καθολικά καλύτερες λύσεις, και δεν υπάρχει εκ των προτέρων γνώση για το αν ένας αλγόριθμος λειτουργήσει καλά για κάποιο συγκεκριμένο πρόβλμα. Η ιδέα είναι να έχουμε έναν αποτελεσματικό και πρακτικό αλγόριθμο που θα λειτουργεί σε αρκετά διαφορετικά προβλήματα και θα είναι σε θέση να παράγει καλής ποιότητας λύσεις. Μεταξύ των λύσεων που βρέθηκαν κάποιες θα έχουν καλή ποιότητα και μια απο αυτές τις λύσεις θα είναι κοντά στη βέλτιστη λύση, χωρίς να υπάρχει όμως εγγύηση για κάτι τέτοιο.

Δύο είναι τα βασικά στοιχεία του κάθε μεθευρετικού αλγορίθμου: εντατικοποίηση και διαφοροποίηση, ή εκμετάλλευση και εξερεύνηση. Διαφοροποιήση σημαίνει να παράγονται διαφορετικές λύσεις, έτσι ώστε να εξερευνάται μεγάλο πεδιο απο το χώρο αναζήτησης, σε καθολική κλίμακα, ενώ η εντατικοποίηση σημαίνει να επικεντρωθεί η αναζήτηση σε ένα μικρό σημείο του χώρου, καθώς γίνεται αξιοποίηση των πληροφοριών που βρίσκεται σε μια παρούσα καλή λύση για περιοχή αυτή. Αυτό πραγματοποιείται σε συνδυασμό με την διαρκή επιλογή των καλύτερων λύσεων. Η επιλογή των καλύτερων λύσεων εξασφαλίζει ότι οι λύσεις θα συγκλίνουν προς τη βέλτιστη, ενώ η διαφοροποίηση μέσω της τυχαιοτητας κάνει τον αλγόριθμο να αποφεύγει τον εγκλωβισμο της λύσης του προβλήματος σε τοπικά βέλτιστα και, την ίδια στιγμή, αυξάνει την ποικιλία των λύσεων. Ο καλός συνδυασμός αυτών των δύο γενικών εννοιών εξασφαλίζει οτι συνήθως το καθολικά βέλτιστο είναι εφικτό. Οι μεθευρετικοί αλγόριθμοι μπορούν να ταξινομηθούν με πολλούς τρόπους. Ένας τρόπος είναι να τους κατατάξει κανείς ως εξής: με βάση τον πληθυσμό και την τροχιά πάνω στην οποία βασίζονται. Για παράδειγμα, γενετικοί αλγόριθμοι είναι βασισμένοι στον πληθυσμό, δεδομένου ότι χρησιμοποιούν μια σειρά από strings, όπως συμβαίνει και με την βελτιστοποίηση σμήνους σωματιδίων (particle swarm optimization,PSO), το οποίο χρησιμοποιεί πολλαπλούς πράκτορες ή σωματίδια.

Από την άλλη, η προσομοιωμένη ανόπτηση (simulated annealing) χρησιμοποιεί ένα μεμονωμένο παράγοντα ή λύση που κινείται μέσα στον χώρο αναζήτησης με τμηματικό τρόπο. Μια καλύτερη κίνηση ή λύση είναι πάντα αποδεκτή, ενώ μια όχι και τόσο καλή κίνηση μπορεί να γίνει δεκτή με μια ορισμένη πιθανότητα. Τα βήματα ή οι κινήσεις που πραγματοποιούνται, ακολουθούν μια συγκεκριμένη τροχιά στο χώρο της αναζήτησης, με μη μηδενική την πιθανότητα ότι αυτή η τροχιά μπορεί να φτάσει στην καθολικά βέλτιστη λύση.

Ιστορική Αναδρομή

Στο παρελθόν, ειδικά στις πρώιμες περιόδους της ανθρώπινης ιστορίας, η προσέγγιση των ανθρώπων στην επίλυση προβλημάτων υπήρξε πάντα ευρετική ή μεθευρετικές – μέσω διαδικασιών δοκιμής και σφάλματος. Πολλές σημαντικές ανακαλύψεις έγιναν μέσω σκέψης εκτός του συνηθισμένου (out of the box thinking), και συχνά κατα λάθος, όπως ακριβώς συμβαίνει σε διαδικασίες ευρεστικών αλγορίθμων. Η στιγμή που ο Αρχιμήδης αναφώνησε το γνωστό «Εύρηκα» είναι η πιο λαμπρή στιγή των ευρετικών διαδικασιών αναζήτησης λύσεων. Στην πραγματικότητα, η καθημερινή μας μαθησιακή εμπειρία είναι κατα κόρον ευρετική διεργασία. Παρά την πανταχού παρούσα φύση των μεθευρετικών διαδικασιών, οι συγκεκριμένες διεργασίες ως επιστημονική μέθοδος για την επίλυση προβλημάτων είναι πράγματι ένα σύγχρονο φαινόμενο, αν και είναι δύσκολο να εντοπίσει κανείς ακριβώς ποτε χρησιμοποιήθηκε για πρώτη φορά κάποια μεθευρετική μέθοδος (Schrijver, 2005).

Ο Alan Turing ήταν πιθανότατα ο πρώτος ερευνητής που χρησιμοποιεί ευρετικούς αλγορίθμους κατά τη διάρκεια του Β' Παγκοσμίου Πολέμου όταν χρειάστηκε το σπάσιμο των κρπυτογραφημένων κωδικών επικοινωνίας της Γερμανίας με τη χρήση αλγορίθμων αποκρυπτογράφησης που αφορούσαν τον κώδικα Enigma, έρευνα που έλαβε χώρο στο Bletchley Park. Ο Turing αποκάλεσε τη μέθοδο του «ευρετική αναζήτηση», και όπως ήταν αναμενόμενο λειτουργούσε στην πλειοψηφία των περιπτώσεων, αλλά δεν υπήρχε καμία εγγύηση οτι ήταν δυνατό να βρεθεί η σωστή λύση, αλλά σαν γενικότερο πλαίσιο ήταν μια τεράστια επιτυχία (Turing & Copeland, 2004). Το 1945, ο Turing είχε προσληφθεί απο το Εθνικό Εργαστήριο Φυσικής (National Physical Laboratory,NPL), (Turing, 1948), όπου καταθέτει το σχέδιό του για την αυτόματη μηχανή υπολογισμών (Automatic Computing Engine). Σε μια έκθεση για το NPL σχετικά με τα ευφυή μηχανήματα το 1948, περιέγραψε τις καινοτόμες ιδέες του περί τεχνητής νοημοσύνης και της μηχανικής μάθησης, καθώς και για τα νευρωνικά δίκτυα και τους εξελικτικούς αλγόριθμους (Siegelmann & Sontag, 1991).

Οι δεκαετίες του 1960 και 1970 ήταν οι δύο σημαντικές για την ανάπτυξη των εξελικτικών αλγορίθμων. Αρχικά, ο John Holland και οι συνεργάτες του στο Πανεπιστήμιο του Michigan ανέπτυξαν τους γενετικούς αλγόριθμους στη δεκαετία του 1960 και του 1970 (Holland, 1975). Ήδη από το 1962, ο Holland μελέτησε

το προσαρμοστικό σύστημα και ήταν ο πρώτος που χρησιμοποιεί crossover (διασταυρωση) και χειρισμούς των ανασυνδυασμών για τη μοντελοποίηση ενός τέτοιου συστήματος. Το 1975, ο De Jong παρουσιάζει τη σημαντική διατριβή που δείχνει τις δυνατότητες και τη δύναμη των γενετικών αλγορίθμων για ένα ευρύ φάσμα των αντικειμενικών συναρτήσεων (De Jong, 1975).

Στην ουσία, ένας γενετικός αλγόριθμος (Genetic Algorithm) είναι μια μέθοδος βασιζόμενη στις γενικές αποδοχές της δαρβινικής θεωρίας της εξέλιξης και της φυσικής επιλογής των βιολογικών συστημάτων και που πραγματοποιείται μεσα απο βασικούς μαθηματικούς τελεστές: το crossover ή ανασυνδυασμό λύσεων, τη μετάλλαξη τους, την καταλληλότητά τους (fitness), και την επιλογή της ισχυρότερης λύσης. Από τότε, οι γενετικοι αλγόριθμοι έχουν μετατραπεί σε τόσο μεγάλη επιτυχία για την επίλυση ενός ευρέος φάσματος προβλημάτων βελτιστοποίησης, ενώ παράλληλα έχουν δημοσιευθεί αρκετές χιλιάδες ερευνητικά άρθρα και εκατοντάδες βιβλία που γράφτηκαν για τους γεννετικούς και παραλλαγές τους. Μερικά στατιστικά στοιχεία δείχνουν ότι η συντριπτική πλειοψηφία των μεγάλων εταιρειών χρησιμοποιούν τώρα τους γεννετικούς αλγορίθμους για την επίλυση δύσκολων προβλημάτων συνδυαστικής βελτιστοποίησης όπως ο σχεδιασμός παραγωγής, πρόβλεψης και προγραμματισμού.

Κατά το το 1963, οι Ingo Rechenberg και Hans-Paul Schwefel απο το Πολυτεχνείο του Βερολίνου, ανέπτυξαν μια τεχνική για επίλυση προβλημάτων βελτιστοποίησης στην αεροδυναμική μηχανική, που ονομάζεται εξελικτική στρατηγική. Αργότερα, ο Peter Bienert συνεισέφερε στην συγκεκριμένη έρευνα και κατασκευάστηκε ένας αυτόματος πειραματιστής που έκανε χρήση απλών κανόνων μετάλλαξης και επιλογής. Δεν υπήρχε crossover σε αυτή την τεχνική, μόνο η μετάλλαξη, που χρησιμοποιήθηκε για την παραγωγή ενός απογόνου και η βελτιωμένη λύση διατηρείται σε κάθε γενειά (Bienert, Rechenberg, & Schwefel, 1966). Η διαδικασία αυτή ουσιαστικά ήταν σαν μια διαδικασία αναρρίχησης λόφων με ενσωματωμένη μια διαδικασία τυχαιότητας (History of optimization,, n.d.).

Ήδη από το 1960, Lawrence J. Fogel σκόπευε να χρησιμοποιήσει προσομοίωση της εξέλιξης ως μια διαδικασία μάθησης, ως ένα εργαλείο για τη μελέτη της τεχνητής νοημοσύνης. Στη συνέχεια, το 1966, οι Fogel,Owen και Walsh, ανέπτυξαν μια εξελικτική τεχνική προγραμματισμού όπου οι λύσεις εκπροσωπούνταν ως πεπερασμένης κατάστασης μηχανές και κάθε μία από αυτές τις μηχανές μεταλλάσσοντας τυχαία (Fogel, Owens, & Walsh, 1966). Οι παραπάνω καινοτόμες ιδέες και μέθοδοι έχουν εξελιχθεί σε ένα κατά πολύ ευρύτερο επιστημονικό τομέα, που ονομάζεται τομέας εξελικτικών αλγορίθμων ή / και εξελικτικοί αλγόριθμοι. Παρά το γεγονός ότι η εστίασή σε αυτή τη διατριβή είναι ο μεθευρετικός αλγόριθμος αποικίας μυρμηγκιών, άλλοι αλγόριθμοι μπορεί να θεωρηθούν ως μια ευρετική τεχνική βελτιστοποίησης. Αυτό περιλαμβάνει τεχνητά νευρωνικά δίκτυα (Neural Networks), μηχανές διανυσμάτων υποστήριξης (Support Vector Machines) και πολλές άλλες τεχνικές μηχανικής μάθησης. Πράγματι, όλες

αυτές οι μέθοδοι σκοπεύουν στο να ελαχιστοποιήσουν το κόστος μαθαίνοντας απο τα σφάλματα καθώς και απο την πρόβλεψη σφάλματών μέσω επαναληπτικής εξέτασής τους.

Τεχνητά νευρωνικά δίκτυα χρησιμοποιούνται τώρα συνήθως σε πολλές εφαρμογές. Το 1943, W. McCulloch και Pitts W. πρότειναν τους τεχνητούς νευρώνες, σαν απλές μονάδες επεξεργασίας των πληροφοριών (McCulloch & Pitts, 1943). Η έννοια του νευρωνικού δικτύου πιθανώς προτάθηκε πρώτη φορά από τον Alan Turing το 1948 έκθεσή του NPL, σχετικά με «Ευφυείς μηχανές». Σημαντικές εξελίξεις πραγματοποιήθηκαν από το 1940 και 1950 καθώς και τη δεκαετία του 1990. Η μηχανές διανυσμάτων υποστήριξης ως μια τεχνική ταξινόμησης μπορεί να χρονολογηθεί ως προηγούμενη εργασία από τον Vapnik το 1963 με τους γραμμικούς ταξινομητές (Vapnik, 2013), καθώς και η μη γραμμική ταξινόμηση με τις τεχνικές πυρήνα που αναπτύχθηκαν από τον ίδιο και τους συνεργάτες του στη δεκαετία του 1990 (Vapnik, Golowich, & Smola, 1997).

Οι δύο δεκαετίες του 1980 και του 1990 ήταν η πιο σημαντική περίοδος για το πεδίο των μεθευρετικών αλγορίθμων. Το επόμενο μεγάλο βήμα είναι η ανάπτυξη της προσομοιωμένης ανόπτησης (Simulated Annealing) το 1983, μια τεχνική βελτιστοποίησης εμπνευσμένη από τη διαδικασία ανόπτησης των μετάλλων (Kirkpatrick, Gelatt, & Vecchi, 1983). Ουσιαστικά, είναι μια διαδικασία που βασίζεται σε μια μια τροχιά, όπου ο αλγόριθμος αναζήτησης ξεκινα με μια αρχική λύση, όπως προέρχεται απο κάποια εικασία, όπου το σύστημα βρίσκεται σε υψηλή θερμοκρασία, και σταδιακά ψύχεται. Μια κίνηση ή νέα λύση γίνεται δεκτή εάν είναι καλύτερη απο την προηγούμενη αλλιώς, γίνεται δεκτό με μια πιθανότητα, η οποία καθιστά δυνατό για το σύστημα να ξεφύγει από οποιοδήποτε τοπικό βέλτιστο. Εν συνεχεία, θεωρείται ότι εάν το σύστημα ψύχεται αρκετά αργά, η καθολικά βέλτιστη λύση είναι δυνατόν να επιτευχθεί.

Η πρώτη χρήση μνήμης στους σύγχρονους μεθευρετικούς αλγορίθμους πιθανών πραγματοποιήθηκε στον Tabu αλγόριθμο, όπως αναπτύχθηκε απο τον Fred Γκλόβερ το 1986 (Glover & Laguna, 2013). Το 1992, ο Marco Dorigo τελείωσε τη διδακτορική του διατριβή για τη βελτιστοποίηση και τους αλγορίθμους εμπνευσμένους απο τη φύση, στην οποία περιέγραψε το καινοτόμο έργο του σχετικά με τη βελτιστοποίηση αποικίας μυρμηγκιών (Ant Colony Optimization) (Dorigo, Optimization, learning and natural algorithms, 1992). Αυτή η τεχνική αναζήτησης πραγματοποιήθηκε λόγω της έμπνευσης από την νοημοσύνη σμήνους, καθώς στην κοινωνια τα μυρμήγκια χρησιμοποιούν φερομόνη ως χημικό αγγελιοφόρος. Η όλη διαδικασία καθώς και η έννοια της νοημοσύνης σμήνους θα περιγραφεί παρακάτω. Στη συνέχεια, το 1992, John R. Koza του Πανεπιστημίου του Στάνφορντ δημοσίευσε μια πραγματεία περί γενετικού προγραμματισμού, η οποία έθεσε τα θεμέλια μιας ολόκληρης νέας περιοχή στη μηχανική μάθηση, μια επαναστατική θεωρία για τον προγραμματισμό ηλεκτρονικών υπολογιστών. Η βασική ιδέα είναι να χρησιμοποιηθούν οι αρχές της

γενετικής για να αναπαραχθούν προγράμματα ηλεκτρονικών υπολογιστών, έτσι ώστε να παράγονται σταδιακά καλύτερα προγράμματα για ένα δεδομένο είδος προβλήματων (Koza, 1992).

Λίγο αργότερα, το 1995, μια άλλη σημαντική πρόοδο αποτέλεσε η ανάπτυξη της βελτιστοποίησης σμήνους σωματιδίων (Particle Swarm Optimization) από τον αμερικανό κοινωνικό ψυχολόγο James Kennedy, και τον μηχανικό Russell Eberhart. Σε γεινικές γραμμές,ο PSO είναι μια ομάδα αλγορίθμων βελτιστοποίησης εμπνευσμένοι από νοημοσύνη σμήνους των ψαριών και πουλιών, ακόμα και την ανθρώπινη συμπεριφορά. Οι πολλαπλοί πράκτορες, που ονομάζονται σωματίδια, διαμοργώνουν ενα σμήνος γύρω από το χώρο αναζήτησης ξεκινώντας από κάποια αρχική τυχαία εικασία ή λύση. Το σμήνος μεταδίδει μεταξύ των μελών του τις τρέχουσες καλύτερες λύσεις και μοιράζεται το στιγμιαίο βέλτιστο, έτσι ώστε να επικεντρώνονται τα μέλη στις ποιοτικές λύσεις (Eberhart & Kennedy, 1995).Απο το σχεδιασμό του PSO, υπήρξαν περίπου 20 διαφορετικές παραλλαγές των τεχνικών βελτιστοποίησης σμήνους σωματιδίων, και έχουν εφαρμοστεί σε όλους σχεδόν τους τομείς των δύσκολων προβλημάτων βελτιστοποίησης. Υπάρχουν κάποιες ισχυρές ενδείξεις οτι οι PSO αλγόριθμοι είναι καλύτεροι από τους παραδοσιακούς αλγόριθμους αναζήτησης και ακόμη καλύτεροι από τους γενετικούς αλγορίθμους για πολλούς τύπους προβλημάτων, αν και αυτό δεν έχει ακόμη πλήρως αποδεικτεί.

Το 1996 και αργότερα το 1997, οι Storn και Price ανέπτυξαν τον εξελικτικό αλγόριθμο βασιζόμενο σε διανύσματα, που ονομάζεται διαφορική εξέλιξη (Differential Evolution), και ο αλγόριθμος αυτός αποδεικνύεται πιο αποτελεσματικός από τους τους γενετικούς αλγόριθμους σε πολλές εφαρμογές (Storn & Price, 1997).

Στις αρχές του 21^{ου} αιώνα, αναπτύχθηκε ο αλγόριθμος αναζήτησης αρμονίας (Harmony Search Algorithm), ο οποίος έχει εφαρμοστεί ευρέως για την επίλυση διαφόρων προβλημάτων βελτιστοποίησης όπως η διανομή νερού, μοντελοποίηση των μεταφορών και τον προγραμματισμό παραγωγής (Geem, Kim, & Logathan , 2001). (Geem et al., 2001)

Το 2004, οι Nakrani και Tovey πρότειναν τον αλγόριθμο βελτιστοποίησης μελισσών, για τη βελτιστοποίηση κέντρων διαδικτυακής φιλοξενίας (Nakrani, 2004). Εν συνεχεία, υπήρξε περαιτέρω ανάπτυξη των αλγορίθμων βασισμένων σε μέλισσες (Pham,, et al., 2011) και ανάπτυξη της αποικίας τεχνητών μελισσών (Artificial Bee Colony) από τον Karaboga το 2005 (Karaboga, 2005).

Το 2008, αναπτύχθηκε ο αλγόριθμος πυγολαμπίδας (Firefly Algorithm). Ακολούθησαν αρκετά ερευνητικά άρθρα σχετικά με τον αλγόριθμο πυγολαμπίδας, καθώς ο αλγόριθμος αυτός έχει προσελκύσει ένα ευρύ φάσμα επιστημόνων λόγω του ενδιαφέροντος που παρουσίαζε καθώς και τις δυνατότητες εφαρμογής του (Yang, 2009).

Το 2009, μια συνεργασία μεταξύ των πανεπιστημίων Cambridge και Raman College of Engineering, οδήγησε στην ανάπτυξη του αλγορίθμου αναζήτησης κούκου (Cuckoo Search), και έχει αποδειχθεί ότι ο συγκεκριμένος αλγόριθμος είναι πολύ πιο αποτελεσματικός από τους περισσότερους υπάρχοντες μεθευρετικών αλγορίθμων συμπεριλαμβανομένων και αυτών του σμήνους σωματιδίων (Yang & Deb, 2009). Όπως γίνεται εύκολα αντιληπό το πεδίο των μεθευρετικών παρουσιάζει μεγάλο πεδίο έρευνας.

Μετά απο μια γρήγορη ανασκόπηση των αλγορίθμων και της ιστορίας τους, γίνεται μια παρουσίαση της έννοιας της νοημοσύνης σμήνους, βασική για την κατανόηση του αλγόριθμου αποικίας μυρμηγκιών.

Νοημοσύνη Σμήνους (Swarm Intelligence)

Σμήνος είναι ένας μεγάλος αριθμός ομοιογενών, απλών πρακτόρων που αλληλεπιδρούν μεταξύ τους σε τοπικό επίπεδο, και με το περιβάλλον τους, χωρίς κεντρικό έλεγχο με στόχο να επιτευχθεί μια καθολικά θετική και άξια μελέτης, συμπεριφορά. Οι αλγόριθμοι σμήνους που έχουν αναδειχθεί πρόσφατα είναι μια οικογένεια αλγορίθμων εμπνευσμένοι από τη φύση, βάσιζόμενοι στον πληθυσμό πρακτόρων που είναι ικανοί να παράγουν χαμηλού κόστους, γρήγορες και ισχυρές λύσεις σε πολλά σύνθετα προβλήματα (Panigrahi, Yuhui , & Meng-Hiot, 2011) (Merkle & Blum, 2008) Νοημοσύνη Σμήνους (Swarm Intelligence, SI) μπορεί να οριστεί ως *ένας σχετικά νέος κλάδος της Τεχνητής Νοημοσύνης που χρησιμοποιείται για να μοντελοποιήσει τη συλλογική συμπεριφορά των κοινωνικών δομών που βρίσκονται στη φύση, όπως αποικίες μυρμηγκιών, οι μέλισσες και τα σμήνη των πουλιών*. Παρά το γεγονός ότι αυτοί οι πράκτορες (έντομα ή άτομα σμήνος) είναι σχετικά απλοί, με περιορισμένες δυνατότητες όταν βρίσκονται ξέχωρα απο την κοινωνία τους, σαν σμήνος αλληλεπιδρούν μαζί, βασιζόμενοι πάνω σε ορισμένα πρότυπα συμπεριφοράς για την επίτευξη συνεργατικών καθηκόντων που είναι αναγκαία για την επιβίωσή τους. Οι κοινωνικές αλληλεπιδράσεις μεταξύ των μελών του σμήνος μπορεί να είναι είτε άμεση είτε έμμεση (Belal, Gaber, El-Sayed, & Almojel, 2006). Παραδείγματος χάρη, άμεση αλληλεπίδραση είναι μέσω οπτικών ή ακουστικής επαφής, όπως ο χορός εύρεσης τροφής των μελισσών. Έμμεση αλληλεπίδραση συμβαίνει όταν ένα άτομο αλλάζει το περιβάλλον και τα άλλα άτομα που ανταποκρίνονται στο νέο περιβάλλον, όπως τα μονοπάτια φερομόνης όπως τα μυρμήγκια, που εναποθέτουν φερομόνη στο δρόμο τους προς αναζητήση πηγής τροφίμων. Αυτός ο έμμεσος τύπος αλληλεπίδρασης αναφέρεται ως στιγμεργία (stigmergy), πράγμα που ουσιαστικά σημαίνει επικοινωνία μέσω του περιβάλλοντος (Dorigo, Bonabeau, & Theraulaz, 2000). Η έννοια της στιγμεργίας θα αναλυθεί περαιτερώ παρακάτω. Η περιοχή της έρευνας που παρουσιάζονται σε αυτήν την εργασία επικεντρώνεται στη Νοημοσύνη Σμήνους. Πιο συγκεκριμένα, η παρούσα διατριβή εξετάζει ένα από τα πιο δημοφιλή μοντέλα της νοημοσύνη σμήνους εμπνευσμένο τη συμπεριφορά των μυρμηγκιών. Τις τελευταίες δεκαετίες, βιολόγοι και άλλοι φυσικοί επιστήμονες έχουν μελετήσει τις συμπεριφορές των κοινωνικών εντόμων, λόγω της εκπληκτικής απόδοσης αυτών των φυσικών

συστημάτων σμήνους. Στα τέλη της δεκαετίας του '80, οι επιστήμονες υπολογιστών μετέφεραν τις επιστημονικές γνώσεις αυτών των συστημάτων φυσικού σμήνος στον τομέα της Τεχνητής Νοημοσύνης. Το 1989, η έκφραση «Swarm Intelligence» εισήχθη για πρώτη φορά από τους Beni και Wang στα πλαίσια καθολικής βελτιστοποίησης, ως ένα σύνολο αλγορίθμων για τον έλεγχο τεχνητών σμηνών (Beni & Wang, 1989). Το 1991, η βελτιστοποιήση αποκίας μυρμηγκιών (Ant Colony Optimization ,ACO) (Dorigo, Maniezzo, & Colorni, 1991) (Dorigo, 1992) (Colorni, Dorigo, Maniezzo, & Trubian, 1994) εισήχθη από τον Dorigo και τους συνεργάτες του ως έναν νέο εμπνευσμένο από τη φύση μεθευρετικό αλγόριθμο, για την επίλυση προβλημάτων συνδυαστικής βελτιστοποίησης. Το 1995, η βελτιστοποίηση σμήνους σωματιδίων εισήχθη από τον Kennedy (Eberhart & Kennedy, 1995) (Kennedy, 1995), και προοριζόταν αρχικά για την προσομοίωση το πτηνών που δημιουργούν κοινωνική συμπεριφορά. Μέχρι τα τέλη της δεκαετίας του '90, αυτοί οι δύο πιο δημοφιλείς αλγόριθμοι νοημοσύνης σμήνους άρχισαν να πηγαίνουν πέρα από ένα καθαρά επιστημονικό ενδιαφέρον και να εισέρχονται στην σφαίρα των εφαρμογών πραγματικού κόσμου. Αξίζει να αναφερθεί εδώ ότι αργότερα, ακριβώς το 2005, προτάθηκε από τον Karabago ο αλγόριθμος τεχνητής αποκίας μελισσών (Artificial Bee Colony Algorithm, ABCA) ως νέο μέλος της οικογένειας των αλγορίθμων νοημοσύνη σμήνους (Karaboga, 2005) (Karaboga & Basturk, 2007). Η δυνατότητα μαθηματικής μοντελοποίησης των σμηνών οδήγησε στην υπολογιστική τους υλοποίηση και έτσι υπήρξε μια σταθερή αύξηση του αριθμού των ερευνητικών εργασιών που αναφέρουν την επιτυχή εφαρμογή των αλγορίθμων νοημοσύνης σμήνους σε διάφορες εργασίες βελτιστοποίησης καθώς και ερευνητικά προβλήματα. Οι αρχές της νοημοσύνης σμήνους έχουν εφαρμοστεί με επιτυχία σε μια ποικιλία προβλημάτων διαφόρων τομέων, συμπεριλαμβανομένων των προβλημάτων βελτιστοποίησης, της εξεύρεσης της βέλτιστης διαδρομής, τον προγραμματισμό παραγωγής και την αναγνώριση εικόνας, ήχου καθώς και στην ανάλυση δεδομένων (Engelbrecht, 2002) (Lim, Jain, & Dehuri, 2009). Υπολογιστική μοντελοποίηση των σμηνών έχει εφαρμοστεί ακόμη σε ένα ευρύ φάσμα διαφορετικών τομέων, συμπεριλαμβανομένων μηχανικής μάθησης (Das, Panigrahi, & Pattnaik, 2009), της βιοπληροφορικής και της ιατρικής (Das, Abraham, & Konar, 2008), καθώς και στα δυναμικά συστήματα και την επιχειρησιακή έρευνα (Parsopoulos, 2010). Τέλος έχουν ακόμη εφαρμοστεί και στον τομέα των οικονομικών και των επιχειρήσεων (Bonabeau & Meyer, 2001). Μέχρι σήμερα, υπάρχουν πολλά μοντέλα νοημοσύνη σμήνους που βασίζονται σε διαφορετικά συστήματα φυσικού σμήνους και έχουν προταθεί στη βιβλιογραφία, καθώς εφαρμόζονται με επιτυχία σε πολλές πρακτικές εφαρμογές της πραγματικής βιομηχανίας. Παραδείγματα μοντέλων νοημοσύνη σμήνους είναι: Ant Colony Optimization (Stützle, 2004), Particle Swarm Optimization (Eberhart & Kennedy, 1995), Artificial Bee Colony (Karaboga, TR06, 2005), Bacterial Foraging (Passino, 2002), Cat Swarm Optimization (Chu, Tsai, & Pan, 2006), Artificial Immune System (Bakhouya & Gaber, 2007), και Glowworm Swarm Optimization (Krishnanand, & Ghose, 2009). Αυτή την

εργασία, θα επικεντρωθεί στην χρήση ενός απο τα πιο δημοφιλή μοντέλα νοημοσύνης σμήνους, δηλαδή τον Αλγόριθμο Βελτιστοποιήσης Αποικίας Μυρμηγκιών.

Αλγόριθμος Βελτιστοποιήσης Αποικίας Μυρμηγκιών (ACO)

Ο ACO που προτάθηκε αρχικά από τον Marco Dorigo το 1992 στη διδακτορική διατριβή του, είναι ο αλγόριθμος που αναζητά βέλτιστη διαδρομή σε γράφημα βάσει της συμπεριφοράς που έχουν αναπτύξει τα μυρμήγκια κατά την αναζήτηση τροφής απο την αποικία τους. Παραλλαγές του συγκεκριμένου αλγορίθμου υπάρχουν, βασισμένες σε συμπεριφορές διαφόρων ειδών μυρμηγκιών και ο συγκεκριμένος αλγόριθμος έχει παρουσιάσει αρκετές εφαρμογές ιδιαίτερα στο πεδίο των προβλημάτων της βελτιστοποιήσης διαδρομών.

Στιγμεργία

Η έννοια της στιγμεργίας είναι πολύ βασική σε αυτόν τον αλγόριθμο, αφού έιναι ο τρόπος με τον οποίο τα μυρμηγκια-πράκτορες επικοινωνούν. Η έννοια της στιγμεργίας δείχνει οτι δύο άτομα αλληλεπιδρούν έμμεσα, όταν ένας από αυτούς τροποποιεί το περιβάλλον και η άλλη ανταποκρίνεται στο νέο περιβάλλον σε μεταγενέστερο χρόνο. Τα μυρμήγκια το πραγματοποιούν αυτο μέσω εναπόθεσης φερομόνης. Η φερομόνη αυτή είναι «ορατή» μόνο για τα μυρμήγκια που βρίσκονται σε στενή γειτνίαση. Επίσης τα περισσότερα μυρμήγκια ακολουθούν ένα μονοπάτι, αυτό με την περισσότερη φερομόνη. Όσο πιο ελκυστικό γίνεται το μονοπάτι, τόσο περισσότερο ακολουθείται και εμπεριέχει περισσότερη φερομόνη. Η μέθοδος χαρακτηρίζεται έτσι από ένα θετικό βρόχο ανάδρασης, όπου η πιθανότητα επιλογής μιας συγκεκριμένης διαδρομής αυξάνεται ανάλογα με το πόσες φορές το ίδιο μονοπάτι ήδη επιλέχθηκε .

Διαφορές Τεχνητών και Πραγματικών Μυρμηγκιών

Προφανώς υπάρχουν διακριτές διαφορές μεταξύ των πραγματικών μυρμηγκιών και των τεχνητών, παρά την παραδοχή οτι ο αλγόριθμος είναι εμπνευσμένος απο την ίδια τους τη φύση. Οι διαφορές και οι ομοιότητές του μπορούν να συνοψισθούν ως εξής:

Διαφορές

- Τα τεχνητά μυρμηγήγκια επιχειρούν μέσα σε ένα διακριτό κόσμο με πεπερασμένα στοιχεία,αντίθετα απο τα πραγματικά που έχουν να βελτιστοποιήσουν διαδρομές σε ένα σχεδόν αχανές περιβάλλον.
- Τα τεχνητά μυρμηγκια περιέχουν εσωτερική μνήμη που καταγράφει τις ήδη εκτελεσμένες τους κινήσεις, ενώ τα πραγματικά δεν έχουν τέτοια δυνατότητα σαν μονάδες.

- Τα τεχνητά μυρμήγκια δεν είναι εντελώς τυφλά αφού έχουν δυνατότητα να διαλέγουν επόμενους κόμβους ενώ τα πραγματικά δεν έχουν τη δυνατότητα αυτή.
- Τα ποσά της εναποτιθέμενης φερομόνης καθώς και του ρυθμού εξάτμισής της κάνουν την ποιότητα της λύσης να διαφέρει
- Τα τεχνητά μυρμήγκια έχουν τη δύνατότητα να κάνουν τοπική αναζήτηση για βελτίωση της λύσης που βρίσκουν, αντίθετα απο τα κανονικά που δεν έχουν προφανώς τέτοιες τεχνητές δυνατότητες

Ομοιότητες

- Και τεχνητά και πραγματικά μυρμήγκια δημιουργούν μια αποικία συνεργαζόμενων μυρμηγκιών/πρακτόρων
- Τεχηντά και πραγματικά μυρμήγκια στηρίζονται στο ίχνος φερομόνης και στην έννοια της στιγμεργίας για να επηρεάσουν το περιβάλλον τους και να βρούν λύση.
- Τεχνητά και πραγματικά μυρμήγκια κάνουν χρήση κάποιου πιθανοτικού κανόνα για να κινηθούν και αναπτύσσουν μια τοπικότητα αναφορικά με τη στρατηγική που θα ακολουθήσουν.
- Πραγματοποιείται μια τοπική τροποποίηση της κατάστασης, κάτι που προκαλείται από τα προηγοούμενα μυρμήγκια, είτε τεχνητα είτε πραγματικά.

Λειτουργία

Βασική ιδέα, όπως δείχθηκε στην εννοια της στιγμεργίας, είναι ότι τα μυρμηγκια αφήνουν, κατά την αναζήτηση τροφής, μια ουσία πίσω τους, τη φερομόνη. Η ουσία αυτή προσελκύει τα μυρμήγκια,έτσι όσο περισσότερη υπάρχει σε ένα μονοπάτι τόσο μεγαλύτερη η πιθανότητα να προσπελαστεί απο περισσότερα μυρμηγκια που παρομοίως αφήνουν πίσω τους φερομόνη. Λόγω της εξάτμισης της ουσίας αυτής, μονοπάτια μακρυνά τείνουν να χάνουν τη φερομόνη τους, κι έτσι να μην τα ξαναεπισκέπτεται κανένα άλλο μυρμήγκι, αντίθετα, κοντινά μονοπάτια διατηρούν τη φερομόνη τους περισσότερο, κι έτσι προσελκύουν περισσότερα μέλη της αποκίας, μέχρι να βρεθεί το μονοπάτι εκείνο το οποίο θα έχει τη μεγαλύτερη ποσότητα φερομόνης, δηλαδή θα είναι το πιο κοντινό.

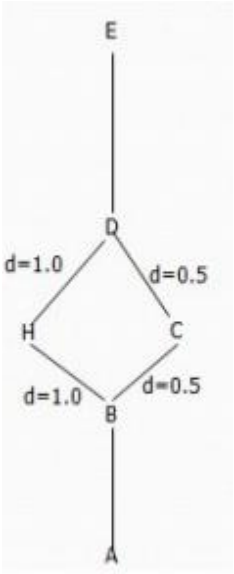
Η μετατροπή της φυσικής διαδικασίας εύρεσης της βέλτιστης διαδρομής σε αλγόριθμο υπολογισμού της σε κάποιο ψηφιακό μοντέλο, προϋποθέτει ορισμένες τροποποιήσεις. Τα μυρμήγκια που χρησιμοποιεί ο αλγόριθμος θα είναι απλουστευμένα σε σχέση με τα πραγματικά ωστόσο έχουν επιπλέον ιδιότητες, ανάλογα το πρόβλημα που επιλύουν. Οι αντιστοιχίες φυσικών ιδιοτήτων με τις ψηφιακές είναι οι εξής :

- Ο αλγόριθμος χρησιμοποιεί ένα πληθυσμό από μυρμήγκια όπου κάθε ένα από αυτά θα ακολουθήσει ένα μονοπάτι το οποίο είναι μια λύση του προβλήματος. Ο πληθυσμός των μυρμηγκιών είναι μια σημαντική παράμετρος του αλγορίθμου που επηρεάζει το χρόνο προσέγγισης της βέλτιστης λύσης.

- Η φερομόνη στον αλγόριθμο βελτιστοποίησης της Αποικίας Μυρμηγκιών αντιστοιχεί σε μια αριθμητική πληροφορία που θα εναποθέτουν τα μυρμήγκια και θα είναι προσβάσιμη από όλα τα υπόλοιπα, σχετικά με τη διαδρομή που ακολούθησαν. Αυτή η πληροφορία θα πρέπει να φθίνει στο χρόνο με τρόπο αντίστοιχο της εξάτμισης της φερομόνης στη φύση.
- Η επιλογή της διαδρομής που θα ακολουθήσουν δεν είναι καθαρά αιτιοκρατικής φύσης, αλλά είναι σε ένα βαθμό τυχαία. Τα ψηφιακά μυρμήγκια δεν έχουν μνήμη και επομένως η επιλογή της διαδρομής που θα ακολουθήσουν (στο αιτιοκρατικό της μέρος), εξαρτάται μόνο από την πληροφορία που λαμβάνει από στο συγκεκριμένο χώρο, τον συγκεκριμένο χρόνο και δεν γνωρίζει τι συνέβαινε σε παρελθόντα χρόνο ούτε τη πληροφορία υπάρχει στον υπόλοιπο χώρο.

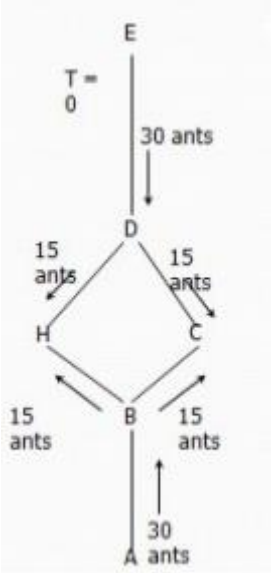
Χαρακτηριστικό παράδειγμα για να γίνει η όλη λειτουργία κατανοητή είναι:

Έστω ότι υπάρχει η παρακάτω κατάσταση για τα μυρμήγκια



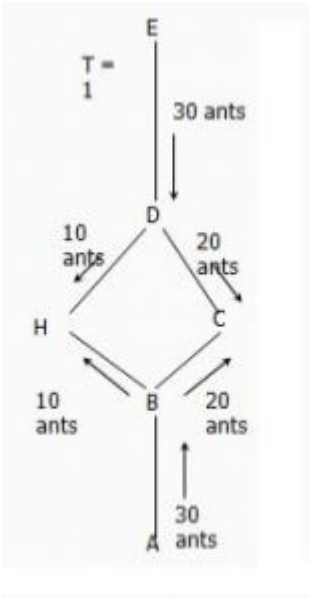
Εικόνα 1: Αρχικά Δεδομένα Προβλήματος

Όπου τα μυρμήγκια πρέπει να πάνε απο το A στο E μέσω των κόμβων B,C,D ή μέσω B,H,D. Με d δίνεται η απόσταση μεταξύ των κόμβων. Όπως είναι προφανές η βέλτιστη λύση είναι μέσω B,C,D. Αρχικά,την στιγμή T=0, για 30 μυρμήγκια έστω, θα έχουμε το εξής:



Εικόνα 2: Εικόνα του προβλήματος τη στιγμή T=0

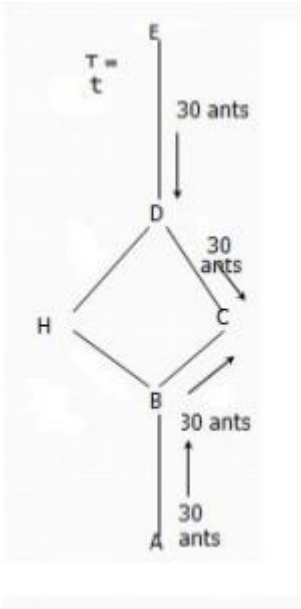
Βλέπουμε οτι απο τα 30 αυτά μυρμήγκια κινήθηκαν τα μισά μεταξύ ABCDE και τα άλλα μισά στο ABHDE. Αυτά απο την διαδρομή ABHDE έκαναν μεγαλύτερη διαδρομή, έτσι η φερομόνη τους εξατμίστηκε σε μεγαλύτερο βαθμό σε σχέση με αυτά της διαδρομής ABCDE. Τη στιγμή T=1, θα σχηματιστεί η παρακάτω εικόνα:



Εικόνα 3:Εικόνα προβλήματος τη στιγμή T=1

Αυτό συμβαίνει λόγω του μεγαλύτερου ποσού φερομόνης που έχει η διαδρομή ABCDE, κάτι που προσελκύει κι άλλα μυρμήγκια, που με τη σειρά τους θα αφήσουν περισσότερη φερομόνη. Έτσι, το

μονοπάτι γίνεται ελκυστικότερο σε κάθε επανάληψη, λόγω του μικρού του μήκους καθώς λόγω αυτού η φερομόνη του δεν έχει προλάβει να εξατμιστεί όπως στο ABHDE. Στο τέλος, με T=t κάποια στιγμή στο μέλλον, θα παρουσιαστει στο πρόβλημα η εξής εικόνα:



Εικόνα 4:Εικόνα προβλήματος το χρόνο T=t

Όπου κανένα μυρμήγκι δε θα πηγαίνει απο τη μακρυνή διαδρομή λόγω του ελάχιστου ή και μηδενικού ποσού φερομόνης που φέρει. Έτσι, επελέγει το καλύτερο και κοντυνότερο μονοπάτι για αυτή την περίπτωση.

Τα ψηφιακά μυρμήγκια εκτός από τα βασικά χαρακτηριστικά τους, μπορούν ανάλογα με την εφαρμογή να εφοδιαστούν με επιπλέον ιδιότητες, που θα επηρεάζουν την ταχύτητα σύγκλισης στη βέλτιστη λύση και γενικά την απόδοση του αλγορίθμου. Τα μυρμήγκια του αλγορίθμου μπορούν να αποκτήσουν μνήμη των προηγούμενων πράξεων τους και μονοπατιών που έχουν διασχίσει. Επίσης η ύπαρξη όρασης στο χώρο πέραν του σημείου που βρίσκονται, και ο υπολογισμός των αποστάσεων προτού διασχίσουν μια διαδρομή. Τα ψηφιακά μυρμήγκια έχουν την ικανότητα να εναποθέτουν τη φερομόνη κατά τη διάρκεια της περιπλάνησης τους ή μετά το πέρας αυτής. Ένα επιπλέον προσόν είναι η ικανότητα να χρησιμοποιούν αλγορίθμους ώστε να βελτιστοποιούν τοπικά τη διαδρομή. Το κεντρικό κομμάτι του αλγόριθμου ACO είναι ένα παραμετροποιημένο μοντέλο πιθανοτήτων (probabilistic model) που ονομάζεται μοντέλο φερομόνης. Αρχικά, κάθε ένα από τα μυρμήγκια ξεκινά από κάποια πόλη επιλεγμένη με τυχαίο τρόπο και έχει μνήμη η οποία αποθηκεύει πρώτα την πόλη-αφετηρία και καθώς το μυρμήγκι προχωρά προσθέτει σε αυτήν τη μνήμη τη λίστα των πόλεων που έχει επισκεφθεί μέχρι εκείνη τη στιγμή

Στην εφαρμοσμένη εκδοχή αυτής της διαδικασίας θα σχετιστεί η έννοια της φερομόνης με το τόξο που σχηματίζεται μεταξύ κόμβου i και j, και δείχνει την ελκυστικότητα του τόξου, με τον όρο τ_{ij}. Η αρχική φερομόνη για όλα τα μυρμήγκια μπορεί να υπολογιστεί σαν :

$$\tau_{ij} = \frac{m}{TC}, \tag{1}$$

όπου TC είναι το κόστος ενός πρώτου αρχικού κύκλου, που μπορεί να δημιουργηθεί απο μια ευρετική λύση, και με m παρουσιάζεται ο αριθμός των μυρμηγκιών.

Η ελκυστικότητα αυτή συνδυάζεται μια ευρετική πληροφορία, που αντιπροσωπεύει μια εκ των προτέρων πληροφορία για το πρόβλημα και η οποία παρέχεται από μία ανεξάρτητη πηγή από τα μυρμήγκια. Η πληροφορία αυτη συμβολίζεται με n_{ij} και δημιουργείται ως:

$$n_{ij} = \frac{1}{c_{ij}} \tag{40}$$

Ως c_{ij} συμβολίζεται το κόστος της διαδρομής απο τον κόμβο i στον j. Στην περίπτωση του πλανόδιου πωλητή αυτό που συμβαίνει είναι να αφήνουμε ένα μυρμήγκι σε μια τυχαία πόλη (ο συνολικός αριθμός των μυρμηγκιών δε μπορει να υπερβαίνει τον αριθμό των πόλεων), και να ολοκληρώνει ένα κύκλο. Για την επιλογή μετάβασης απο πόλη σε πόλη το κάθε μυρμηγκι χρησιμοποιεί ένα πιθανοτικό κανόνα ρουλέττας.

Η πιθανότητα μετάβασης απο μια πόλη i σε πόλη j βγαίνει απο τον τύπο:

$$p_{ij} = \frac{[\tau_{ij}]^{\alpha}[n_{ij}]^{\beta}}{\sum_{l=1}^{M}[\tau_{il}]^{\alpha}[n_{il}]^{\beta}} \tag{41}$$

όπου M είναι το σύνολο των πόλεων και α,β ο βαθμός που επηρρεάζει η φερομόνη και η ευρετική πληροφορία την λύση.

Μετά που θα τελειώσει το κάθε μυρμηγκι τον κύκλο του, εξατμίζεται ένα ποσό.Η φερομόνη εξατμίζεται μέσα απο τον τύπο :

$$\tau_{ij} \leftarrow (1-\rho)\tau_{ij}, \tag{42}$$

με ρ να κυμαίνεται μεταξύ 0 και 1. Έτσι πιθανές κακές επιλογές, με την πάροδο των επαναλήψεων σβήνουν. Μετά την εξάτμιση ακολουθεί η αύξηση της φερομόνης στα χρησιμοποιημένα τόξα. Η καινούρια

ποσότητα ουσίας προστίθεται στα τόξα που χρησιμοποιήθηκαν από επιλεγμένα μυρμήγκια, πχ το καλύτερο, το 5 καλύτερα κ.α. Η φερομόνη προστίθεται μέσω του εξής τύπου:

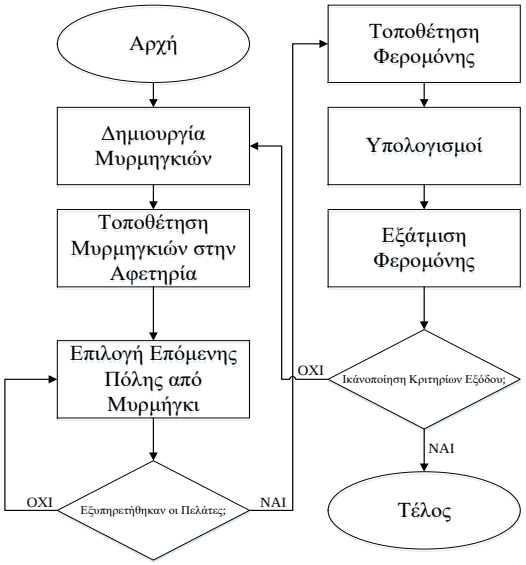
$$\tau_{ij} = \tau_{ij} + \sum_{k=1}^{m} \Delta\, \tau_{ij}^{k}, \tag{43}$$

με $\Delta\tau_{ij}^{k}$ είναι η ποσότητα φερομόνης που αφήνει το κάθε μυρμήγκι k, με τύπο:0

$$\Delta\tau_{ij}^{k} = \begin{cases} \dfrac{1}{C^{k}}, & \text{εάν το τόξο έχει επιλεγεί από το μυρμήγκι } k \\ 0, & \text{αλλιώς} \end{cases} \tag{44}$$

όπου C^{k} είναι το κόστος του κύκλου που έκανε το μυρμήγκι. Όταν το κόστος του κύκλου είναι χαμηλό, τότε αφήνεται περισσότερη φερομόνη, κι έτσι αυξάνεται η πιθανότητα επιλογής του σε επόμενη επανάληψη.

Το διάγραμμα ροής της παραπάνω διαδικάσίας μπορεί περιγράφεται ώς εξής:



Εικόνα 5:Διάγραμμα Ροής

Ο ψευδοκώδικας για την εφαρμογή του ACO σε προβλήτα TSP δίδεται παρακάτω.

Αρχικός υπολογισμός n_{ij} για όλους τους πελάτες i πρός όλους τους πελάτες j

Αρχικός υπολογισμός τ_{ij} για όλους τους πελάτες i πρός όλους τους πελάτες j

Επιλογή Αριθμού επαναλήψεων

Για αριθμό επαναληψεων

Τυχαία επιλογή πόλης εκκίνησης του μυρμηγκιών

Για κάθε μυρμήγκι

Όσο τα κριτήρια τερματισμού δεν πληρούνται

Επιλογή επόμενου κόμβου βάσει πιθανοτικού κανόνας ρουλεττας

Υπολογισμός συνάρτησης ποιότητας

Τέλος Όσο

Τέλος Για

Ενημέρωση Φερομόνης σε όλα τα τόξα βάσει του βέλτιστου μυρμηγκιού

Τοπική Αναζήτηση

Τέλος Για

Τύπωσε βέλτιστη διαδρομή

Ο συγκεκριμένος αλγόριθμος μπορεί να χρησιμοποιηθεί και για το VRP σε μια πιο ολιστική προσεγγιση, αλλα με την ίδια λογική , όπως θα δειχθεί στο επόμενο κεφάλαιο.

Παραλλαγές

Πολλές διαφορετικές παραλλαγές του αρχικού αλγορίθμου (Colorni, Dorigo, Maniezzo, & Trubian, 1994) έχουν προταθεί στη βιβλιογραφία, όπως, ο ελιτιστικός ACO (Dorigo, Maniezzo, & Colorni, 1996), Ant-Q (Dorigo & Gambardella, 1995), Max-Min ACO (Hoos & Stützle, 1996), ACO Ταξινόμησης (Bullnheimer, Hartl, & Strauss, 1999), Σύστημα Αποικιών Μυρμηγκιών (Gambardella & Dorigo, 1997), και ACO Hypercube (Blum, Roli, & Dorigo, 2001). Μία από τις βασικές διαφορές μεταξύ του αρχικού ACO και των επεκτάσεών του βρίσκεται στη διαδικασία ενημέρωσης της φερομόνης. Είτε γίνεται μια τοπική ενημέρωση φερομόνης κατά την κατασκευή της λύσης, είτε εκτελείται η διαδικασία τοποθέτησης φερομόνης στο τέλος της διαδικασίας, καθώς επίσης είναι πιθανό να γίνει η ενημέρωση του μονοπατιού απο όλα τα μυρμήγκια ή απο το καλύτερο/ καλύτερο ανα περιοδεία ή στο σύνολο. Οι παραλλαγές είναι αρκετές και μπορούν να βρεθούν σε πολλά έγγραφα της βιβλιογραφίας (Stützle, 2004) (Zhao, Wu, Zhao, & Quan, 2010) για

περισσότερες λεπτομέρειες σχετικά με τις διαφορές μεταξύ του ACO και των επεκτάσεών του. Θα πρέπει να σημειωθεί ότι ενώ η ACO ήταν αρχικά προταθεί για την επίλυση συνδυαστικών (διακριτών) προβλημάτων βελτιστοποίησης, ορισμένες εκδόσεις των αλγορίθμων ACO έχουν έχουν προταθεί για να χειριστούν όχι μόνο συνεχή προβλήματα βελτιστοποίησης (Dorigo & Socha, 2008) (Tsutsui, 2004), αλλά και μικτά προβλήματα βελτιστοποίησης με συνεχείς και διακριτές μεταβλητές (Socha, 2004).

Πλεονεκτήματα

Η διαδικασία που προαναφέρθηκε, όπως είναι λογικό, έχει εμφανή και μη εμφανή, πλεονεκτήματα. Υπάρχει δεδομένη ευελιξία του αλγορίθμου και δύναται να χρησιμοποιηθεί σε δυναμικές εφαρμογές (Haldenbilen, Cenk, & Baskan, 2013). Επιπλέον, είναι αρκετά εύκολο να χρησιμοποιηθεί είτε σε μικρές είτε σε μεγάλες εφαρμογές, αφού μπορεί να κυμαίνεται απο λίγα μόνο μυρμήγκια, έως χιλιάδες απο αυτά (Jaiswal & Aggarwal, 2011). Επιπλέον, η αποκεντρωμένη και μη καθοδηγούμενη διαδικασία εύρεσης λύσης βοηθάει στην έξοδο απο τυχόν τοπικά ελάχιστα, και τέλος ακόμη κι αν μερικά μυρμήγκια αποτύχουν να βρουν καλές λύσεις, η πλειοψηφία θα καταφέρει να συγκλίνει σε μια καλής ποιότητας λύση.

Μειονεκτήματα

Παρά τα πλεονεκτήματα του ACO, το κάνει να έχει κάποιους περιορισμούς (όπως πολλοί άλλοι αλγόριθμοι βελτιστοποίησης) που δεν επιτρέπουν πάντα την ορθή λειτουργία (Bonabeau, Dorigo, & Theraulaz, 1999). Για παράδειγμα, ο ACO δεν λειτουργεί επαρκώς όταν ένας μεγάλος αριθμός ακμών στο γράφημα είναι εξίσου πιθανές να είναι μέρος της καλής λύσης. Αυτό συμβαίνει όταν πολλές ακμές έχουν παρόμοιο κόστος και ως εκ τούτου παρόμοια πιθανότητα να επιλεγούν ως τμήματα των καλών διαδρομών (π.χ., ένα πρόβλημα TSP του οποίου οι πόλεις είναι ομοιόμορφα κατανεμημένες τυχαία με σχετικά ίση απόσταση η μια από την άλλη). Δεδομένου ότι στόχος ACO είναι να ενισχύσει όλες τις ακμές στο γράφημα κατασκευής του προβλήματος που ανήκουν σε καλές λύσεις / μονοπάτια, ένας μεγάλος αριθμός των ακμών θα λάβει σχετικά ίση μεγάλη ποσότητα φερομόνης και θα είναι εξίσου πιθανό να επιλεγεί. Στην περίπτωση αυτή, η αρχική έκδοση του ACO δεν θα αποδίδει καλά, δεδομένου ότι θα χρειαστεί περισσότερος χρόνος για να γίνει διάκριση μεταξύ αυτών των καλών διαδρομών, σε μια προσπάθεια να επιλεγεί τελικά μια απο αυτό το καλό σύνολο (Bonabeau, Dorigo, & Theraulaz, 2000).

Κεφάλαιο 3

Θεωρητικό Υπόβαθρο

Με στόχο τη μεγαλύτερη κατανόηση του προβλήματος προς επίλυση, θα παρουσιασθούν τα 3 βασικά προβλήματα της δρομολόγησης οχημάτων, καθώς επίσης και η παραλλαγή τους με την χρήση των χρονικών παραθύρων (Time Windows). Τα 3 προβλήματα που θα παρουσιασθούν είναι:

- Πρόβλημα Πλανόδιου Πωλητή (Travelling Salesman Problem, TSP)
- Πρόβλημα Δρομολόγησης Οχημάτων (Vehicle Routing Problem, VRP)
- Πρόβλημα Δρομολόγησης Αποθέματος (Inventory Routing Problem, IRP)

Στα πλαίσια της παρουσίασης που θα ακολουθήσει θα δειχθεί τόσο το πρόβλημα το οποίο επιλύουν όσο και η μαθηματική μοντελοποίηση του.

Πρόβλημα Πλανόδιου Πωλητή (TSP)

Το πρόβλημα προς επίλυση στην περίπτωση του πλανόδιου πωλητή είναι η εύρεση της συντομότερης διαδρομής, είτε σε απόσταση είτε σε χρόνο ή σε οποιοδήποτε άλλο κόστος επιλεγεί, για ένα πωλητή/όχημα. Ο πωλητής/όχημα ξεκινάει απο ένα κόμβο αφετηρία (σπίτι πωλητή,κεντρο διανομής,εργοστάσιο παραγωγής,κ.ά) και μετά απο επίσκεψη σταθερού αριθμού πελατών ακριβώς μια φορά έκαστο, επιστρέφει στον αρχικό κόμβο αφετηρία.

Το μαθηματικό μοντέλο του συγκεκριμένου προβλήματος απαιτεί οι κόμβοι να μην παρουσιάζουν κάποια διάταξη καθώς και το γράφημά τους να είναι πλήρες. Έστω ο κόμβος αφετηρίας $i = 1$ και $i = 2 \dots . n$ οι κόμβοι που εκφράζουν τους πελάτες. Επιπλέον, τα ζεύγη κόμβων $\{i; j\}$ με $i \neq j$, $i = 1 \dots . n$ και $j = 1 \dots . n$ δημιουργούν συνδέσμους/ακμές στο γράφημα. Κάθε τέτοιος σύνδεσμος αντιστοιχεί σε ένα κόστος c_{ij} όπου υποδηλώνει το κόστος μετάβασης απο τον κόμβο i στον j ή αντίστροφα.

Η σύνδεση μεταξύ 2 κόμβων στην προκειμένη περίπτωση θεωρείται ότι είναι η διαδρομή ελαχίστου κόστους και πληροί τις εξής συνθήκες.

Επειδή ένας τέτοιος σύνδεσμος δεν αντιστοιχεί πάντοτε σε κάποιο φυσικό τμήμα δρόμου γίνεται η υπόθεση ότι τα σταθμά έχουν υπολογισθεί έτσι ώστε να αντιστοιχούν στην διαδρομή ελαχίστου κόστους μεταξύ των δύο κόμβων, ώστε να ικανοποιούν τις δύο συνθήκες :

1. Συνθήκη Συμμετρίας: $c_{ij} = c_{ji}$, με $i \neq j$, $i = 1 \dots . n$, $j = 1 \dots . n$
2. Συνθήκη Τριγωνικής ανισότητας: $c_{ij} \leq c_{ki} + c_{ik}$, $i \neq j \neq k$, $i = 1 \dots . n$, $j = 1 \dots . n$, $k = 1 \dots . n$

Η πρώτη συνθήκη δείχνει οτι το κόστος της διαδρομής είναι ανεξάρτητο απο την κατεύθυνση σε ένα ζευγάρι κόμβων $\{i; j\}$, ενώ η δεύτερη οτι ο συντομότερος δρόμος μεταξύ τους είναι απευθείας.

Η χρήση δυαδικών μεταβλητών είναι απαραίτητη. Έτσι αυτές ορίζονται ως:

$$x_{ij} = \begin{cases} 1, & \sigma\varepsilon\ \pi\varepsilon\rho\acute{\iota}\pi\tau\omega\sigma\eta\ \chi\rho\acute{\eta}\sigma\eta\varsigma\ \tau o\upsilon\ \tau\acute{o}\xi o\upsilon\ \mu\varepsilon\tau\alpha\xi\acute{\upsilon}\ \{i; j\} \\ 0, & \alpha\lambda\lambda\iota\acute{\omega}\varsigma\ \forall i, \forall j, i \neq j \end{cases}$$

Το συνολικό πρόβλημα που δημιουργείται βάση των παραπάνω είναι:

$$\min \sum_{i=1}^{n} \sum_{\substack{j=1 \\ j \neq i}}^{n} c_{ij} x_{ij} \tag{1}$$

$$\upsilon\pi\acute{o} \sum_{j=1}^{n} x_{ij} = 2\ , i = 1 \dots n \tag{2}$$

$$\sum_{i \in C} \sum_{j \in \bar{C}} x_{ij} \geq 1\ , \forall C \subset \{1 \dots n\}, C \neq \emptyset \tag{3}$$

$$x_{ij} \in \{0,1\}, \forall i, \forall j, i \neq j \tag{4}$$

Το πρώτο σετ περιορισμών υποχρεώνει τη λύση να έχει δυο συνδέσμους σε κάθε κόμβο. Επιβάλλει έτσι την είσοδο του οχήματος κατά μήκος του ενός κόμβου και την έξοδο του κατά μήκος του άλλου. Στην συνέχεια οι περιορισμοί 3 οδηγούν στην εξάλειψη κυκλικών διαδρομών που δεν περιέχουν όλους τους κόμβους. Έτσι, εξασφαλίζεται οτι το όχημα θα διέλθει απο όλους τους κόμβους, σε κάθε πιθανή λύση.

Σε περίπτωση που η κατεύθυνση αλλάζει το κόστος, δηλαδή όταν δεν ισχύει η συνθήκη συμμετρίας, το μαθηματικό μοντέλο αλλάζει, με προσθήκη περιορισμών στη θέση του 3. Οι περιορισμοί αυτοί είναι :

$$\sum_{j=1}^{n} x_{ij} = 1\ , i = 1 \dots n \tag{5}$$

$$\sum_{j=1}^{n} x_{ij} = 1\ , j = 1 \dots n \tag{6}$$

όπου λόγω των παραπάνω περιορισμών επιβάλλεται ακριβώς ένα τόξο απο οποιονδήποτε κόμβο i προς j, και αντίστοιχα για οποιονδήποτε κόμβο j προς i.

Ο αριθμός των περιορισμών σε κάθε μια απο τις περιπτώσεις γίνεται εξαιρετικά μεγάλος, σε περίπτωση αυξημένου αριθμού κόμβων. Έτσι, με στόχο τη μείωση των περιορισμών γίνονται κάποιες παραδοχές όπως:

1. Στην αφετηρία παράγεται μια μονάδα κάποιου αγαθού για κάθε πελάτη.
2. Τα αγαθά δεν αναμειγνύονται.
3. Η αφετηρία «κλωνοποιείται» ώστε να δημιουργήσει ένα πλασματικό πελάτη $n_o = n + 1$.

Βάση των παραπάνω μια νέα μεταβλητή y_{ij}^k που αναφέρει την ποσότητα που μεταφέρεται απο ένα πελάτη k σε ένα τόξο (ij), με $k = 2 \dots . n$.

Συνολικά το νέο μοντέλο είναι το εξής:

$$\min \sum_{i=1}^{n_o} \sum_{\substack{j=1 \\ j \neq i}}^{n_o} c_{ij} x_{ij} \tag{7}$$

$$\text{υπό} \sum_{j=1}^{n_o} x_{ij} = 1 \, , i = 1 \dots n \tag{8}$$

$$\sum_{i=1}^{n_o} x_{ij} = 1 \, , j = 2 \dots n_o \tag{9}$$

$$\sum_{\substack{j=1 \\ j \neq i}}^{n_o} y_{ij}^k - \sum_{\substack{j=1 \\ j \neq i}}^{n_o} y_{ij}^k = \begin{cases} 1, \alpha \nu \; i = 1 \\ -1, \alpha \nu \; i = k \, , \forall i, \forall k \\ 0, \alpha \lambda \lambda \iota \acute{\omega} \varsigma \end{cases} \tag{10}$$

$$y_{ij}^k \leq x_{ij}, \forall i, \forall k, \forall j, i \neq j \tag{11}$$

$$y_{ij}^k \geq 0, \forall i, \forall k, \forall j, i \neq j \tag{12}$$

$$x_{ij} \in \{0,1\} \, \forall i, \forall j, i \neq j \tag{13}$$

Η αντικειμενική συνάρτηση και οι περιορισμοί (3) – (4) και (7) παραμένουν στην ίδια μορφή με τους

αντίστοιχους των αρχικών μοντέλων. Ο περιορισμός (4) είναι οι εξισώσεις ισορροπίας της ροής και εγγυώνται ότι η μία μονάδα από το σημείο αφετηρίας με προορισμό τον πελάτη k φτάνει στον προορισμό της χωρίς να παραδοθεί σε κάποιον από ενδιάμεσους κόμβους i. Για να διασφαλιστεί οτι τα αγαθά για τον πελάτη k κινούνται πάνω σε επιλεγμένα τόξα $x_{ij} = 1$, στα πλαίσια της βέλτιστης λύσης, υπάρχει ο περιορισμός (5).

Στο μοντέλο αυτό, όπως φαίνεται απο τους περιορισμούς (2)-(3), υπάρχει ένα τόξο εισόδου και εξόδου για κάθε κόμβο, εκτός απο τον κόμβο αφετηρίας, που έχει ένα τόξο εξόδου, και τον κλώνο της αφετηρίας που έχει μόνο τόξο εισόδου.

Βάσει των περιορισμών ροής, απαιτείται για μια εφικτη λύση, η άφιξη των κατάλληλων αγαθών στον κατάλληλο πελάτη, καθώς και να πραγματοποιηθεί ένα πλήρης κύκλος απο όλους τους πελάτες συμπεριλαμβανομένου και του πελάτη «κλώνου» της αφετηρίας. Για να γίνει αυτό απαιτούνται $n - 1$ τόξα για να εξυπηρετηθούν $n - 1$ πελάτες. Η βέλτιστη λύση θα αποτελείται από n τόξα τα οποία, εάν συμπτύξουμε την αφετηρία και τον «κλώνο» της, θα μας δώσουν μια εφικτή λύση στο TSP (Μυγδαλάς & Μαρινάκης, 2008).

Πρόβλημα Δρομολόγησης Οχημάτων (VRP)

Το VRP είναι άμεσα συνδεδεμένο με το TSP καθώς αποτελεί συνέχειά του και είναι απο τα πιο μελετημένα προβλήματα στο χώρο των logistics και της συνδυαστικής βελτιστοποίησης. Στο παρόν πρόβλημα υπάρχουν παραπάνω απο ένα οχήματα τα οποία καλούνται να κινηθούν μεταξύ των κόμβων, δηλαδή αφορά στόλο διαθέσιμων οχημάτων. Όπως και στο TSP, τα οχήματα πρέπει να κινηθούν μεταξύ όλων των κόμβων, καθώς και να ικανοποιούν τους περιοσμούς χωρητικότητας και χρόνου.

Το πρόβλημα εξαρτάται από τη γνώση αναφορικά με το μέγεθος του στόλου, τον αριθμό των διαθέσιμων οδηγών, τον αριθμό των καθημερινών διαδρομών,το κόστος των οδηγών, ενδεχόμενες μελλοντικές βλάβες οχημάτων αλλά και την υπολογιστική ισχύ που κατέχει η επιχείρηση για τη βελτιστοποίηση του προβλήματος, καθώς και συνδυασμό των δρομολογίων με τις υπόλοιπες δραστηριότητες της επιχείρησης.

Επιπλέον, απο την πλεύρά των πελατών πρέπει να είναι γνωστό το σημείο στο οποίο βρίσκεται ο στόλος, η ποσότητα και το είδος που απαιτείται να παραδοθεί ή να συλλεχθεί απο αυτόν, τα χρονικά περιθώρια εξυπηρέτησης (Time Windows), όπως αυτά θα αναφερθούν παρακάτω, ο χρόνος που απαιτείται για την εξυπηρέτηση του πελάτη και τέλος το είδος του οχήματος που απαιτείται για να γίνει ορθή εξυπηρέτηση αυτού.

Οι αποθήκες του δικτύου εξυπηρετούν συγκεκριμένους πελάτες, βάσει των οποίων εξάγονται οι διαδρομές που προκύπτουν. Κάθε αποθήκη έχει χαρακτηριστικά χωριτικότητας προϊόντων και πλήθους οχημάτων που μπορεί να βρίσκονται σε αυτή. Επιπλέον τα χαρακτηριστικά των οχημάτων που βρίσκονται σε αυτές συνοψίζονται σε αποθήκη προέλευσης οχήματος, αποθήκη τερματισμού διαδρομής, χωρητικότητα οποιασδήποτε μονάδας (βάρους, όγκου, αριθμού πελατών που μπορεί να εξυπηρετηθεί κλπ), αριθμός τμημάτων του οχήματος και χαρακτηριστικά φόρτωσης του οχήματος,διαθεσιμότητα φορτοεκφορτωτικών μηχανημάτων, εφικτές διαδρομές (χωματόδρομος κλπ) και κόστος λειτουργίας.

Οι οδηγοί των οχημάτων διέπονται και αυτοί απο συγκεκριμένους περιορισμούς, φυσικούς και νομικούς, όπως οκτώ ώρες ύπνου την ημέρα υποχρεωτικά, λιγότερες από δέκα ώρες συνεχόμενης οδήγησης, λιγότερες από έξι ημέρες οδήγησης εβδομαδιαίως, λιγότερες από δεκαπέντε ώρες οδήγησης ημερησίως.

Οι διαδρομές απο μόνες τους πρέπει να πληρούν κάποιους περιορισμούς, όπως οτι η μεταφερόμενη ποσότητα ανα διαδρομή δε μπορεί να ξεπερνά την χωρητικότητα του οχήματος, οι ανάγκες του πελάτη στο συγκεκριμένο δρομολόγιο (μόνο παραλαβή, μόνο παράδοση ή συνδυασμός), προτεραιότητα αναφορικά με την εξυπηρέτηση, σύμφωνα με τη ζήτηση, τα χρονικά παράθυρα, διαθεσιμότητα οδηγών σε συγκεκριμένο χρόνο και ικανότητα μεταφοράς οχημάτων.

Η γραφική απεικόνιση των λύσεων γίνεται με τη χρήση γραφημάτων, με τους κόμβους να περιγράφουν τους πελάτες και τα τόξα να περιγράφουν τις διαδρομές μεταξύ τους. Τα τόξα μπορούν να είναι είτε μονής είτε διπλής κατεύθυνσης, πράγμα που εξαρτάται απο τον κανονισμό του δρόμου (πχ. μονόδρομος), και εμπεριέχουν κάποιο κόστος. Το κόστος του τόξου εξαρτάται απο το μήκος του, καθώς και το χρόνο που κάνει το όχημα να το διανύσει, κάτι το οποίο κάνει το εκάστοτε κόστος της διαδρομής δυναμικό, λόγω φόρτου διαδρομής (κίνηση) ή είδος του οχήματος.

Το συνολικό κόστος της λύσης υπολογίζεται μέσα απο την πληροφορία του κόστους και του χρόνου της διαδρομής μεταξύ ζεύγους πελατών ή πελατών και αποθήκης. Έτσι στο τελικό γράφημα, μεταξύ κόμβων i και j σχηματίζεται ένα τόξο $(i; j)$ του οποίου το κόστος c_{ij} αντιστοιχεί στο κόστος της συντομότερης διαδρομής που ξεκινά από τον κόμβο i και καταλήγει στον κόμβο j του γραφήματος. Αντίστοιχα, ο χρόνος t_{ij} υποδεικνύει τον χρόνο μετάβασης απο τον κόμβο i στον κόμβο j. Αθροίζοντας το κόστος και τον χρόνο των τόξων ανά διαδρομή μπορεί να βρεθεί το συνολικό κόστος, είτε χρηματικό είτε χρονικό της λύσης.

Τα προβλήματα δρομολόγησης στόχο έχουν να βελτιστοποιήσουν το κόστος απο διάφορες σκοπιές, όπως το σύνολο της απόστασης που χρειάζεται να διανυθεί για τη μεταφορά όλων των προϊόντων, το σύνολο του χρόνου που απαιτείται για να γίνει αυτό. Επιπλέον, είναι πιθανό να υπάρχει ανάγκη ελαχιστοποίησης του αριθμού των οχημάτων ή των οδηγών που χρειάζονται καθώς και τις ποινές απο μη έγκαιρη

εξυπηρέτηση πελατών. Είναι δυνατόν να γίνει συνδυασμός στόχων με την χρήση βαρών, ανάλογα με τη σημασία που δίνει ο επιλύτης σε κάθε ένα απο τους προς ελαχιστοποιήση στόχους.

Όπως το TSP, το απλό VRP είναι ένα πρόβλημα ακέραιου προγραμματισμού. Με $i = 1 \dots . . n$ οι κόμβοι που αντιπροσωπεύουν τους πελάτες, με $i = 1$ ο κόμβος-αποθήκη. Έστω d_i η ζήτηση του πελάτη i σε προϊόντα, με την αποθήκη να έχει μηδενική, και c_{ij} να είναι το κόστος μεταφοράς απο τον πελάτη i στον j. Η εταιρεία έχει ένα στόλο K οχηματων με την χωρητικότητα του καθενός να συμβολίζεται με Q. Έτσι το μοντελο που δημιουργείται είναι το εξής:

Κάθε πελάτης i έχει ζήτηση di ποσότητα προϊόντων (η ζήτηση του κόμβου 1, δηλαδή της αποθήκης είναι d1 = 0) και το κόστος μετάβασης από τον πελάτη i στον j ορίζεται ως cij . Η εταιρεία διαθέτει K οχήματα για τις μεταφορές, ενώ η χωρητικότητα κάθε οχήματος συμβολίζεται με Q. Ζητείται να ελαχιστοποιηθεί η παρακάτω συνάρτηση:

$$\min \sum_{k=1}^{K} \sum_{i=1}^{n} \sum_{j=1}^{n} c_{ij} x_{ijk} \tag{14}$$

$$\text{υπό} \sum_{k=1}^{K} \sum_{j=1}^{n} x_{1jk} = K \tag{15}$$

$$\sum_{i=1}^{n} \sum_{j=1}^{n} d_i x_{ijk} \leq Q \,, \forall\, k \in \{1, \dots K\} \tag{16}$$

$$\sum_{k=1}^{K} \sum_{j=1}^{n} x_{ijk} = 1 \,, \forall\, j \in \{2, \dots n\} \tag{17}$$

$$\sum_{j=1}^{n} x_{ijk} - \sum_{j=1}^{n} x_{ijk} = 0, \forall\, k \in \{1, \dots K\}, \forall\, i \in \{2, \dots n\} \tag{18}$$

$$\sum_{i,j \in S}^{n} x_{ijk} \leq |S| - 1, \forall\, k \in \{1, \dots K\}, \forall\, S \subseteq \{2 \dots n\} \tag{19}$$

$$x_{ijk} \in \{0,1\}, \forall\, k \in \{1, \dots K\}, \forall\, j \in \{1, \dots n\}, \forall\, i \in \{2, \dots n\} \tag{20}$$

$$x_{ijk} = \begin{cases} 1, & \textit{εάν το όχημα } k \textit{ πάει στον κόμβο } j \textit{ αμέσως μετά τον κόμβο } i \\ 0, & \textit{αλλιώς} \end{cases} \tag{21}$$

Ο περιορισμός 15 δηλώνει ότι από την αποθήκη φεύγουν K οχήματα.Η χωρητικότητα εξασφαλίζεται απο τον περιορισμό 16 να είναι ίση με Q. Η μοναδική επισκεψιμότητα του κόμβου με τον περιορισμό 17 ενώ με τον 18 εξασφαλίζεται ότι το όχημα που εισέρχεται σε ένα κόμβο είναι το ίδιο με εκείνο που εξέρχεται από αυτόν. Η ολοκληρωμένη διαδρομή αφαιρείται απο το σύνολο, μέσω του περιορισμού 19.

Πρόβλημα Δρομολόγησης Αποθεμάτων (IRP)

Ο διευρυμένος ρόλος της εφοδιαστικής αλυσίδας έχει οδηγήσει τις επιχειρήσεις στο να αντιλαμβάνονται συνολικότερα τον πελάτη, αφού έχει γίνει κατανοητή η ανάγκη για ποιοτικότερη εξυπηρέτηση τόσο σε βάθος χρόνου όσο και στην ευκολία παραγγελιών, κόστους αποθεματοποιήσης κλπ.

Η ανάγκη για καλύτερο σχεδιασμό απο άποψη προμηθευτή και η επαναλαμβανόμενη ζήτηση του καταναλωτή για προϊόντα προερχόμενα απο έναν προμηθευτή, οδήγησε στη δημιουργία του IRP. Η καθημερινή ζήτηση θέτει στον προμηθευτή το πρόβλημα του σχεδιασμού πόρων και της διαχείρισης του στόλου καθώς και του συντονισμού των πολιτικών ανεφοδιασμού για όλο το πελατολόγιο του. Έτσι, η οικονομική αποδοτικότητα της επίλυσης του προβλήματος ανεφοδιασμού σε καθημερινή βάση είναι μείζον ζήτημα στην εφοδιαστική αλυσίδα.

Όπως υπάρχει συσχέτιση μεταξύ του VRP με το TSP, έτσι και το IRP με το VRP έχουν άμεση σύνδεση αφου είναι μια εμπλουτισμένη εκδοχή του δεύτερου. Μέσω της έννοιας της αποθεματοποίησης το πρόβλημα έχει σαν αντικείμενο την επαναλαμβανόμενη διανομή ενός μόνο προϊόντος, από μία αποθήκη, σε ένα σύνολο n πελατών, για ένα δεδομένο χρονικό ορίζοντα T. Υπάρχει κατανάλωση d του προϊόντος, και διατήρηση ενός αποθέματος D απο πλευράς πελατών. Ο στόλος Kοχημάτων χωρητικότητας Q διανέμει τα κατάλληλα προϊόντα στους πελάτες. Η κατανομή του στόλου αυτού έχει σαν στόχο την ελαχιστοποιήση τόσο του κόστους αποθεματοποίησης όσο και του κόστου δρομολόγησης, χωρίς να δημιουργεί έλλειμμα αποθεμάτων στον πελάτη. Έτσι αποφασίζεται, πότε θα εξυπηρετηθεί ο πελάτης (Μυγδαλάς & Μαρινάκης, 2008) , ποσότητα προϊόντος που θα του παραδοθεί και προφανώς ποιες διαδρομές θα ακολουθηθούν.

Παρομοίως με τα 2 προηγούμενα προβλήματα, το πρόβλημα περιγράφεται μέσω της θεωρίας γραφημάτων. Έστω ότι έχουμε ένα πλήρες γράφημα $G = (V; A)$, όπου $V = \{0;\ 1 \dots n\}$ είναι το σύνολο των κόμβων και $A = \{(i,j), i,j \in V;\ i \neq j\}$ είναι το σύνολο των τόξων που συνδέουν τους κόμβους μεταξύ τους. Ο κόμβος 1 αντιστοιχεί στην αποθήκη και οι υπόλοιποι κόμβοι αντιστοιχούν στους πελάτες.

Η αποθήκη λειτουργεί σαν χώρος στάθμευσης και ανεφοδιασμού για τα οχήματα. Κόστος φόρτωσης ή εκφόρτωσης των οχημάτων και κόστος αποθεματοποίησης στην αποθήκη δεν λαμβάνεται υπόψη. Κάθε δρομολόγιο ξεκινάει από την αποθήκη και καταλήγει σε αυτήν. Επίσης, αν κάποια ημέρα $t = \{1;\ 2; \dots T\}$ το επίπεδο του αποθέματος κάποιου πελάτη i πέσει στο μηδέν θεωρείται ότι θα εξυπηρετηθεί αμέσως από κάποιο όχημα, συνεπώς δεν λαμβάνεται υπόψη κόστος έλλειψης λόγω καθυστερημένης άφιξης οχήματος.

Έτσι κάθε πελάτης $i = \{1 \dots n\}$ έχει μια ζήτηση d_i ανά ημέρα και ένα κόστος αποθεματοποίησης h_i και το c_{ij} αντιπροσωπεύει το κόστος μετάβασης απο τον κόμβο i στον κόμβο j. Ισχύει η συνθήκη της συμμετρίας και οτι υπάρχει επαρκής αποθηκευτικός χώρος στους πελάτες, οπότε $D_i = Td_i$. Για κάθε μέρα $t = \{1 \dots n\}$ το όχημα περνάει μια και μοναδική φορά απο τον πελάτη, απο ένα οποιοδήποτε όχημα του στόλου, με

ποσότητα προϊόντος το πολύ ίση με D_i. Προφανώς το όχημα δεν μπορέι να μεταφέρει μεγαλύτερη ποσότητα απο την χωρητικότητα Q. Ο στόχος του προβλήματος είναι η εύρεση κατάλληλων ποσοτήτων ανά μέρα για ελαχιστοποίηση κόστους μεταφοράς αλλά και αποθεματοποίησης. Όπως στα προηγούμενα προβλήματα, θα γίνει χρήση δυαδικής μεταβλητής x_{ijkt} που δείχνει αν επισκέφθηκε το όχημα k τη μέρα t τον κόμβο j με αφετηρία τον κόμβο i. Με q_{it} συμβολίζεται η ποσότητα προϊόντος που μεταφέρεται στον πελάτη i την μέρα t, ποσότητα που είναι ακέραιο πολλαπλάσιο της ημερήσιας ζήτησης d_i του πελάτη . Το επίπεδο αποθέματος του πελάτη την συμβολίζεται με τον όρο sb_{it} ενώ με z_{it} συμβολίζεται το αν παρέλαβε ή όχι ο πελάτης i κάποια ποσότητα q_{it}. Η μεταβλητή e_{it} δείχνει αν έχει απόθεμα sb_{it} ο πελάτης μετά τον ανεφοδιασμό την μέρα t. Το συνολικό πρόβλημα που δημιουργείται είναι το εξής:

$$\min \sum_{t=1}^{T}\sum_{i=1}^{n}(sb_{it} + q_{it} - \frac{d_{it}}{2}) * h_i + \sum_{k=1}^{K}\sum_{t=1}^{T}\sum_{i=0}^{n}\sum_{j=0}^{n} c_{ij}x_{ijtk} \tag{22}$$

$$\sum_{i=0}^{n} x_{ijtk} - \sum_{p=0}^{n} x_{jptk} = 0, \forall j \in \{0 \dots n\}, \forall k \in \{1, .. K\}, \forall t \in \{1, \dots T\} \tag{23}$$

$$\sum_{j=1}^{n} x_{0jtk} \leq 1, \forall k \in \{1 \dots K\}, \forall t \in \{1 \dots T\} \tag{24}$$

$$q_{it} \leq D_i z_{it}\ , \forall i \in \{1 \dots n\}, \forall t \in \{1 \dots T\} \tag{25}$$

$$q_{it} \geq z_{it}\ , \forall i \in \{1 \dots n\}, \forall t \in \{1 \dots T\} \tag{26}$$

$$\sum_{i=0}^{n} x_{ijtk} - z_{it} = 0, \forall i \in \{1 \dots n\}, \forall t \in \{1 \dots T\} \tag{27}$$

$$\sum_{i=0}^{n} x_{ijtk} q_{it} \leq Q\ , \forall k \in \{1 \dots K\}, \forall t \in \{1 \dots T\} \tag{28}$$

$$sb_{i(t+1)} = sb_{it} + q_{it} - d_i, \forall i \in \{1 \dots n\}, \forall t \in \{1 \dots T\} \tag{29}$$

$$sb_{i1} = sb_{iT} + q_{iT} - d_i, \forall i \in \{1 \dots n\} \tag{30}$$

$$sb_{it} \geq e_{it}, \forall i \in \{1 \dots n\}, \forall t \in \{1 \dots T\} \tag{31}$$

$$\sum_{t=1}^{T} e_{it} \leq T, \forall i \in \{1 \dots n\} \tag{32}$$

$$\sum_{i \in B}\sum_{j \in B} x_{ijk} \leq |B| - 1\ , \forall k \in \{1, .. K\}, \forall t \in \{1 \dots T\}, B \subseteq V \backslash \{i = 0\}, |B| \succ 1 \tag{33}$$

$$x_{ijkt} \in \{0,1\}\ , \forall k \in \{1, .. K\}, \forall t \in \{1 \dots T\}, \forall i \in \{1, \dots n\}, \forall j \in \{1 \dots n\} \tag{34}$$

$$z_{it} \in \{0,1\}, \forall t \in \{1 \dots T\}, \forall i \in \{1, \dots n\} \tag{35}$$

$$e_{it} \in \{0,1\}, \forall t \in \{1 \dots T\}, \forall i \in \{1, \dots n\} \tag{36}$$

$$q_i \in \{0, d_i, 2d_i, \dots D_i\}, \forall t \in \{1 \dots T\}, \forall i \in \{1, \dots n\} \quad (37)$$

Όπως παρατηρείται η αντικειμενική εμπεριέχει την έννοια της ελαχιστοποίησης τόσο για τους όρους του κόστους αποθεματοποίησης όσο και του συνολικού κόστους μεταφοράς, για τον χρονικό ορίζοντα T. Ο περιορισμός 23 πραγματώνει την έξοδο του οχήματος απο τον πελάτη, ενώ ο επόμενος περιορισμός δημιουργεί το μοναδικό περάσμα του οχήματος ανά ημέρα ανά πελάτη. Οι περιορισμοί 25 και 26 εξασφαλίζουν ότι $z_{it} = 1$ αν και μόνο αν παραδοθεί κάποια ποσότητα q_{it}. Ο περιορισμός 28 αφορά την χωρητικότητα των οχημάτων και εξασφαλίζει οτι θα μεταφερθεί ποσότητα ίση ή μικρότερη με την χωρητικότητα. Οι περιορισμοί 29 και 30 φανερώνουν ότι το επίπεδο του αποθέματος ενός πελάτη την ημέρα $t + 1$ εκφράζεται ως συνάρτηση του επιπέδου του αποθέματος και της ποσότητας ανεφοδιασμού την ημέρα t. Ο περιορισμός 34 συμπεριλαμβάνει την αποθήκη σε κάθε δρομολόγιο. Οι τελευταίοι περιορισμοί καθορίζουν τη φύση των μεταβλητών.

Χρονικά Παράθυρα (Time Windows-TW)

Η χρήση των χρονικών παραθύρων είναι πολύ βασική σε μια σύγχρονη επιχείρηση. Τα χρονικά παράθυρα καθορίζουν τις ώρες τις οποίες η επιχείρηση μπορεί να εξυπηρετηθεί και να λάβει/επιστρέψει προϊόντα. Η χρήση των χρονικών περιορισμών ανεβάζει το επίπεδο των προσφερόμενων υπηρεσιών, όμως δυσκολεύει περαιτέρω τον πρόβλημα και δημιουργεί μεγαλύτερη εξάρτηση της εφοδιαστικής αλυσίδας στην υπολογιστική ισχύ που μπορεί να έχει στη διάθεση του ο διανομέας. Τα χρονικά παράθυρα παρουσιάζουν μια νέα πρόκληση για την εξυπηρέτηση του πελάτη καθώς ανεβάζει το επίπεδο επιπλοκότητας και δυσκολεύουν την εύρεση πιθανών βέλτιστων λύσεων. Η επίλυση τους μπορεί να βασιστεί είτε στην δημιουργία penalty κόστους σε περίπτωση υπέρβασης των περιορισμών, είτε στον πλήρη αποκλεισμό της λύσης που υπερβαίνει τον περιορισμό. Επιπλέον, η εφαρμογή διαδικασιών απεμπλοκής απο τοπικά ελάχιστα γίνεται πιο δύσκολη. Η χρήση των χρονικών παραθύρων μπορεί να γίνει με την προσθήκη επιπλέον περιορισμών στις μοντελοποιήσεις. Έστω η μεταβλητή απόφασης s_{ik} που ορίζει τη χρονική στιγμή που το όχημα k εκκινεί να για να εξυπηρετήσει τον πελάτη i. Σε περίπτωση που ο πελάτης i δεν εξυπηρετείται απο το συγκεκριμένο όχημα, η ύπαρξη της μεταβλητής αυτής στερείται νοήματος. Επιπλέον οι μεταβλητές a_i και b_i που δείχνουν το χρονικό περιθώριο εξυπηρέτησης του πελάτη, με a_i την εκκίνηση αυτού και b_i τη λήξη του. Επιπλέον, ορίζεται ένας χρόνο t_{ij} όπου παρουσιάζεται ο χρόνος μετάβασης του οχήματος απο τον πελάτη i στον πελάτη j. Ο χρόνος αυτός είναι ουσιαστικά η Ευκλείδια απόσταση μεταξύ των πελατών αυτών, συμπεριλαμβανομένου του χρόνου εξυπηρέτησης του πελάτη i. Στην περίπτωση της αποθήκης ορίζεται $a_0 = 0$, όπου το όχημα απαγορεύται να ξεκινήσει νωρίτερα απο αυτό το χρόνο, και το όχημα απαγορεύεται να επιστρέψει στην αποθήκη σε χρόνο μεγαλύτερο του b_{n+1}. Η σχέση της τριγωνικής ανισότητας ισχύει και στην περίπτωση των μεταβλητών t_{ij}. Τέλος, θεωρείται ότι τα a_i, b_i είναι μη

αρνητικοί ακέραιοι αριθμοί και ότι τα t_{ij} είναι θετικοί ακέραιοι αριθμοί. Οι νέοι περιορισμοί που προστίθενται στο VRP πρόβλημα είναι:

$$s_{ik} + t_{ij} - M_{ij}(1 - x_{ijk}) \leq s_{jk} \quad \forall i, j \in N, \forall k \in K \tag{38}$$

$$a_i \leq s_{ik} \leq b_i \ \ \forall i \in N, \forall k \in K \tag{39}$$

Ο πρώτος τύπος περιορισμών δηλώνει ότι ένα όχημα k δεν μπορεί να επισκεφθεί τον πελάτη j πριν την χρονική στιγμή $s_{ik} + t_{ij}$ εαν μεταβαίνει από τον πελάτη i στον πελάτη j. Το M_{ij} είναι μια σταθερή τιμή που να λάβει είτε μια πολύ μεγάλη τιμή αυθαίρετα, είτε να χρησιμοποιηθεί η τιμή $\max\{b_i + t_{ij} - a_j\}$, (i, j) ∈ A. Ο δεύτερος τύπος περιορισμών ξεκαθαρίζει οτι η εξυπηρέτηση πρέπει να πραγματοποιηθεί εντός των χρονικών πλαισίων που έχουν τεθεί απο τον πελάτη.

Με την παρουσίαση των παραπάνω προβλημάτων και μοντελοποιήσεων ολοκληρώνεται η εισαγωγή στα προβλήματα δρομολόγησης όπως αυτά θα επιλύθηκαν στα πλαίσια της παρούσας εργασίας.

Κεφάλαιο 4

Προβλήματα Solomon και IRP

Η εφαρμογή του προτεινόμενου αλγορίθμου, στο κομμάτι του VRP, έγινε πάνω στα γνωστά προβλήματα του Solomon, καθώς η χρήση τους είναι διαδεδομένη και τα αποτελέσματά τους γνωστά (Solomon, 1987) Στα προβλήματα αυτά παρουσιάζονται σύνηθη χαρακτηριστικά διαδικασιών δρομολόγησης, όπως αυτά δύναται να εμφανιστούν και σε πραγματικές εφαρμογές. Έτσι, δίδεται το πλήθος των πελατων, οι συντεταγμένες τους σε καρτεσιανό επίπεδο και τα χρόνικα τους παράθυρα, η ζήτησή τους και ο χρόνος εξυπηρέτησης του κάθε ενός απο αυτούς. Επιπλέον, σα δεδομένο υπάρχει και σύνολο των διαθέσιμων οχημάτων και η χωρητικότητα του κάθε οχήματος. Το σύνολο των προβλημάτων προς επίλυση είναι 56, καθένα απο αυτά με διαφορετικά χαρακτηριστικά αλλά και κοινά στοιχεία. Λεπτομερέστερα:

- Όλα τα προβλήματα έχουν συνολικά 100 πελάτες και η κατανομή τους γίνεται εντός τετραγώνου πλευράς 100 μονάδων μήκους (Repoussis, Tarantilis, & Ioannou, 2009).
- Συνολικά τα προβλήματα χωρίζονται σε 2 κατηγορίες συνολικά 3 κλάσεων. Συγκεκριμένα οι κλάσεις ονομάζονται R,C και RC και κάθε μια απο αυτές εχεις 2 κατηγορίες (R1,R2 και ομοίως και για τις άλλες κλάσεις).
- Η κάθε κλαση παρουσιάζει τη δική της κατανομή αναφορικά με τους πελάτες. Έτσι, στην κλάση R οι συνεταγμένες των πελατών έχουν παραχθεί τυχαία μέσω της ομοιόμορφης κατανομής, ενώ στην κλάση C είναι τοποθετημένοι σε σμήνη. Τέλος στην RC κλάση, κάποιοι πελάτες είναι σε σμήνη και κάποιοι έχουν συντεταγμένες παραχθείσες απο γεννήτρια ομοιόμορφης κατανομής.
- Οι κατηγορίες 1 (C1,R1,RC1) έχουν έχουν μικρά χρονικά παράθυρα και αντίστοιχα μικρο χρονικό ορίζοντα καθώς και τα οχήματα παρουσιάζουν μικρή χωρητικότητα, της τάξης των 200 μονάδων (Repoussis, Tarantilis, & Ioannou, 2009). Τα παραπάνω χαρακτηριστικά οδηγούν σε αυξημένο αριθμό οχημάτων που γίνεται χρήση (5-10) (Kohl, Desrosiers, Madsen, & Solomon, 1999).
- Οι κατηγορίες 2 (C2,R2,RC2) έχουν μεγαλύτερα χρονικά παράθυρα και χρονικό ορίζοντα. Τα οχήματα έχουν αυξημένη χωρητικότητα (Repoussis, Tarantilis, & Ioannou, 2009). Έτσι, λόγω των λιγότερων οχημάτων που γίνεται χρήση, τα προβλήματα αυτά έχουν περισσότερα κοινά χαρακτηριστικά με τα TSP και επίσης είναι αρκετά δύσκολα προς επίλυση μέσω αναλυτικών λύσεων (Nagata, Bräysy, & Dullaert, 2010).
- Όλα τα προβλήματα σε κοινή κατηγορία μοιράζονται κοινές συντεταγμένες κόμβων, όχι όμως και χρονικά παράθυρα, τα οποία αλλάζουν από πρόβλημα σε πρόβλημα.

Για την μετατροπή των προβλημάτων σε IRP χρησιμοποιήθηκε σε όλα τα προβλήματα κοινό κόστος αποθεματοποιήσης όπως αυτό δημιουργήθηκε μέσα απο γεννήτρια τυχαίων αριθμών, αφού δεν υπήρχαν τέτοια δεδομένα στα προβλήματα του Solomon. Έτσι όλοι οι πελάτες είχαν κοινό κόστος αποθεματοποίησης.

Κεφάλαιο 5

Υλοποίηση και Εφαρμογή Αλγορίθμου

Η παρούσα διπλωματική είχε σαν στόχο την επίλυση του προβλήματος Δρομολόγησης Αποθεμάτων με τη χρήση χρονικών παραθύρων μέσω του αλγορίθμου της Αποικιας Μυρμηγκιών. Ο προτεινόμενος αλγόριθμος υλοποιήθηκε σε πλατφόρμα προγραμματισμού Matlab 2015b. Το πρόγραμμα εκτελέστηκε σε Laptop HP Pavillion DV6 με επεξεργαστή 2.10 GHz Intel Pentium T4300 και 4GB μνήμη. Ο μέσος χρόνος εκτέλεσης του προγράμματος για την επίλυση κάποιου από τα 56 προβλήματα ήταν περίπου 163 δευτερόλεπτα. Η διαδικασία που πραγματοποιήθηκε είναι η εξής:

1. Επίλυση του εμπλουτισμένου με χρονικούς περιορισμους VRP μέσω ACO.
2. Μετατροπή του προγράμματος επίλυσης του VRP σε πρόγραμμα επίλυσης του προβλήματος IRP με προσθήκη περιορισμών χρονικού ορίζοντα και κόστος αποθεματοποίησης.

Η επίλυση του προβλήματος έγινε με απευθείας επίλυση του VRP και όχι επίλυση του TSP και μετατροπή του σε VRP, καθώς αυτό κρίθηκε άσκοπο να γίνει. Το κάθε μυρμήγκι ολοκληρώνει ένα κύκλο, και αφήνει τη φερομόνη του μετά το πέρας αυτού. Ο κάθε κύκλος εμπεριέχει τους κόμβους που επισκέπτεται το κάθε μυρμήγκι, συμπεριλαμβανομένου και της αποθήκης. Έτσι, το κάθε μυρμήγκι ουσιαστικά δείχνει την πορεία που ακολούθησε για να εξυπηρετηθούν όλοι οι πελάτες. Οι πελάτες που επιλέγονται είναι αυτοί που δύναται να εξυπηρετηθούν και δεν χρησιμοποιείται συνάρτηση penalty, αφού αυτό κρίθηκε οτι θα κατανάλωνε περισσότερο χρόνο για να έρθει το πρόβλημα σε σύγκλιση. Για κάθε μυρμήγκι λοιπόν γίνονται σε κάθε επόμενο στόχο οι παρακάτω έλεγχοι:

- Μπορεί να φτάσει την ώρα που θέτει ο πελάτης;
- Μπορεί να φύγει την ώρα που θέτει ο πελάτης;
- Έχει αρκετό χώρο το φορτηγό για να εξυπηρετηθεί ο επόμενος πελάτης;
- Μπορεί να φτάσει έγκαιρα το φορτηγό απο τον επόμενο πελάτη, πριν κλείσει η αποθήκη;

Οι έλεγχοι αυτοί πραγματοποιούνται μετά που θα λάβει μέρος ο πιθανοτικός κανόνας επιλογής πελατη όπως αυτός περιγράφηκε παραπάνω. Σε περίπτωση που δεν μπορεί να πάει σε κάποιον πελάτη το φορτηγό απο την αφετηρία, δίδεται ένα διάστημα αναμονής 10 μονάδων χρόνου ώστε να βρεθεί κάποιος κατάλληλος. Μετά την επιλογή του επόμενου πελάτη απο την αφετηρία, ή κάποιον κόμβο σημειώνεται το κόστος του τόξου. Όταν ολοκληρωθεί ο κύκλος του φορτηγού (όχι του μυρμηγκιού), κάτι που πραγματοποιείται σε περίπτωση μη εύρεσης εφικτού μελλοντικού κόμβου,όπως αυτό συμπεραίνεται αν δε μπορεί να πραγματοποιηθεί έγκαιρη παράδοση ή επιστροφή πριν το πέρας του χρονικού ορίζοντα, τότε σημειώνεται το κόστος το φορτηγού. Ο κύκλος κλείνει με την επιστροφή του φορτηγού στην αποθήκη, και ο αλγόριθμος φροντίζει πάντα να επιστρέφει πρίν το πέρας του χρονικού ορίζοντα. Η διαδικασία αυτή

επαναλαμβάνεται μέχρι να εξυπηρετηθούν όλοι οι πελάτες. Μετά το πέρας της εξυπηρέτησής τους προσμετράται το κόστος κάθε φορτηγού και συνολικά της διαδρομής. Αυτό το κόστος είναι ουσιαστικά το κόστος του μυρμηγκιού. Το μυρμήγκι αυτό προστίθεται στη λίστα των μυρμηγκιών και αφήνει την φερομόνη στα τόξα τα οποία ακολούθησε. Η φερομόνη εξατμίζεται και ακολουθεί το επόμενο μυρμήγκι. Αν το επόμενο μυρμήγκι έχει χαμηλότερο κόστος τότε αυτό μόνο θα συνεισφέρει στην προσθήκη φερομόνης στα τόξα. Σε κάθε επόμενο μυρμήγκι, μόνο το καλύτερο θα προσθέτει φερομόνη με στόχο να συγκλίνει πιο γρήγορα ο αλγόριθμος, και σε περίπτωση που τυχαία βρεθεί μια καλύτερη λύση να μην χαθεί κάτω απο το βάρος παρόμοιων λύσεων. Στο τέλος της διαδικασίας αυτής, αφού επέλθει η σύγκλιση και για να αποφευχθεί τυχόν τοπικό ελάχιστο που θα χαμήλωνε πολύ την ποιότητα της λύσης ο κύκλος που δημιουργείται περνά απο διαδικασίες τοπικής αναζήτησης. Κατά της διαδικασίες αυτές πραγματοποιείται το εξής:

1. Ο κάθε κόμβος αλλάζει φορτηγό, και δημιουργείται ένας νέος κύκλος.
2. Δοκιμάζεται σε κάθε φορτηγό η διαδικασία τοπικής αναζήτηση , όπως αυτή θα εξηγηθεί παρακάτω.
3. Ελέγχεται σε κάθε επανάληψη η εφικτότητα.
4. Αν η τελική λύση μετά απο τις τοπικές αναζητήσεις είναι και εφικτή αλλά και φθηνότερη απο την αρχική, διατηρείται.

Σκοπός της διαδικασίας αυτής δεν είναι μόνο να μειώσει το κόστος αλλά και τον αριθμό των φορτηγών που γίνονται χρήση. Για να γίνει πιο κατανοητή η διαδικασία τοπικής αναζήτησης θα δοθεί η μορφή της τελικής λύσης του VRP, όπως αυτή προέρχεται απο την αρχική επίλυση και τα βήματα που ακολουθούνται. Η όλη διαδικασία δεν διαφέρει με αυτήν που περιγράφηκε παραπάνω, αρκεί να υπάρξουν οι κατάλληλες μετατροπές στις οποίες το μυρμήγκι να επιστρέφει στην αποθήκη, καθώς και η επιλογή του καλύτερου μυργηγκιού ως αυτού που συνεισφέρει στη λύση. Έτσι ολοκληρώνεται το VRP κομμάτι της επίλυσης. Θα πρέπει να αναφερθεί οτι η τιμή της αρχικής φερομόνης είναι κοινή σε όλες τις εκτελέσεις και ισούται με τον μοναδιαίο πίνακα καθώς αυτό βρέθηκε οτι δίνει καλύτερα γενικά αποτελέσματα σε σχέση με τον τύπο $\tau_{ij} = \frac{m}{TC}$, με TC να είναι το κόστος που βγαίνει απο το TSP του κάθε προβλήματος όταν αυτό λύνεται χωρίς χρονικούς περιορισμούς και κάνοντας χρήση του αλγόριθμου του Πλησιεστερου Γείτονα. Για την μετατροπή του σε IRP, λαμβάνεται η λύση του VRP και με διαδικασία που θα περιγραφεί, επιλύεται το πρόβλημα. Έτσι διατηρούνται οι χρονικοί περιορισμοί και βελτιστοποιείται το κόστος αποθεματοποιήσης.

Τοπική Αναζήτηση

Για να δείχθεί η διαδικασία της διπλής τοπικής αναζήτησης που πραγματοποιείται δίδεται μια ενδεικτική λύση απο τα προβλήματα που επιλύθηκαν. Στόχος της τοπικής αναζήτησης δεν είναι μόνο να μειωθεί το κόστος αλλά και να μειωθούν τα φορτηγά. Έστω ότι η τελική λύση του VRP, πριν την τοπική αναζήτηση έχει αυτή τη μορφή:

Πίνακας 1: Αρχική μορφή λύσης

1	69	66	50	56	55	60	54	57	59	61	58	41	45	47	46	51	52	53	48	44	43	42	49	84	83	88	1						
1	68	64	63	75	73	62	65	67	70	6	76	3	99	2	4	8	89	92	87	97	86	77	72	71	74	81	80	82	79	78	96	95	1
1	94	23	25	28	31	30	35	32	34	38	36	39	40	37	27	29	13	24	19	20	17	15	1										
1	21	22	1																														
1	100	101	98	93	1																												
1	7	33	1																														
1	5	90	1																														
1	85	1																															
1	16	18	14	26	10	12	11	1																									
1	91	9	1																														

Όπου κάθε γραμμή είναι ένα ξεχωριστό φορτηγό, και σε κάθε κελί είναι ένας πελάτης/κόμβος. Ο κόμβος 1 συμβολίζει την αποθήκη. Σκοπός της τοπικής αναζήτησης είναι να βρεί κάποια νέα διαφορετικη λύση ώστε να χαμηλώσει περαιτέρω το κόστος. Στην συγκεκριμένη εφαρμογή έγινε αρχικά μετακίνηση του τελευταίου κόμβου στο τελευταίο φορτηγό στην αρχή του πρώτου φορτηγού. Δηλαδή :

Πίνακας 2: Αρχική Μορφή φορτηγού

1	9	69	66	50	56	55	60	54	57	59	61	58	41	45	47	46	51	52	53	48	44	43	42	49	84	83	88	1

Εν συνεχεία σειριακά αλλάζει εναλλάξ με τον κάθε κόμβο, σε μια διαδικασία swap. Δηλαδή:

Πίνακας 3: Πρώτο Swap

1	69	9	66	50	56	55	60	54	57	59	61	58	41	45	47	46	51	52	53	48	44	43	42	49	84	83	88	1

Πίνακας 4:Δεύτερο Swap

1	69	66	9	50	56	55	60	54	57	59	61	58	41	45	47	46	51	52	53	48	44	43	42	49	84	83	88	1

Και συνεχίζεται μέχρι τέλους αυτή η δαδικασία. Σε περίπτωση που βρεθεί κάποια εφικτή λύση με λιγότερο κόστος τότε διατηρείται. Στη συνέχεια, εισέρχεται ο κόμβος αυτός δοκιμάζεται στο δεύτερο φορτηγό, αντιστοίχως με swap με όλους τους κόμβους, μετά πάει στο τρίτο φορτηγό κλπ. Η διαδικασία της μεταφοράς κόμβων απο το ένα φορτηγό στο άλλο συμβαίνει για όλους τους κόμβους μεταξύ όλων των φορτηγών, καθώς και το swap. Στο τέλος της διαδικασίας θα διατηρηθεί η διαδρομή με το μικρότερο κόστος. Επιπλέον για να μειωθεί όσο το δυνατόν περισσότερο ο αριθμός των φορτηγών, συγχωνεύονται όλα τα φορτηγά σειριακά, γινόταν swap μεταξύ όλων των κόμβων των συγχωνευμένων φορτηγών και ανάλογα με το αν είναι εφικτή και χαμηλότερου κόστους η νέα διαδρομή, διατηρείται. Δηλαδή:

Πίνακας 5: Συγχώνευση πρώτου με τελευταίο φορτηγό

1	91	9	69	66	50	56	55	60	54	57	59	61	58	41	45	47	46	51	52	53	48	44	43	42	49	84	83	88	1

Στο τέλος της διαδικασίας αυτής το κόστος έχει μειωθεί σχετικά με το προηγούμενο και μάλιστα υπάρχουν περιπτώσεις όπου έγινε μείωση και του αριθμού των φορτηγών. Οι χρονικοί περιορισμοί λαμβάνονται διαρκώς υπόψιν και η διαδικασία σταματάει οταν το κόστος δεν κατεβαίνει περαιτέρω. Μετά το πέρας της περιγραφείσας διαδικασίας το τελικό δρομολόγιο έχει γίνει ως εξής:

Table 6: Τελική μορφή της λύσης

1	69	66	50	56	55	60	54	57	59	61	58	41	45	47	46	51	52	53	48	44	43	42	49	84	83	88	1						
1	68	64	63	75	73	62	65	67	70	6	76	3	99	2	4	8	89	92	87	97	86	77	72	71	74	81	80	82	79	78	96	95	1
1	94	23	25	28	31	30	35	32	34	38	36	39	40	37	27	29	13	24	19	20	17	15	1										
1	21	22	1																														
1	100	101	98	93	1																												
1	7	33	1																														
1	5	90	1																														
1	85	1																															
1	16	18	14	26	10	12	11	9	1																								
1	91	1																															

Όπου ο κόμβος 9 έχει μετακινηθεί στο τέλος του προτελευταίου φορτηγου. Με το παραπάνω παραδειγμα ολοκληρώνεται και η διαδικασία της τοπικής αναζήτησης στο συγκεκριμένο πρόγραμμα.

Ρύθμιση Αλγορίθμου

Όπως αναφέρθηκε παραπάνω ο αλγόριθμος αποτελείται απο 4 μεταβλητά στοιχεία τα οποία μπορουν να επηρρεάσουν την ικανότητα του να βρίσκει ποιοτικές λύσεις:

- Ο εκθέτης α, που δείχνει το πόσο επηρεάζει η ευρετική λύση τον πιθανοτικό κανόνα.
- Ο εκθετης β, που δείχνει το πόσο επηρεάζει η φερομόνη τον πιθανοτικό κανόνα.
- Ο ρυθμος εξάτμισης φερομόνης ρ.
- Ο αριθμός των μυρμηγκιών που ξεκινάνε απο την αφετηρία.

Για να γίνει όσο το δυνατό καλύτερη ρύθμιση του αλγορίθμου έγιναν κάποιες εξαντλητικές δοκιμές-πειράματα στην πρώτη κλάση καθε κατηγοριας, δηλαδή δοκιμάστηκαν τα προβλήματα C101,C201,R101,R201,RC101,RC201, με μυρμήγκια απο 1 έως 101, με βήμα 1, οι μεταβλητές α,β να αλλάζουν απο 1 έως 10 αντίστοιχα, και η φερομόνη να αλλάζει τιμή εξάτμισης απο 0.01 έως 0.5 με βήμα 0.01. Συνολικά για ένα πρόβλημα δηλαδή γίνονταν 505.000 διαδοχικά πειράματα. Μέσα απο αυτή τη διαδικασία βρέθηκε ότι οι καλύτερες τιμές αναφορικά με το κόστος βρίσκονται οταν τα μυρμήγκια ειναι απο 40 έως 70, οι μεταβλητές α,β απο 2 έως 6 ενώ η τιμή του ρυθμού εξάτμισης της φερομόνης κυμαίνεται από 0.1 έως 0.5. Έτσι, με επιλογή κατάλληλου βήματος ανά μεταβλητή επιτεύχθει η βέλτιστη σειρά πειραμάτων ανά πρόβλημα, με τα πειράματα να είναι 145 ανά πρόβλημα. Πιθανόν σε πιο εξαντλητικά πειράματα να υπάρξει περαιτέρω πτώση του κόστους των διαδρομών, αλλά αυτό θα γίνει εις βάρος του χρόνου. Τα αποτελέσματα που θα δειχθούν όμως είναι μια καλή εικόνα για το που κυμαίνεται ο αλγόριθμος αναφορικά με το κόστος. Θα πρέπει να σημειωθεί οτι σε κάθε εκτέλεση, με τις ίδιες ρυθμίσεις ο αλγόριθμος απέδιδε λίγο διαφορετικά κάθε φορά, λογω του πιθανοτικού κανόνα. Η διαφοροποιήση αυτή δεν είναι μεγαλύτερη της τάξης του 0.05-0.5% ανά εκτέλεση, οπότε θα δειχθεί ουσιαστικά στα αποτελέσματα η καλύτερη εκτέλεση που έχει βρεθεί για όλα τα πειράματα, σε κάθε διαφορετικό πρόβλημα.

Μετατροπή σε πρόβλημα Δρομολόγησης Αποθεμάτων (IRP)

Για να γίνει η μετατροπή του προβλήματος απο VRP σε IRP αναπαράγεται η λύση του προηγούμενου προβλήματος για το χρονικό ορίζοντα που έχει οριστεί. Θα πρέπει να σημειωθεί οτι όλες οι λύσεις του IRP περνάνε απο το φάσμα της μετατροπής, όμως επιλέγεται αυτή που έχει το μικρότερο κόστος. Έτσι δεν προκαλεί ερώτημα η διαφορά βέλτιστης λύσης VRP και IRP, αφού ένα καλό ημερήσιο δρομολόγιο μπορεί σε βάθος χρόνου να έχει μεγάλο κόστος αποθεματοποίησης. Ακολουθεί η διαδικασία βελτιστοποίησης δύο βημάτων. Αρχικά, πραγματοποιείται μια κατάταξη των πελατών σε φθίνουσα σειρά, λαμβάνοντας ως βάση της κατάταξης το γινόμενο της ημερήσιας ζήτησης και του κόστους αποθεματοποίησης του εκάστοτε

πελάτη της λύσης. Εν συνεχεία, ο πρώτος στην κατάταξη πελάτης μετατοπίζεται στο τέλος του χρονικού ορίζοντα, με περιορισμό να έχει μόνιμα απόθεμα ο πελάτης. Επιπλέον ελέγχεται για κάθε μέρα ο περιορισμός χωρητικότητας και σε περίπτωση που η λύση είναι εφικτή υπολογίζεται το κόστος της. Σε περίπτωση που βρεθεί πλεονάζον φορτίο σε κάποια μέρα, εντοπίζεται η ποσότητά του καθώς και η μέρα στην οποία συμβαίνει αυτό. Για εκείνη τη μέρα υπολογίζεται η απολυτη διαφορά μεταξύ μεταφερόμενης ποσότητας και πλεονάζοντος φορτίου και επιλέγεται ο πελάτης με την ελάχιστη διαφορά. Η ποσότητα του συγκεκριμένου πελάτη για εκείνη τη μέρα μεταφέρεται στη μέρα που το όχημα βρίσκεται πιο άδειο. Η παραπάνω διαδικασία επαναλαμβάνεται μέχρι να βρεθεί μια εφικτή λύση. Σε περίπτωση που κρίνεται ότι δεν πρόκειται να υπάρξει λύση στη συγκεκριμένη διαδικασία, τότε απορρίπτεται εκείνη η μέρα και ξεκινά για κάποια άλλη μέρα και η λύση επιστρέφει στην προηγούμενη βέλτιστη. Επιπλέον, η διαδικασία αυτή λαμβάνει χώρα για κάθε πελάτη, σε κάθε όχημα, σειριακά. Όταν τελειώσει αυτή η διαδικασία, συνεχίζεται η βελτιστοποίηση, βάσει της προηγούμενης λύσης. Αφού εξεταστεί η διαθεσιμότητα του χώρου ανά ημέρα και το ποσό της χωρητικότητας, εντοπίζονται οι πελάτες οι οποίοι έχουν ζήτηση μικρότερη ή ίση με το διαθέσιμο χώρο στο όχημα. Προτεραιότητα δίνεται στους πελάτες οι οποίοι δύναται να μετοπιστούν ποσότητες προς μεταφορά για παραπάνω απο μια μέρες. Αν ούτε αυτό είναι εφικτό, επιλέγεται ο πελάτης που θα καλύψει περισσότερο χώρο στο όχημα. Η διαδικασία ολοκληρώνεται ελέγχοντας πάντα τους περιορισμους χωρηρικότητας των οχημάτων χωρίς να μένει κάποιος πελάτης χωρίς απόθεμα. Για να γίνει κατανοητή η διαδικασία ακολουθεί παράδειγμα όπως αυτο παρουσιάζεται μετά τη χρήση του αλγορίθμου. Έτσι έστω οτι η λύση του VRP για ένα όχημα, η οποία έχει το φθήνοτερο κόστος έχει αυτή τη μορφή:

1	98	101	100	3	2	76	1

,η οποία μετατρέπεται σε

1	98	101	100	3	2	76	1
0	30	20	10	30	10	20	0
0	90	80	50	120	10	20	0

Όπου η δεύτερη σειρά δείχνει την ζήτηση του πελάτη (0 για την αποθήκη προφανώς) και η τρίτη σειρά εμπεριέχει το αποτέλεσμα του πολλαπλασιασμού με το κόστος αποθεματοποίησης για να γνωρίζουμε το κόστος που έχει η διατήρηση του αποθέματος απο τον πελάτη κάθε μέρα. Εν συνεχεία, δημιουργείτε ένας πίνακας για το συνολικό πρόγραμμα του χρονικού ορίζοντα ως εξής:

30	20	10	30	10	20	30
30	20	10	30	10	20	30
30	20	10	30	10	20	30
30	20	10	30	10	20	30
30	20	10	30	10	20	30
30	20	10	30	10	20	30
30	20	10	30	10	20	30

Στο τέλος της διαδικασίας που περιγράφηκε το τελικό πρόγραμα γίνεται ώς εξής:

0	0	0	0	20	20	0
30	20	10	30	10	20	30
30	20	10	30	10	20	30
30	20	10	30	10	20	30
30	20	10	30	10	20	30
30	20	10	30	10	20	30
60	40	20	60	0	20	60

Στην λύση του IRP όπως αυτή δημιουργείται απο τον αλγόριθμο, στα σημεία που αναγράφεται 0 σημαίνει οτι το φορτηγό δεν επισκέπτεται τον πελάτη. Αυτό που συμβαίνει είναι οτι το φορτηγό αναμένει στη θέση του ώστε να ανοίξει ο επόμενος πελάτης, έτσι δεν αλλάζει κάτι στο κόστος.

Κεφάλαιο 6

Σε αυτό το κεφάλαιο παρατίθενται τα αποτελέσματα του αλγορίθμου για τα προβλήματα της δρομολόγησης οχημάτων (VRP) και της Δρομολόγησης Αποθεμάτων (IRP) που ήταν και ο τελικός στόχος αυτής της εργασίας. Ελέγχθηκαν διάφορες τιμές για τις παραμέτρους του προβλήματος, όπως αυτό αναφέρθηκε και παραπάνω και επιλέχθηκαν αυτές με τα καλύτερα αποτελέσματα όσον αφορά την ποιότητα της λύσης και τον υπολογιστικό χρόνο που χρειάστηκε για την εύρεση της. Για το πρόβλημα του VRP η τιμή των παραμέτρων αυτών πήρε διαφορετική τιμή ανά πρόβλημα, τιμή που βρέθηκε μετά το πέρας πειραμάτων. Τέλος, στο πρόβλημα IRP το απόθεμα των πελατών πριν την έναρξη του κύκλου, από παραδοχή πήρε την τιμή της ημερήσιας ζήτησης καθενός πελάτη. Η παραδοχή αυτή έγινε λόγω έλλειψης στοιχείων από τη βιβλιογραφία.

Αποτελέσματα προβλήματων για το VRP

Πρόβλημα	Αριθμός Οχημάτων	Αριθμός μυρμηγκιών	Παράγοντας α	Παράγοντας β	Ρυθμός Εξάτμισης	Κόστος	Απόκλιση απο βέλτιστο
C101	11	29	4	2	0.2	839.3	+5.4%
C102	12	61	6	6	0.1	893.2	+7.1%
C103	14	60	6	6	0.3	1192.4	+38.08
C104	12	71	5	5	0.1	1090.3	+32.5%
C105	12	26	3	3	0.3	876.1	+6.1%
C106	11	91	4	4	0..3	885.7	+7 %
C107	12	91	5	3	0.05	855.7	+3.38
C108	11	101	5	5	0.05	899.5	+8.7%
C109	11	86	5	3	0.1	918.6	+11%
C201	4	61	2	3	0.1	612.9	+3.9%
C202	4	61	6	4	0.1	713.4	+23.6%
C203	9	51	4	4	0.1	900	+46.5%
C204	9	51	2	4	0.1	832.2	40.77%
C205	4	71	6	6	0.3	619.7	+5.6%
C206	4	61	6	2	0.2	653.4	+11.4%
C207	4	51	6	6	0.1	692.1	+18%
C208	4	51	6	6	0.1	663.6	+13%
R101	25	61	4	4	0.1	2351.8	+45%

R102	24	61	6	4	0.3	1966.0	32.3%
R103	21	51	6	2	0.1	1530.3	26.6%
R104	18	51	6	4	0.1	1559.8	60.6%
R105	22	51	4	2	0.1	1704.3	25.7%
R106	15	61	6	6	0.2	1661.3	34.5%
R107	17	51	4	2	0.1	1780.8	67.2%
R108	13	51	6	6	0.1	1338.0	43.5%
R109	18	51	2	2	0.2	1502.3	30.9%
R110	16	61	2	4	0.1	1431.3	34.0%
R111	15	81	6	4	0.1	1476.2	40.7%
R112	12	71	6	6	0.2	1203.7	26.8%
R201	7	21	6	4	0.1	1468.8	30.0%
R202	7	81	6	4	0.1	1736.8	68.6%
R203	7	81	6	4	0.4	1175.6	35.1%
R204	10	71	6	2	0.2	1221.9	56.2%
R205	9	71	6	4	0.3	1513.3	58.2%
R206	4	71	6	2	0.4	1351.9	49.1%
R207	4	71	6	2	0.2	1241.3	37.9%
R208	4	51	6	2	0.5	1053.1	40.4%
R209	7	91	6	2	0.4	1341.4	44.5%
R210	7	51	6	4	0.1	1447	55.5%
R211	4	71	6	4	0.1	1096.4	14.6%
RC101	20	71	6	6	0.3	2225	33.3%
RC102	18	82	6	2	0.3	1992.1	28.3%
RC103	15	71	6	4	0.1	1823.1	40.2%
RC104	13	61	6	6	0.2	1559.3	21.7%
RC105	24	51	2	4	0.3	2305.8	42.4%
RC106	15	51	4	4	0.3	1750.7	20.9%
RC107	16	61	2	4	0.2	1685.1	20.5%
RC108	13	71	4	4	0.1	15382.3	38%
RC201	7	91	6	4	0.1	1691.2	15.9%
RC202	11	61	6	6	0.1	1259	10.9%
RC203	7	61	6	4	0.1	1127.4	36.8%

RC204	4	81	6	4	0.4	948.9	14.3%
RC205	10	41	6	6	0.2	1338.2	12.7%
RC206	6	61	2	4	0.3	1185.3	12%
RC207	6	21	6	2	0.2	1139	15.7%
RC208	4	41	4	6	0.2	990.1	20.7%

Για το πρόβλημα του VRP χρησιμοποιήθηκαν όλα τα προβλήματα του Solomon. Επιπλέον, τα αποτελέσματα έγιναν σύγκριση με τα βέλτιστα όπως αυτά βρέθηκαν στην διατριβη του Δ.Δημήτρουλα (2012) καθώς και γενικότερα στη βιβλιογραφία.

Αποτελέσματα προβλήματος IRP

Το IRP πραγματοποιήθηκε με χρονικό ορίζοντα 1 εβδομάδα και λήφθηκε σαν βέλτιστη λύση αυτή με το χαμηλότερο κόστος στο βάθος του χρονικού ορίζοντα. Έτσι, όπως θα παρατηρηθεί και παρακάτω, βέλτιστη λύση του IRP διαφέρει απο αυτή του VRP σε μεγάλο βαθμό.

Πρόβλημα	Αριθμός Οχημάτων	Αριθμός μυρμηγκιών	Παράγοντας α	Παράγοντας β	Ρυθμός Εξάτμισης	Κόστος Αποθεματοποίησης	Κόστος Μεταφοράς
C101	12	17	2	5	0.2	37540	44516
C102	18	11	4	2	0.3	35610	45647
C103	15	61	4	6	0.3	37100	46974
C104	14	81	5	5	0.5	38280	47049
C105	12	26	2	2	0.55	37650	45510
C106	13	6	2	4	0.5	37120	45129
C107	11	61	2	2	0.4	35810	44357
C108	12	26	2	3	0.2	36670	44367
C109	12	26	5	5	0.5	38230	46065
C201	5	21	2	5	0.5	31830	37337
C202	10	61	4	4	0.1	29860	37229
C203	11	51	4	4	0.3	29300	37895
C204	14	51	4	4	0.3	30780	42507
C205	6	61	2	2	0.4	31570	36742
C206	9	81	2	6	0.2	32050	38360

C207	8	61	2	2	0.2	29430	36670
C208	9	51	2	4	0.1	30320	37144
R101	24	61	4	2	0.1	21072	35591
R102	24	61	6	4	0.3	22116	36529
R103	24	61	4	4	0.4	21834	36005
R104	17	51	4	2	0.5	24744	34799
R105	22	51	2	4	0.4	20872	35062
R106	17	51	4	4	0.3	23352	33849
R107	18	51	2	4	0.1	23449	35328
R108	14	51	4	4	0.3	24395	33096
R109	19	81	2	4	0.3	20426	32495
R110	19	81	2	4	0.4	20479	31947
R111	18	61	2	2	0.5	22071	33473
R112	15	71	2	6	0.4	21641	32614
R201	13	31	2	2	0.3	31012	43155
R202	8	101	6	4	0.4	18782	28256
R203	8	61	6	6	0.4	19601	28457
R204	14	61	6	4	0.3	31242	40696
R205	11	51	6	2	0.2	32698	44331
R206	6	51	5	2	0.1	18971	27195
R207	4	101	6	2	0.2	21080	28620
R208	5	51	4	4	0.3	20283	27916
R209	5	91	6	4	0.1	18714	26603
R210	6	101	4	4	0.4	18856	27815
R211	4	61	4	2	0.1	18578	25798
RC101	20	71	6	6	0.3	22005	34540
RC102	18	81	6	2	0.3	21891	33278
RC103	20	71	2	4	0.4	28264	41701
RC104	15	61	2	4	0.4	28457	41574
RC105	20	81	4	4	0.1	24781	40246
RC106	16	51	4	4	0.1	29112	40651
RC107	17	101	2	6	0.5	27208	40775
RC108	16	71	2	2	0.2	28617	41508

RC201	7	91	6	4	0.1	21947	34250
RC202	11	31	6	2	0.5	33186	44244
RC203	7	71	2	4	0.2	22448	34927
RC204	10	91	2	2	0.1	22611	33120
RC205	10	31	6	2	0.4	22307	34190
RC206	6	11	2	4	0.2	21941	34756
RC207	7	41	6	4	0.1	22080	32084
RC208	4	101	6	6	0.2	24907	32470

Συζήτηση αποτελεσμάτων

Η συζήτηση περί αποτελεσμάτων έχει επικέντρωση κυρίως στο κομμάτι του IRP, μιας και αυτό είναι το κύριο κομμάτι της εργασίας που εκπονήθηκε. Κάποια σημαντικά όμως σημεία που αφορούν το VRP θα παρουσιαστούν.

Έτσι, όπως δυναται να παρατηρηθεί δια της απλής εποπτείας τα αποτελέσματα του VRP δεν είναι ενθαρρυντικά. Η σχετικά μεγάλη απόκλιση, παρά το σχετικά γρήγορο χρόνο επίλυσης προκαλεί προβληματισμό, κυρίως για την υλοποιήση. Όμως ο χρόνος εκτέλεσης είναι αρκετά μικρός. Επιπλέον, όπως είναι λογικό, τα αποτελέσματα ποικίλουν ανάλογως την πολυπλοκότητα του προβλήματος. Έτσι, το VRP στα τύπου C1 προβλήματα λειτουργεί σε ικανοποιητικό βαθμό, λόγω του μικρού αριθμού λύσεων που είναι εφικτές, όπως αυτό περιορίζεται λόγω χωρητικότητας και χρόνου. Όμως στην πλειοψηφία των C2 η απόκλιση εκτοξεύεται, ενώ στα πιο περίπλοκα R δεν δείχνει να δουλεύει τόσο καλά, αφού η τυχαιότητα της κατανομής τον κόμβων δείχνει να επηρρεαζει τον αλγόριθμο. Το ίδιο συμβαίνει και στα προβλήματα RC, όπου ναι μεν τα αποτελέσματα είναι ελάχιστα πιο ενθαρρυντικά, αλλά και πάλι δεν είναι αυτά που θα θέλαμε. Επιπλέον μπορεί να παρατηρηθεί οτι όσο μεγαλύτερη γκάμα κόμβων έχει να ψάξει ο αλγόριθμος (βλ. Προβλήματα τύπου 2) τόσο χειρότερα αποδίδει στο VRP. Όμως η μετατροπή σε IRP και ο σεβασμός στους χρονικούς περιορισμούς είναι το κύριο θετικό. Η έλλειψη στοιχείων για σύγκριση σε αυτόν τον τύπο προβλήματων δεν επιτρέπει ασφαλή συμπεράσματα. Επιπλέον, η ευρετική πληροφορία φαίνεται να είναι αρκετά σημαντική στην ανάκτηση καλών αποτελεσμάτων συγκριτικά με την φερομόνη.

Κεφάλαιο 7

Σύνοψη

Στην παρούσα εργασία όπως αυτή περιγράφηκε, επιχειρήθηκε η βελτιστοποίηση ενός προβλήματος που στοιχεία του δεν υπήρχαν στη βιβλιογραφία, αυτό του IRP με περιορισμούς χρόνου. Στα πλαίσια αυτής περιγράφηκαν τα προβλήματα του Πλανόδιου Πωλητή, της Δρομολόγησης Οχημάτων και της Δρομολόγησης Αποθεμάτων. Επιπλέον, παρουσιάστηκε η Νοημοσύνη Σμήνους σαν πλαίσιο επίλυσης προβλημάτων βελτιστοποιήσης διαδρομών και γενικότερα, συνδυαστικής βελτιστοποίησης. Μέσα από αυτό το πλαίσιο επιλέχθηκε ο αλγόριθμος Βελτιστιστοποιήσης Αποικίας Μυρμηγκιών. Έπειτα δημιουργήθηκε σε περιβάλλον Matlab αντίστοιχο πρόγραμμα το οποίο επιλύει τα προβλήματα του Solomon. Ο αλγόριθμος επιλύει πρώτα το πρόβλημα του VRP και μετά το μετατρέπει σε αντίστοιχη λύση για το IRP προσθέτοντας τον χρονικό ορίζοντα και το κόστος αποθεματοποίησης. Δεν λαμβάνεται κάποια προηγούμενη επίλυση σαν βάση του αλγορίθμου καθώς αυτο βρέθηκε να δίνει χειρότερα αποτελέσματα απο τον να μπαίνει σαν τιμή βάσης ο μοναδιαίος πίνακας για όλες τις επιλύσεις και όλους τους πελάτες. Η βέλτιστη λύση του VRP διαφέρει απο αυτή απο την οποία παράγεται η βέλτιστη του IRP. Τα αποτελέσματα του VRP, αν και παράγονται αρκετα γρήγορα, παρουσιάζουν απόκλιση απο τη βέλτιστη της βιβλιγοραφίας. Αναφορικά με το IRP, λόγω του οτι δεν υπάρχει κάτι αντίστοιχο στη βιβλιογραφία δεν ήταν δυνατό να γίνει σύγκριση με πρότερα αποτελέσματα.

Συμπεράσματα

Απο τα αποτελέσματα για το VRP συμπεραίνεται οτι ο αλγόριθμος έχει περιθώρια βελτίωσης τόσο στην ποιότητα των λύσεων όσο και στο χρόνο εκτέλεσης. Η τοπική αναζήτηση λειτουργεί ικανοποιητικά, αλλά η χρήση του swap μπορεί να διευρυνθεί και να δώσει ακόμη καλύτερα αποτελέσματα. Αναφορικά με τα αποτελέσματα του IRP, δείχνουν ενδιαφέροντα, αφού εκτελούνται εντός των χρονικών περιορισμών και ξεκινάνε απο διαδρομή με χαμηλό σχετικά κόστος, όμως ελλείψη βιβλιογραφικών αποτελεσμάτων, η ποιότητα τους είναι δύσκολο να ελεγχθεί. Συνολικά ο αλγόριθμος φαίνεται να λειτουργεί σωστά και σε καλό χρόνο. Η ικανότητα του να βρίσκει εφτικτές και καλές λύσεις κρίνεται ενδιαφέρουσα για το VRP, και η μετατροπή σε IRP με σεβασμό στους χρονικούς περιορισμούς είναι σχετικά πρωτοποριακή όπως προκύπτει απο την βιβλιογραφία.

Μελλοντικές Εργασίες

Τα αποτελέσματα της συγκεκριμένης εργασίας και οι διαδικασίες με τις οποίες διεκπεραιώνεται η βελτιστοποίηση δείχνουν περιθώρια σαφούς μελέτης στο μέλλον. Αλλαγές όπως τοπική αναζήτηση μεγάλης κλίμακας σε κάθε μυρμήγκι, με κόστος το χρόνο επίλυσης ανά επανάληψη δύναται να λάβουν χώρα. Επιπλέον θα μπορούσε να γίνει επανεπίλυση του προβλήματος, μετά την έκδοση της IRP λύσης, με τη χρήση κάποιου άλλου μεθευρετικού αλγορίθμου, ο οποίος θα λαμβάνει σαν είσοδο του ημερήσιους διαθέσιμους πελάτες, τα χρονικά τους παράθυρα και τη ζήτησή τους και θα αναλάμβανε να βγάλει ακόμη καλύτερα ημερήσια προγράμματα. Όμως αυτό θα είχε ξανά αντίκτυπο στο συνολικό χρόνο επίλυσης αν και θα έκανε μεγάλες βέλτιώσεις στο τελικό εβδομαδιαίο πρόγραμμα. Πιθανό, με κάποια άλλη αρχική επίλυση για το TC, να υπήρχει καλύτερη βάση για την φερομόνη και ο αλγόριθμος να είχε καλύτερα αποτελέσματα, κάτι που θα μπορούσε να διερευνυθεί περαιτέρω. Έτσι, να βρεθεί ποιά είναι η βέλτιστη βάση για κάθε πρόβλημα. Επίσης, παραλλαγές όπως το να καταθέτουν ορμόνη παραπάνω απο 1 βέλτιστο μυρμήγκι πάντα με κόστος το χρόνο επίλυσης θα μπορούσαν να αποτελέσουν παραλλαγές με ενδιαφέρον. Επιπλέον, θα μπορούσε να γίνει χρήση του αλγορίθμου σε πλατφόρμα που έχει σχεδιαστεί στο παρελθόν για βελτιστοποίηση διαδρομών (Μαρκουλάκης, 2013), με στόχο να λαμβάνεται το καλύτερο δυνατό αποτέλεσμα για τον χρήστη-βιομηχανία.

Βιβλιογραφία

Bakhouya, M., & Gaber, J. (2007). An Immune Inspired-based Optimization Algorithm:Application to the Traveling Salesman Problem. *Advanced Modeling and Optimization, 9*(1), 105-116.

Belal, M., Gaber, J., El-Sayed, H., & Almojel, A. (2006). *Swarm Intelligence, In Handbook of Bioinspired Algorithms and Applications. Series: CRC Computer & Information Science.* (Τόμ. 7). Chapman & Hall Eds.

Beni, G., & Wang, J. (1989). Swarm intelligence in cellular robotic systems. *NATO Advanced Workshop on Robots and Biological Systems.* Tuscany.

Bienert, P., Rechenberg, I., & Schwefel, H. P. (1966). Messung kleiner Wandschubspannungen bei turbulenten Grenzschichten in Ablösenähe. *Interner Bericht des Hermann Föttinger-Instituts für Strömungstechnik*.

Blum, C., Roli, A., & Dorigo, M. (2001). HC–ACO: The hyper-cube framework for Ant Colony Optimization. *Metaheuristics International Conference*, *2*, σσ. 399-403.

Bonabeau, E., & Meyer, C. (2001). Swarm Intelligence: A Whole New Way to Think About. *Harvard Business Review, 79*(5), σσ. 106-114.

Bonabeau, E., Dorigo, M., & Theraulaz, G. (1999). *Swarm Intelligence: From Natural to Artificial Systems.* Oxford: Oxford University Press.

Bonabeau, E., Dorigo, M., & Theraulaz, G. (2000). Inspiration for optimization from social insect behaviour. *Nature*, *406*, 39–42.

Bullnheimer, B., Hartl, F. R., & Strauss, C. (1999). A new rank-based version of the Ant System: A computational study. *Central European Journal for Operations Research and Economics, 7*(1), 25-38.

Chu, S. C., Tsai, P. W., & Pan, J. S. (2006). Cat swarm optimization. *9th Pacific Rim International Conference on Artificial Intelligence*, (σσ. 854-858).

Colorni, A., Dorigo, M., Maniezzo, V., & Trubian, M. (1994). Ant System for Job-shop Scheduling. *Belgian Journal of Operations Research, Statistics and Computer Science, 34*(1), 39-54.

Das, S., Abraham, A., & Konar, A. (2008). Swarm Intelligence Algorithms in Bioinformatics. *Studies in Computational Intelligence, 94*, 119-127.

Das, S., Panigrahi, B. K., & Pattnaik, S. S. (2009). Nature-Inspired Algorithms for Multi-objective Optimization. Στο *Handbook of Research on Machine Learning Applications and Trends: Algorithms Methods and Techniques* (Τόμ. 1, σσ. 95–108). New York: Hershey.

De Jong, A. (1975). *Analysis of the behavior of a class of genetic adaptive systems.* Engineering. University of Michigan.

Dorigo, M. (1992). *Optimization, learning and natural algorithms.* Politecnico di Milano, Dipartimento diElettronica, Milan.

Dorigo, M., & Gambardella, L. M. (1995). Ant-Q: A Reinforcement Learning Approach to the Traveling Salesman Problem. Στο A. P. Russell (Επιμ.), *12th International Conference on Machine Learning* (σσ. 252-260). Tahoe City: Morgan Kaufmann.

Dorigo, M., & Socha, K. (2008). Ant colony optimization for continuous domains. *European Journal of Operations Research, 185*(3), 1155-1173.

Dorigo, M., Bonabeau, E., & Theraulaz, G. (2000). Ant algorithms and stigmergy. *Future Gener. Comput. Syst., 16*(8), 851–871.

Dorigo, M., Maniezzo, V., & Colorni, A. (1991). *Positive feedback as a search strategy.* Politecnico di Milano, Dipartimento di Elettronica. Milan: Citeseer.

Dorigo, M., Maniezzo, V., & Colorni, A. (1996). Ant System: Optimization by a colony of cooperating agents. *Systems, Man, and Cybernetics. 26*, σσ. 29-41. IEEE.

Eberhart, R. C., & Kennedy, J. (1995). A new optimizer using particle swarm theory. *Proceedings of the sixth international symposium on micro machine and human science, 1*, σσ. 39-43.

Eberhart, R. C., & Kennedy, J. (1995). Particle Swarm Optimization. Στο IEEE (Επιμ.), *International Conference on Neural Networks*, (σσ. 1942-1948). Perth.

Engelbrecht, A. P. (2002). *Computational Intelligence: An Introduction.* (A. P. Engelbrecht, Επιμ.) England: John Wiley & Sons.

Fogel, L. J., Owens, A. J., & Walsh, M. J. (1966). Artificial Intelligence Through.

Gambardella, M., & Dorigo, M. (1997). Ant Colony System: A cooperative learning approach to the traveling salesman problem. *Transactions on Evolutionary Computation. 1*, σσ. 53-66. IEEE.

Geem, Z. W., Kim, J. H., & Logathan , G. V. (2001). A new heuristic optimization algorithm: harmony search. *Simulation, 76*(2), 60-68.

Glover, F., & Laguna, M. (2013). *Tabu Search.* New York: Springer.

Haldenbilen, S., Cenk, O., & Baskan, O. (2013). *An Ant Colony Optimization Algorithm for Area Traffic Control.* INTECH Open Access Publisher.

History of optimization,. (n.d.). Ανάκτηση από http://www.mitrikitti.fi/opthist.html

Holland, J. (1975). *Adaptation in Natural and Artificial systems.* Ann Arbor: University of Michigan.

Hoos, H. H., & Stützle, H. (1996). *Improving the Ant System: A detailed report on the MAX-MIN Ant System.* TU Darmstadt, FG Intellektik.

Jaiswal, U., & Aggarwal, S. (2011). Ant Colony Optimization. *International Journal of Scientific & Engineering Research, 2*(7), 1-7.

Judea, P. (1984). *Heuristics: intelligent search strategies for computer problem solving.*

Karaboga, D. (2005). *An idea based on honey bee swarm for numerical optimization.* Technical report, Erciyes university, Computer Engineering Department.

Karaboga, D. (2005). *An Idea Based On Honey Bee Swarm for Numerical Optimization.* Technical Report, Erciyes University, Engineering Faculty, Computer Engineering Department, Erciyes.

Karaboga, D., & Basturk, B. (2007). A Powerful And Efficient Algorithm For Numerical Function Optimization: Artificial Bee Colony (ABC) Algorithm. *Journal of Global Optimization, 39*(3), 459-471.

Kennedy, R. C. (1995). A new optimizer using particle swarm theory. *Sixth International Symposium on Micro Machine and Human Science*, (σσ. 39-43). Nagoya.

Kirkpatrick, S., Gelatt, C. D., & Vecchi, M. P. (1983). Optimization by simmulated annealing. science. *Science, 220*(4598), 671-680.

Kohl, n., Desrosiers, J., Madsen, O. B., & Solomon, M. M. (1999, February). 2-path cuts for the vehicle routing problem with time windows. *Transportation Science, 33*(1), 101-116.

Koza, J. R. (1992). *On Genetic Programming: One the Programming of Computers by* (Τόμ. 1). MIT press.

Krishnanand,, K. N., & Ghose, D. (2009). Glowworm swarm optimization for searching. (L. C. C. P. Lim, Επιμ.) *Innovations in Swarm Intelligence*.

Lim, C. P., Jain, L. C., & Dehuri, S. (2009). Studies in Computational Intelligence. *Innovations in Swarm Intelligence, 248*.

McCulloch, W. S., & Pitts, W. (1943). A logical calculus of the ideas immanent in nervous activity. *The bulletin of mathematical biophysics, 5*(4), 115-133.

Merkle, D., & Blum, C. (2008). *Swarm Intelligence – Introduction and Applications. Natural Computing.* Berlin: Springer.

Nagata, Y., Bräysy, O., & Dullaert, W. (2010). A penalty-based edge assembly memetic algorithm for the vehicle routing problem with time windows. *Computers & Operations Research, 37*, 724–737.

Nakrani, S. &. (2004). On honey bees and dynamic server allocation in internet hosting centers. *Adaptive Behavior, 12*(3), 223-240.

Panigrahi, B. K., Yuhui , S., & Meng-Hiot, L. (2011). *Handbook of Swarm Intelligence.* (Τόμ. 7). Berlin: Springer.

Parsopoulos, K. E. (2010). Particle Swarm Optimization and Intelligence: Advances and Applications. *Information Science Reference*.

Passino, K. M. (2002). Biomimicry of Bacteria Foraging for Distributed Optimization. *IEEE Control Systems Magazine, 22*, 55-67.

Pham,, D. T., Ghanbarzadeh,, A., Koc, E., Otri, S., Rahim, S., & Zaimi, M. (2011). The bees algorithm–a novel tool for complex optimisation. *Intelligent Production Machines and Systems-2nd I* PROMS Virtual International Conference.*

Repoussis, P. P., Tarantilis, C. D., & Ioannou, G. (2009). Arc-guided evolutionary algorithm for the vehicle routing problem with time windows. *Transactions on Evolutionary Computation. 13*, σσ. 624-647. IEEE.

Russell, S. J. (2003). *Artificial Intelligence: a modern approach* (Τόμ. 2). Upper Saddle River, New Jersey, USA: Prentice hall.

Schrijver, A. (2005). On the history of combinatorial optimization (till 1960). Στο *Handbooks in operations research and management science* (Τόμ. 12, σσ. 1-68).

Siegelmann, H. T., & Sontag, E. D. (1991). Turing computability with neural nets. *Applied Mathematics Letters, 4*(6), 77-80.

Socha, K. (2004). ACO for continuous and mixed-variable optimization. Στο L. G. M. Dorigo (Επιμ.), *4th International Conference on Swarm Intelligence*, (σσ. 25-36).

Solomon, M. M. (1987). Algorithms for the vehicle routing and scheduling problems with time window constraints. *Operations Research, 35*(2), 254–265.

Storn, R., & Price, K. (1997). Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces. *Journal of global optimization, 11*(4), 341-359.

Stützle, M. D. (2004). *Ant Colony Optimization.* Cambridge: MIT Press.

Tsutsui, S. (2004). Ant colony optimisation for continuous domains with aggregation pheromones metaphor. *5th International Conference on Recent Advances in Soft Computing*, (σσ. 207-212).

Turing, A. M. (1948). *Intelligent Machinery.* National Physical Laboratory.

Turing, A. M., & Copeland, B. J. (2004). *The essential turing.* Oxford University Press.

Vapnik, V. (2013). *The nature of statistical learning theory.* Springer Science & Business Media.

Vapnik, V., Golowich, S. E., & Smola, A. (1997). Support vector method for function approximation, regression estimation, and signal processing. *Advances in neural information processing systems*, 281-287.

Yang, X. S. (2009). Firefly algorithms for multimodal optimization. *International Symposium on Stochastic Algorithms* (σσ. 169-178). Springer Berlin Heidelberg.

Yang, X. S., & Deb, S. (2009). Cuckoo search via Lévy flights. *Nature & Biologically Inspired Computing* (σσ. 210-214). IEEE.

Zhao, N., Wu, Z., Zhao, Y., & Quan, T. (2010). Ant colony optimization algorithm with mutation mechanism and its applications. *Expert Systems with Applications, 37*(7), 4805-4810.

Διονύσιος, Δ. (2012). *Επίλυση προβλήματος δρομολόγησης οχημάτων με χρονικά παράθυρα μέσω Μιμητικού Αλγορίθμου.* Μηχανικών Παραγωγής και Διοίκησης. Χανιά: Πολυτεχνείο Κρήτης.

Μαρκουλάκης, Β. (2013). *Αλγόριθμος περιορισμένης αναζήτησης για το πρόβλημα δρομολόγησης και αποθεματοποίησης.* Χανιά: Πολυτεχνείο Κρήτης.

Μυγδαλάς, Α., & Μαρινάκης, Ι. (2008). *Σχεδιασμός και Βελτιστοποίηση της Εφοδιαστικής Αλυσίδας.* Θεσσαλονίκη: Εκδόσεις Σοφία.