Βέλτιστός Σχεδιασμός Γραμμής Προϊόντων με Χρήση Αλγορίθμων Εμπνευσμένων από τη Φύση

Φραγκογιάννης Ιωάννης

Τομέας Επιχειρησιακής Έρευνας , ΑΕΜ 2015019039

Μεταπτυχιακή εργασία για την εκπλήρωση των απαιτήσεων του ΠΜΣ
Επιχειρησιακή Έρευνα

Τμήμα Μηχανικών Παραγωγής και Διοίκησης

Πολυτεχνείο Κρήτης

Επιβλέπων:
Στέλιος Τσαφαράκης

Ευχαριστίες

Ολοκληρώνοντας με την παρούσα εργασία το Μεταπτυχιακό Πρόγραμμα Επιχειρησιακή Έρευνα, του τμήματος Μηχανικών Παραγωγής και Διοίκησης του Πολυτεχνείου Κρήτης, θα ήθελα να ευχαριστήσω τον επιβλέποντα της εργασίας κ. Στέλιο Τσαφαράκη για την συνεργασία και τις εύστοχες επισημάνσεις του οι οποίες με βοήθησαν να την ολοκληρώσω. Επίσης ένα μεγάλο ευχαριστώ στην οικογένειά μου για την υπομονή και τη συμπαράσταση τους κατά τη διάρκεια των σπουδών μου.

Σύντομο βιογραφικό

Ο Φραγκογιάννης Ιωάννης γεννήθηκε και μεγάλωσε στα Χανιά. Είναι απόφοιτος του τμήματος Μαθηματικών του Πανεπιστημίου Κρήτης και κάνει το μεταπτυχιακό του στην Επιχειρησιακή Έρευνα στο τμήμα Μηχανικών Παραγωγής και Διοίκησης του Πολυτεχνείου Κρήτης. Στα ερευνητικά του ενδιαφέροντα περιέχονται η επιχειρησιακή έρευνα, λήψη αποφάσεων, στατιστική και πιθανότητες καθώς και τα εφαρμοσμένα μαθηματικά και οι εφαρμογές τους.

ΠΕΡΙΕΧΟΜΕΝΑ

1. Εισαγωγή

1.1 Σκοπός της εργασίας

Σκοπός της παρούσας εργασίας είναι να γίνει πλήρως κατανοητό πως τα προϊόντα μιας γραμμής προϊόντων αλληλοεπιδρούν μεταξύ τους και με ποιο τρόπο επηρεάζουν τους καταναλωτές έναντι μιας ανταγωνιστικής γραμμής, με σκοπό να δημιουργηθεί μια γραμμή προϊόντων που θα καλύπτει όσο το δυνατόν μεγαλύτερο εύρος της αγοράς. Στη συνέχεια πρέπει να γίνει κατανοητός ο τρόπος με τον οποίο οι ευρετικοί και μεθευρετικοί αλγόριθμοι συνδυάζουν τα χαρακτηριστικά και τα επίπεδα των χαρακτηριστικών, για να βρεθούν ικανοποιητικές ή βέλτιστες λύσεις. Για αυτό το σκοπό θα χρησιμοποιηθούν ο αλγόριθμος Cuckoo search και ο αλγόριθμός Bat για πρώτη φορά στο εν λόγω πρόβλημα ως μια νέα μέθοδος επίλυσής του. Τα αποτελέσματα των δυο αυτών αλγορίθμων θα συγκριθούν με τα αποτελέσματα των υλοποιήσεων που έχουν γίνει έως τώρα. Σε αυτό το σημείο πρέπει να τονιστεί ότι δεν παράγουν όλοι οι αλγόριθμοι εξίσου καλά αποτελέσματα σε ένα συγκεκριμένο πρόβλημα. Για αυτό το λόγο έχουν αναπτυχθεί τόσοι διαφορετικοί γενετικοί και εξελικτικοί αλγόριθμοι, ώστε να χρησιμοποιείται κάθε φορά διαφορετικός αλγόριθμος ανάλογα το πρόβλημα που θέλουμε να βελτιστοποιηθεί. Θα ελεγχθεί λοιπόν κατά πόσο μπορούν να εφαρμοστούν εν γένει στο πρόβλημα, καθώς και να υπάρξει μια πιο αντικειμενική σύγκριση με τις άλλες υλοποιήσεις.

2. Το πρόβλημα του βέλτιστου σχεδιασμού γραμμής προϊόντων

2.1 Πρόβλημα βέλτιστου σχεδιασμού γραμμής προϊόντων

Στη σημερινή κοινωνία της παγκοσμιοποίησης οι επιχειρήσεις δυσκολεύονται να επιβιώσουν και αντιμετωπίσουν τον ανταγωνισμό. Με τη συνεχή τεχνολογική ανάπτυξή καθώς και την όλο ένα αυξανόμενη επιθυμία των καταναλωτών να αγοράζουν συνεχώς καινούργια και πιο εξελιγμένα προϊόντα, οι εταιρίες δέχονται συνεχώς πιέσεις να παράγουν νέα και πιο ανταγωνιστικά προϊόντα. Συνέπεια αυτού είναι τα στελέχη των εταιριών να προσπαθούν να προβλέψουν πριν την κυκλοφορία ενός νέου προϊόντος, την επιτυχία του στις αγορές και αν τους συμφέρει να το κυκλοφορήσουν σε αυτή τη μορφή. Με αυτά τα δεδομένα έχει αναδειχθεί ένας νέος κλάδος τα τελευταία χρόνια, αυτός του βέλτιστου σχεδιασμού γραμμής προϊόντων και βοηθάει τα στελέχη να δώσουν λύση στα προβλήματα σχεδιασμού νέων προϊόντων.

Στα προβλήματα αυτά υπάρχουν πολλοί περιορισμοί οι οποίοι μπορεί να είναι οικονομικοί, νομικοί, έλλειψη ή περιορισμός στην χρήση πρώτων υλών και άλλα. Εξαιτίας αυτού δημιουργούνται πολύπλοκα προβλήματα με εκατομμύρια πιθανούς συνδυασμούς και λύσεις, κάτι το οποίο δεν μπορεί να επεξεργαστεί και να βρει την λύση ο ανθρώπινος νους. Οι αλγόριθμοι που έχουν αναπτυχθεί για αυτό το σκοπό έρχονται να δώσουν λύση σε τέτοια προβλήματα αξιοποιώντας τα σύγχρονα πληροφοριακά συστήματα και τις εικονικά απεριόριστες δυνατότητες που έχουν.

Ο βέλτιστός σχεδιασμός γραμμής προϊόντων βασίζεται στις προτιμήσεις των καταναλωτών. Πρέπει λοιπόν να μετρηθούν με κάποιο τρόπο οι προτιμήσεις των καταναλωτών. Ο πιο συνηθισμένος τρόπος μέτρησης των προτιμήσεων αυτών είναι η έρευνα αγοράς με ερωτηματολόγια. Οι καταναλωτές καλούνται να απαντήσουν σε μια σειρά ερωτήσεων οι οποίες θα βοηθήσουν αυτόν που διεξάγει την έρευνα να βγάλει αποτελέσματα τα οποία θα οδηγήσουν στο σχεδιασμό του βέλτιστου προϊόντος με βάση τις ανάγκες των καταναλωτών. Οι απαντήσεις συνήθως δίνονται σε μορφή κατάταξης (ranking) προτίμησης ανάμεσα σε διαφορετικά προϊόντα ή και με απευθείας σύγκριση δύο προϊόντων.

Με αυτή την μέθοδο ο καταναλωτής δίνει μεγαλύτερη η μικρότερη αξία σε κάθε προϊόν ανάλογα με τις προτιμήσεις του. Έπειτα εφαρμόζεται η μέθοδος **conjoint analysis** η οποία αντιστοιχίζει σε κάθε επίπεδο κάθε χαρακτηριστικού τη μερική αξία (part worth) για κάθε

καταναλωτή. Στη συνέχεια αθροίζονται οι μερικές αξίες κάθε επιπέδου κάθε χαρακτηριστικού και υπολογίζεται η συνολική αξία (utility) κάθε προϊόντος για κάθε καταναλωτή. Ο γραμμικός συνδυασμός των part worths αναφέρεται στα χαρακτηριστικά των προϊόντων που δείχνουν την συνολική αξία (utility) που εκτιμά να λάβει ο καταναλωτής από το συγκεκριμένο προϊόν. Πρέπει επίσης να σημειωθεί ότι οι προτιμήσεις των καταναλωτών εκτιμώνται σε τρία επίπεδα. Σε ατομικό επίπεδο, οπού για κάθε καταναλωτή εκτιμάται ένα μοναδικό σύνολο μερικών αξιών. Σε τμηματικό επίπεδο, στο οποίο η αγορά θεωρεί ότι χωρίζεται σε ομογενή τμήματα στα οποία οι καταναλωτές που ανήκουν στο ίδιο τμήμα έχουν παρόμοιες προτιμήσεις και τέλος σε συνολικό επίπεδο, οπού οι μερικές αξίες υπολογίζονται για το σύνολο των καταναλωτών του δείγματος.

Το πρόβλημα βέλτιστου σχεδιασμού γραμμής νεών προϊόντων κατατάσσεται στην κατηγορία των **NP – hard** προβλημάτων, καθώς κανένας αλγόριθμος δεν μπορεί να πιστοποιήσει σε πολυωνυμικό χρόνο ότι το βέλτιστο που προσδιορίζει είναι το ολικό βέλτιστο. Αυτός είναι και ο λόγος που χρησιμοποιούμε αλγόριθμους βελτιστοποίησης για να προσεγγίσουμε το ολικό βέλτιστο, διότι μια εξαντλητική έρευνα είναι πρακτικά αδύνατη καθώς απαιτεί πολύ μεγάλο χρονικό διάστημα προκειμένου να διεξαχθεί.

Στην παρούσα μελέτη θα εφαρμοστούν δυο αλγόριθμοι εμπνευσμένοι από τη φύση για την επίλυση του προβλήματος βέλτιστου σχεδιασμού γραμμής προϊόντων για τα δεδομένα που χρησιμοποιήθηκαν στα πλαίσια της δημοσίευσης «*Optimizing Product Line Designs: Efficient Methods and Comparisons», Alexandre Belloni, Robert Freund, Matthew Selove και Duncan Simester, 2008*. Ο 1ος είναι ο Cuckoo Search algorithm (Xin-she Yang and Suash Deb, 2009) και ο 2ος είναι ο bat algorithm (Xin-she Yang, 2010).

2.1.1 Conjoint analysis

Η conjoint analysis είναι μια τεχνική που χρησιμοποιείται στην έρευνα αγοράς και βοηθάει τον ερευνητή να καθορίσει πως οι καταναλωτές αξιολογούν τα διαφορετικά χαρακτηριστικά ενός προϊόντος. Οι καταναλωτές καλούνται να αξιολογήσουν μια σειρά προϊόντων και αναλύοντας τις απαντήσεις που δίνονται η conjoint analysis είναι σε θέση να ορίσει πως ο κάθε καταναλωτής αξιολογεί κάθε χαρακτηριστικό σε αυτά τα προϊόντα.

Για να μπορεί να εφαρμοστεί η conjoint analysis πρέπει να πληρούνται τρεις προϋποθέσεις.

1. Το εκάστοτε προϊόν ή υπηρεσία που αξιολογείται πρέπει να μπορεί να αναλυθεί σε ένα σύνολο χαρακτηριστικών με κάθε χαρακτηριστικό να λαμβάνει διάφορες τιμές.
2. Οι καταναλωτές διαλέγουν το προϊόν ή την υπηρεσία που μεγιστοποιεί τη χρησιμότητα για αυτούς, την αντιλαμβανόμενη δηλαδή αξία η οποία προκύπτει

από τα χαρακτηριστικά του προϊόντος και από την κατανάλωση του με σκοπό την ικανοποίηση μια ανάγκης.

3. Η conjoint analysis θεωρεί πως οι καταναλωτές αξιολογούν τη χρησιμότητα ενός προϊόντος συνδυάζοντας τις επιμέρους αξίες partworths, οι οποίες προκύπτουν από τις τιμές των επιμέρους χαρακτηριστικών του προϊόντος.

Η διαδικασία της conjoint analysis αποτελείται από τρία στάδια.

- Πρώτο στάδιο είναι ο σχεδιασμός της ερευνάς. Σε αυτό το στάδιο επιλέγονται τα χαρακτηριστικά ανάλογα με τη κατηγορία των προϊόντων, η επιλογή των επιπέδων για κάθε χαρακτηριστικό καθώς και η δημιουργία των προφίλ των προϊόντων που θα αξιολογηθούν.
- Στο δεύτερο στάδιο συλλέγονται τα δεδομένα από τους καταναλωτές και περιλαμβάνεται ο σχεδιασμός της διαδικασίας συλλογής και η επιλογή των μεθόδων επεξεργασίας των προφίλ.
- Τέλος στο τρίτο στάδιο αξιοποιούνται τα αποτελέσματα που προκύπτουν από τα προηγούμενα στάδια. Η αξιοποίηση αυτή περιλαμβάνει την τμηματοποίηση των καταναλωτών με βάση την αξία που έχουν δώσει στα χαρακτηριστικά, την προσομοίωση της αγοράς και τη βελτιστοποίηση του προϊόντος.

2.2 Τα τρία κριτήρια βελτιστοποίησης

Το πρώτο κριτήριο βελτιστοποίησης εισήχθη από τους Shocker και Srinivasan το 1974. Αυτό είναι το κριτήριο της μεγιστοποίησης του μεριδίου αγοράς. Σύμφωνα με το κριτήριο αυτό στόχος της εταιρίας είναι να αυξήσει όσο το δυνατόν περισσότερο τις πωλήσεις της έναντι των ανταγωνιστών της.

Το επόμενο κριτήριο βελτιστοποίησης είναι το κριτήριο του κέρδους του πωλητή. Αυτό το κριτήριο πρωτοεμφάνισε ο Green et al το 1981. Σε αυτό το κριτήριο στόχος της εταιρίας είναι η βελτιστοποίηση του κέρδους της, αλλά είναι πιο περίπλοκο από τα άλλα κριτήρια διότι πρέπει στην αντικειμενική συνάρτηση να προστεθεί το οριακό κόστος από κάθε επίπεδο του χαρακτηριστικού όπως το έχει υπολογίσει η εταιρία.

Το τελευταίο κριτήριο βελτιστοποίησης είναι το κέρδος του αγοραστή το οποίο εισήχθη από τους Green και Krieger το 1988. Το κέρδος του αγοραστή είναι το λιγότερο χρησιμοποιούμενο κριτήριο διότι αφορά τη βελτιστοποίηση προϊόντων και υπηρεσιών τα οποία προσφέρονται από δημόσιους φορείς. Με βάση αυτό το κριτήριο λοιπόν δεν υπάρχει καθόλου ανταγωνισμός και μοναδικός στόχος είναι η μεγιστοποίηση των αξιών που προσφέρουν τα προϊόντα και οι υπηρεσίες στους καταναλωτές.

2.3 Μοντελοποίηση απόφασης

Μοντελοποίηση της απόφασης είναι η διαδικασία μέσω της οποίας προσομοιώνεται η συμπεριφορά ενός καταναλωτή όταν έχει να επιλέξει ανάμεσα σε ένα σύνολο εναλλακτικών επιλογών ενός προϊόντος. Για αυτό το σκοπό υλοποιείται ένα μοντέλο επιλογής μέσω του οποίου ο καταναλωτής συλλέγει πληροφορίες προκειμένου να διαλέξει ένα προϊόν από ένα πεπερασμένο σύνολο διαφορετικών προϊόντων. Το μοντέλο αυτό είναι ένα μαθηματικό μοντέλο το οποίο μετατρέπει τις χρησιμότητες των προϊόντων, όπως τις ορίζει ο κάθε καταναλωτής, σε πιθανότητες επιλογής του συγκεκριμένου προϊόντος. Η συμπεριφορά επιλογής ενός προϊόντος από ένα καταναλωτή είναι μια περίπλοκη διαδικασία, με πολλούς αστάθμητους παράγοντες. Αυτό είχε ως συνέπεια την ανάπτυξη μοντέλων καταμερισμού των προτιμήσεων που βοηθάνε τον ερευνητή να καταλάβει καλύτερα την διαδικασία λήψης αποφάσεων των καταναλωτών. Τα μοντέλα αυτά χωρίζονται στα μοντέλα σταθερής αξίας και στα μοντέλα τυχαίας αξίας.

2.3.1 Μοντέλα σταθερής αξίας

Τα μοντέλα σταθερής αξίας θεωρούν ότι οι αξίες των προϊόντων είναι σταθερές και παρουσιάζουν τη στοχαστική φύση της ανθρώπινης συμπεριφοράς με τη χρήση ενός επίπεδου αβεβαιότητας στον κανόνα της απόφασης. Το πιο δημοφιλές και πιο συχνά χρησιμοποιούμενο μοντέλο σταθερής αξίας είναι το BTL. Η μαθηματική του αναπαράσταση είναι:

$$p_{ij} = \frac{U_{ij}}{\sum_{j=1}^{n} U_{ij}}$$

όπου

p_{ij} η πιθανότητα ο καταναλωτής I να επιλέξει το προϊόν j

U_{ij} η συνολική αξία που δίνει ο καταναλωτής I στο προϊόν j

n ο αριθμός των ανταγωνιστικών προϊόντων

Αργότερα αναπτύχθηκαν, από τους Pessemier και Lesourne, μοντέλα που επεκτείνουν το BTL. Στο μοντέλο που πρότειναν οι αξίες των προϊόντων μετασχηματίζονται με τη χρήση ενός εκθέτη ο οποίος ελέγχει τις τιμές των πιθανοτήτων επιλογής διατηρώντας όμως την αρχική κατάταξή των προτιμήσεων. Το τροποποιημένο μοντέλο του Pessemier είναι το εξής:

$$p_{ij} = \frac{U_{ij}^{a}}{\sum_{j=1}^{n} U_{ij}^{a}}$$

όπου

p_{ij} η πιθανότητα ο καταναλωτής I να επιλέξει το προϊόν j

U_{ij} η συνολική αξία που δίνει ο καταναλωτής I στο προϊόν j

n ο αριθμός των ανταγωνιστικών προϊόντων

a Εκθέτης μετασχηματισμού

Ο κανόνας **πρώτης επιλογής ή μέγιστης χρησιμότητας** (First Choice / Maximum Utility rule) είναι ένα ντετερμινιστικό μοντέλο που υποστηρίζει ότι ο καταναλωτής θα επιλέγει πάντα το προϊόν με τη μεγαλύτερη αξία/χρησιμότητα. Στην επιλογή με τη μεγαλύτερη αξία προσδίδεται πιθανότητα επιλογής 1 ενώ στις υπόλοιπές εναλλακτικές δίνεται τιμή 0. Το μειονέκτημά αυτού του κανόνα είναι ότι λαμβάνει υπόψιν μόνο το προϊόν με τη μεγαλύτερη χρησιμότητα και αγνοεί τα υπόλοιπα.

2.3.2 Μοντέλα τυχαίας αξίας

Τα μοντέλα τυχαίας αξίας έχουν διαφορετική προσέγγιση από αυτά της σταθερής αξίας. Σε αυτή τη περίπτωση θεωρείται ότι ο καταναλωτής επιλέγει πάντα την επιλογή με τη μεγαλύτερη αξία. Η αξία αυτή αποτελείται από το στοχαστικό μέρος και το ντετερμινιστικό μέρος. Το πιο δημοφιλές μοντέλο τυχαίας αξίας είναι το μοντέλο MNL. Η μαθηματική αναπαράσταση του είναι:

$$p_{ij} = \frac{e_{ij}^{u}}{\sum_{j=1}^{n} e_{ij}^{u}}$$

όπου

p_{ij} η πιθανότητα ο καταναλωτής I να επιλέξει το προϊόν j

n ο αριθμός των ανταγωνιστικών προϊόντων

2.4 Ορισμός του προβλήματος

Το συγκεκριμένο πρόβλημα απευθύνεται σε 1η φάση στο βέλτιστο σχεδιασμό μιας γραμμής 5 σακιδίων, έναντι μιας σταθερής γραμμής 3 ανταγωνιστικών προϊόντων. Παίρνουμε τις προτιμήσεις 324 καταναλωτών και με βάση αυτές πρέπει να βελτιστοποιήσουμε το κέρδος της επιχείρησής. Τα σακίδια χαρακτηρίζονται από την τιμή που λαμβάνει 7 επίπεδα (70$ 75$ 80$ 85$ 90$ 95$ 100$) και άλλα 9 χαρακτηριστικά (χερούλι, εναλλακτικό χρώμα, θήκη κινητού και άλλα) τα οποία παίρνουν δυο τιμές (ναι/όχι).

Χαρακτηριστικό	Μέση μερική αξία	Αυξανόμενο οριακό κόστος
Αύξηση τιμής 5$	-7.6	-5.00
Μεγάλο μέγεθος	17.9	3.50
Εναλλακτικό χρώμα	-36.0	0.00
Λογότυπο	9.0	2.00
Χερούλι	37.7	3.50
Θήκη gadget	5.2	3.00
Θήκη κινητού	5.5	3.00
Θήκη πλέγματος	9.7	2.00
Ενισχυμένη θήκη	18.2	3.50
Ενισχυμένος πάτος	24.4	4.50

Με βάση των αριθμό των χαρακτηριστικών και των επιπέδων ο συνολικός αριθμός προϊόντων είναι 3584 και μπορούν να δημιουργηθούν είναι $4,9 \times 10^{15}$ διαφορετικές γραμμές 5 σακιδίων. Ο αριθμός των διαφορετικών γραμμών που μπορούν να παραχθούν είναι πολύ μεγάλος για αυτό η επίλυση του προβλήματος γίνεται δύσκολη.

Σε 2η φάση γίνεται βέλτιστος σχεδιασμός για το ίδιο πρόβλημα με τα ίδια δεδομένα αλλά αυτή τη φορά για μια γραμμή 10 σακιδίων αντί για 5 που έγινε στην αρχή. Αυτό μας επιτρέπει να έχουμε μια καλύτερη εικόνα για το πως δουλεύουν αυτοί οι 2 αλγόριθμοι και ποιος από τους 2 παράγει καλύτερα αποτελέσματα. Σε αυτή τη περίπτωση ο συνολικός αριθμός των διαφορετικών γραμμών παραγωγής που δημιουργούνται είναι $9,5 \times 10^{28}$.

2.5 Αλγόριθμοι που έχουν χρησιμοποιηθεί στο παρελθόν για το παρόν πρόβλημα

Στη συνέχεια θα δούμε τους αλγόριθμους που έχουν χρησιμοποιηθεί κατά το παρελθόν για να την επίλυση του ίδιου προβλήματος. Στη παρούσα μελέτη είναι η πρώτη φορά που χρησιμοποιούνται οι αλγόριθμοί bat algorithm και cuckoo search για αυτό το πρόβλημα. Σε αυτό το σημείο πρέπει να σημειωθεί ότι δε παράγουν όλοι οι αλγόριθμοι εξίσου καλά αποτελέσματα για αυτό το πρόβλημα. Κάποιοι προσεγγίζουν τη λύση καλύτερα από άλλους εξαιτίας του τρόπου που δουλεύουν.

2.5.1 Genetic algorithm

Ο τρόπος λειτουργίας των Γενετικών αλγορίθμων είναι εμπνευσμένος από τη φύση. Η βασική ιδέα πίσω από τη λειτουργία αυτών των αλγορίθμων είναι η μίμηση της των μηχανισμών της βιολογικής εξέλιξης. Χρησιμοποιούνται για την επίλυση μιας μεγάλης γκάμας προβλημάτων και πρωτοεμφανίστηκαν στις αρχές του 1950 όταν επιστήμονες προσπάθησαν να προσομοιώσουν πολύπλοκα βιολογικά συστήματα με τη χρήση υπολογιστών.

Οι Γενετικοί αλγόριθμοι ακολουθούν τα εξής βήματα. Αρχικά ξεκινάνε με τη δημιουργία μιας τυχαίας λύσης αρχικού πληθυσμού. Οι καλύτερες λύσεις αυτού του πληθυσμού επιβιώνουν και αναπαράγονται προκειμένου να δημιουργήσουν καινούργιες λύσεις. Αν οι νέες λύσεις που θα δημιουργηθούν είναι καλύτερες από τις προηγούμενες τότε τις αντικαθιστούν και επαναλαμβάνουν την ίδια διαδικασία. Ο Αλγόριθμος σταματάει και θεωρεί ότι έχει λυθεί το πρόβλημα όταν ικανοποιηθεί το κριτήριο τερματισμού.

2.5.2 Simulated annealing

Ο αλγόριθμος προσομοιωμένης ανόπτησης προτάθηκε πρώτη φορά από τον Kirkpatrick το 1982. Η Ανόπτηση είναι όταν ένα υλικό θερμαίνεται μέχρι το σημείο τήξης του και στη συνέχεια ψύχεται αργά. Όταν η ψύξη ολοκληρωθεί τότε το υλικό πέφτει σε στάδιο χαμηλότερης ενέργειας. Αν όμως το υλικό ψυχθεί απότομα ή αν η αρχική θερμοκρασία δεν είναι αρκετά υψηλή τότε τα μόρια θα ακινητοποιηθούν στις θέσεις τους και το σύστημα μπορεί να παρουσιάσει κενά και ατέλειες. Ο αλγόριθμος της προσομοιωμένης ανόπτησης προσομοιώνει τις αλλαγές στην ενέργεια ενός συστήματος που υπόκειται σε μία διαδικασία ψύξης μέχρι να συγκλίνει σε ένα σημείο ισορροπίας.

Το κύριο χαρακτηριστικό του αλγορίθμου είναι ότι δέχεται και χειρότερες λύσεις με σκοπό να απελευθερωθεί η λύση από πιθανά τοπικά βέλτιστα. Η αποδοχή των χειρότερων λύσεων είναι ανάλογη της θερμοκρασίας. Στην αρχή αποδέχεται ευκολότερα χειρότερες λύσεις λόγω της υψηλής θερμοκρασίας αλλά όσο ψύχεται το υλικό και πλησιάζει το ολικό βέλτιστο γίνεται δυσκολότερο να πάρει χειρότερες λύσεις.

2.5.3 Particle swarm optimization (PSO)

Ο αλγόριθμος αυτός εισήχθη από τους J. Kennedy και R. Eberhart το 1995. Η νοημοσύνη σμήνους είναι μια ιδιότητα συστημάτων που επιδεικνύουν συλλογικά ευφυή συμπεριφορά. Αποτελεί ένα σύστημα που αντιπροσωπεύει μια οντότητα που ανιχνεύει το περιβάλλον προκειμένου να εκτελέσει μια ενέργεια που έχει επιλέξει. Η κάθε οντότητα λειτουργεί αυτόνομα και συνδυάζοντας όλες τις οντότητές προκύπτει μια συνολική συλλογική συμπεριφορά. Δεν υπάρχει κάποιος που δίνει οδηγίες, παίρνουν αποφάσεις και μέσα από απλούς κανόνες εμφανίζονται φαινόμενα (σμήνη πουλιών, κοπάδια ψαριών, τα μυρμήγκια υπολογίζουν βέλτιστες διαδρομές προς την τροφή τους, και άλλα) τα οποία οδηγούνται μέσα από τη συλλογική συμπεριφορά και ευφυΐα.

Στο παρόν πρόβλημα εφαρμόστηκε πρώτη φορά από τους Τσαφαράκης Στέλιος, Μαρινάκης Ιωάννης και Ματσατσίνης Νικόλαος στην εργασία «Particle Swarm Optimization for Optimal Product Line Design» το 2010. Για την λειτουργία του ο αλγόριθμος χρησιμοποιεί την κίνηση των ατόμων ενός πληθυσμού από σμήνος και χρησιμοποιεί ευέλικτους μηχανισμούς για να προσαρμόζεται στις τοπικές και ολικές ικανότητες εξερεύνησης των ατόμων του σμήνους.

2.5.4 Greedy heuristic

Η μέθοδος εμφανίστηκε πρώτη φορά στο πρόβλημα της βέλτιστης γραμμής παραγωγής από τους Green και Krieger το 1985. Αρχικά ο αλγόριθμος σχεδιάζει μια γραμμή προϊόντων από ένα και μοναδικό καλό προϊόν το οποίο μεγιστοποιεί τα κέρδη. Στη συνέχεια προσθέτει κάθε φορά το προϊόν που μεγιστοποιεί την αντικειμενική συνάρτηση της ημιτελούς λύσης.

Ο αλγόριθμός αυτός ολοκληρώνεται όταν ικανοποιηθεί το κριτήριο που έχει ορίσει ο χρήστης στην αρχή του προβλήματος. Το τελικό αποτέλεσμα σε αυτή τη μέθοδο εξαρτάται σε μεγάλο βαθμό από αρχικό προϊόν το οποίο έχει οριστεί. Συνεπώς αν η αρχική επιλογή δεν είναι καλή δε θα βγάλει καλά αποτελέσματα. Αντιθέτως μια σωστή επιλογή αρχικού προϊόντος θα φέρει πολύ καλές λύσεις.

2.5.5 Product - swapping heuristic

Το Product Swapping Heuristic αναφέρεται σε μια μέθοδο των Green και Krieger που αναπτύχθηκε το 1985. Η συγκεκριμένη μέθοδος επιλέγει αρχικά μια τυχαία γραμμή παραγωγής και αξιολογεί τα κέρδη που παράγει αυτή η λύση. Έπειτα ελέγχεται κάθε προϊόν το οποίο όμως δεν βρίσκεται στη τρέχουσα λύση. Αν το προϊόν που ελέγχθηκε βελτιώνει τη λύση και αυξάνει τα κέρδη τότε αντικαθίσταται με το χειρότερο προϊόν που υπάρχει μέσα στη λύση. Η διαδικασία επαναλαμβάνεται μέχρι να σταματήσει να βελτιώνεται η λύση από την αντικατάσταση ενός προϊόντος με ένα άλλο.

2.5.6 Dynamic programming heuristic

Η μέθοδος αυτή αναπτύχθηκε από τους Kohli και Krishnamurti το 1987. Σκοπός τους ήταν να χρησιμοποιήσουν τη μέθοδο αυτή για να λύσουν το πρόβλημα του βέλτιστου σχεδιασμού μια γραμμής προϊόντων. Ο τρόπος λειτουργίας της είναι ο εξής. Αρχικά δημιουργεί μια γραμμή προϊόντος με μόνο ένα χαρακτηριστικό και το οποίο αξιολογείται. Κάθε επόμενη φορά δημιουργείται μια νέα γραμμή παραγωγής από ένα μόνο πάλι χαρακτηριστικό η οποία αξιολογείται. Αυτό επαναλαμβάνεται μέχρι να έχουν δημιουργηθεί` και αξιολογηθεί όλα τα χαρακτηριστικά.

2.5.7 Divide and conquer heuristic

Αυτή αποτελεί μια ακόμα μέθοδο των Green και Krieger η οποία προτάθηκε το 1988. Ο τρόπος λειτουργίας αυτής της μεθόδου είναι όπως λέει και το όνομά της να διαιρεί τη γραμμή προϊόντων σε ομάδες χαρακτηριστικών και να ερευνά όλους τους πιθανούς συνδυασμούς της μιας ομάδας κρατώντας τις υπόλοιπές ομάδες σταθερές. Η διαδικασία ολοκληρώνεται όταν ολοκληρώσει τον έλεγχο όλων των πιθανών συνδυασμών σε όλες τις ομάδες και φτάσει στο σημείο που δεν μπορεί να βελτιώσει άλλο τη λύση.

2.5.8 Beam search heuristic

Ο Αλγόριθμος Beam Search Heuristic χρησιμοποιήθηκε για πρώτη φορά από τον Nair το 1995. Οι αρχικές εφαρμογές του ήταν στην αναγνώριση φωνής και εικόνας για εφαρμογές τεχνητής νοημοσύνης. Αυτή η μέθοδος μοιάζει αρκετά με τη Dynamic Programming Heuristic με δυο σημαντικές διαφορές. Η πρώτη είναι ότι η συγκεκριμένη μέθοδος συνδυάζει διαφορετικά σύνολα χαρακτηριστικών, μελετώντας τα ταυτόχρονα. Η δεύτερη είναι ότι προσθέτουμε ένα προϊόν στη γραμμή προϊόντων κάθε φορά όπως γίνεται

και με το Greedy Heuristic. Επιπλέον προτού τρέξουμε τη συγκεκριμένη μέθοδο τυχαιοποιούμε τη σειρά των χαρακτηριστικών με σκοπό να τρέξουμε πολλές γραμμές παραγωγής και να κρατήσουμε τη καλύτερη.

2.5.9 Nested partitions heuristic

Η συγκεκριμένη μέθοδος αναπτύχθηκε και εφαρμόστηκε για προβλήματα βέλτιστης γραμμής προϊόντων από τον Shi το 2001. Με αυτή τη μέθοδο ο χώρος των λύσεων χωρίζεται σε διαφορετικές περιοχές και εκτιμάται η περιοχή που ενδέχεται να φέρει τα καλύτερα αποτελέσματα. Στη συνέχεια η πιο υποσχόμενη περιοχή χωρίζεται σε μικρότερες περιοχές για να γίνουν επιπλέον έλεγχοι για καλύτερη λύση. Σε αυτή τη μέθοδο είναι εφικτό να χρησιμοποιήσει και άλλες ευχετικές για να υπολογιστεί ο δείκτης της.

2.5.10 Coordinate ascent

Ο αλγόριθμος Coordinate Ascent προτάθηκε από τον Green το 1989. Στην αρχή επιλέγει και αξιολογεί τυχαία μια γραμμή προϊόντων. Έπειτα επιλέγει ένα τυχαίο χαρακτηριστικό και ελέγχει πως η αλλαγή του επιπέδου της τιμής του επηρεάζει τη λύση. Αν η λύση βελτιώνεται τότε δέχεται την αλλαγή αλλιώς την απορρίπτει. Κριτήριο τερματισμού του αλγορίθμου είναι η μη περαιτέρω βελτίωση της λύσης. Επιπλέον έχουν δημιουργηθεί παραλλαγές του συγκεκριμένου αλγορίθμου οι οποίες ελέγχουν ταυτόχρονα την αλλαγή περισσότερών από ένα χαρακτηριστικών του προϊόντος.

2.5.11 Tabu search

Η μέθοδος Tabu Search προτάθηκε από τον Fred Glover το 1986. Κύριο χαρακτηριστικό αυτού του αλγορίθμου είναι ότι δέχεται και χειρότερες λύσεις προκειμένου να ‘ξεφύγει’ η λύση από ένα τοπικό βέλτιστο. Συγκεκριμένα εισάγεται μία λίστα με τις N τελευταίες λύσεις που εξετάστηκαν οι οποίες είναι οι απαγορευμένες λύσεις. Η λίστα αυτή ανανεώνεται δυναμικά κατά τη διάρκεια του αλγορίθμου χρησιμοποιώντας τη λογική First In – First Out βοηθώντας έτσι την έρευνα να απεγκλωβιστεί από τοπικά βέλτιστα και να ψάξει σε χώρους που δεν έχουν ερευνηθεί.

2.5.12 Αλγόριθμοι τεχνητού ανοσοποιητικού συστήματος

Οι αλγόριθμοι του τεχνητού ανοσοποιητικού συστήματος είναι εμπνευσμένοι από τη λειτουργία του φυσικού ανοσοποιητικού συστήματος. Από το τρόπο λειτουργίας του ανοσοποιητικού συστήματος έχουν κατασκευαστεί υπολογιστικά μοντέλα τα οποία λύνουν πραγματικά προβλήματα. Το φυσικό ανοσοποιητικό σύστημα έχει πάρα πολλές και περίπλοκες λειτουργίες και δυνατότητες πάνω στις οποίες βασίστηκαν αυτοί οι αλγόριθμοι και αυτός είναι ο λόγος που βγάζουν τόσο καλά αποτελέσματα.

3. Περιγραφή αλγορίθμων που θα χρησιμοποιηθούν

3.1 Αλγόριθμος Cuckoo search

Ο αλγόριθμος Cuckoo search βασίζεται στην αναπαραγωγή των κούκων. Οι εμπνευστές του αλγορίθμου, Xin-she Yang and Suash Deb(2009), παρατήρησαν την διαδικασία αναπαραγωγής των κούκων και εμπνευστήκαν τον συγκεκριμένο αλγόριθμο.

Η διαδικασία που ακολουθούν οι κούκοι είναι πολύ απλή αλλά ταυτόχρονα και πολύ έξυπνη. Αντί να δημιουργήσουν δικές τους φωλιές προκειμένου να γεννήσουν και να κλωσήσουν τα αυγά τους, κάτι το οποίο θα απαιτούσε πολύ χρόνο και προσπάθεια, πάνε και αφήνουν τα αυγά τους σε φωλιές άλλων πουλιών και αφήνουν αυτά να μεγαλώσουν τους απογόνους τους. Φυσικά κανένα είδος πουλιού δε δέχεται να φροντίζει και να μεγαλώνει αυγά από άλλο είδος, για αυτό οι κούκοι έχουν αναπτύξει διάφορες μεθόδους ώστε να μην καταλαβαίνουν τα υπόλοιπα πουλιά ότι μεγαλώνουν αυγό κούκων. Κάποιοι κούκοι μιμούνται τον χρώμα και το σχήμα των αυγών τους από τα πουλιά στα οποία πρόκειται να αφήσουν το αυγό ενώ άλλοι κούκοι μιμούνται τις φωνές των πουλιών ώστε να μην καταλάβουν ότι δεν είναι δικό τους και το σκοτώσουν.

Βεβαίως υπάρχουν και φορές που τα πουλιά ανακαλύπτουν ότι υπάρχει αυγό κούκου στη φώλια τους, και τα αντιμετωπίζουν είτε πετώντας τα έξω είτε εγκαταλείπουν εντελώς τη φώλια και φεύγουν για να φτιάξουν καινούργια.

3.1.1 Περιγραφή Cuckoo search algorithm

Όπως αναφέρθηκε προηγουμένως οι κούκοι αναπαράγονται γεννώντας τα αυγά τους σε φωλιές άλλων πουλιών. Ο αλγόριθμός Cuckoo search στηρίζεται στους τρεις ακόλουθους κανόνες.

- Κάθε κούκος διαλέγει τυχαία μια φωλιά και αφήνει ένα αυγό μέσα της.
- Οι καλύτερες φωλιές με τα υψηλής ποιότητας αυγά θα κρατηθούν για την επόμενη γενιά.
- Για ένα προκαθορισμένο αριθμό φωλιών, το αυγό του κούκου θα ανακαλυφθεί με πιθανότητα p_a με το p_a να ανήκει στο διάστημα [0,1].

Για τον τελευταίο κανόνα έχει επικρατήσει να αντικαθίσταται ένα ποσοστό p_a των χειρότερων φωλιών (λύσεων) με νέες τυχαίες λύσεις. Η ποιότητα κάθε φωλιάς υπολογίζεται με βάση το πόσο καλή είναι η λύση σε σχέση με το πρόβλημα που έχουμε να λύσουμε. Από εκτελεστική άποψη του αλγορίθμου θεωρούμε ότι κάθε φωλιά έχει ένα αυγό το οποίο είναι και η λύση και ότι κάθε κούκος μπορεί να γεννήσει μόνο ένα αυγό.

Όταν δημιουργούμε νέες λύσεις $x^{(t+1)}$ για το κούκο i εφαρμόζεται μια Levy flight

$$x_i^{t+1} = x_i^t + aLevy(\lambda)$$

όπου $\alpha > 0$ είναι το μέγεθος του βήματος το οποίο συσχετίζεται με την κλίμακα του προβλήματος το οποίο λύνουμε.

Καθοριστικός παράγοντας για τα αποτελέσματα που θα παράγει ο αλγόριθμος είναι και οι τιμές των παραμέτρων που θα επιλεχτούν. Αυτές τις τιμές επιλέγονται από τον χρήστη στην αρχή της υλοποίησης του κώδικα και η επιλογή τους εξαρτάται από το πρόβλημα που επιθυμούμε να λυθεί κάθε φορά. Στον αλγόριθμο cuckoo search οι παράμετροί είναι το **Pa** $\in [0,1]$ το οποία καθορίζει το ποσοστό των φωλιών που θα ανακαλυφθούν και οι παράμετροι **n** και **β** οι οποίες καθορίζουν το μέγεθός του βήματος της εξίσωσής levy flight η οποία παράγει νέες λύσεις.

Έπειτα από πολλές δοκιμές και σύγκριση της απόδοσης του cuckoo search algorithm για διαφορετικές τιμές παραμέτρων επιλέχτηκαν οι εξής τιμές που δίνουν τα βέλτιστα αποτελέσματα στο πρόβλημά μας.

$$Pa = 0.3 \, , \, n = 1 \, , \, \beta = 1$$

Ο ψευδοκώδικας του αλγορίθμου είναι ο εξής:

Αντικειμενική συνάρτηση: $f(\chi)$, $\chi = (\chi_1, \chi_2, \ldots, \chi_d)$

Δημιουργία ενός αρχικού πληθυσμού από n φωλιές

While ο μέγιστος αριθμός των επαναλήψεων δεν έχει ολοκληρωθεί

Διάλεξε ένα τυχαίο κούκο i και δημιούργησε νέα λύση με πτήση Levy

Υπολόγισε τη νέα αντικειμενική συνάρτηση F

Διάλεξε μια τυχαία φωλιά j ;

if $F_i < F_j$,

Αντικατέστησε το j με την καινούργια λύση;

end if

Εγκατέλειψε ένα ποσοστό (P_a) *από τις χειρότερες φωλιές*

Αντικατέστησε με νέες φωλιές χρησιμοποιώντας πτήση Levy

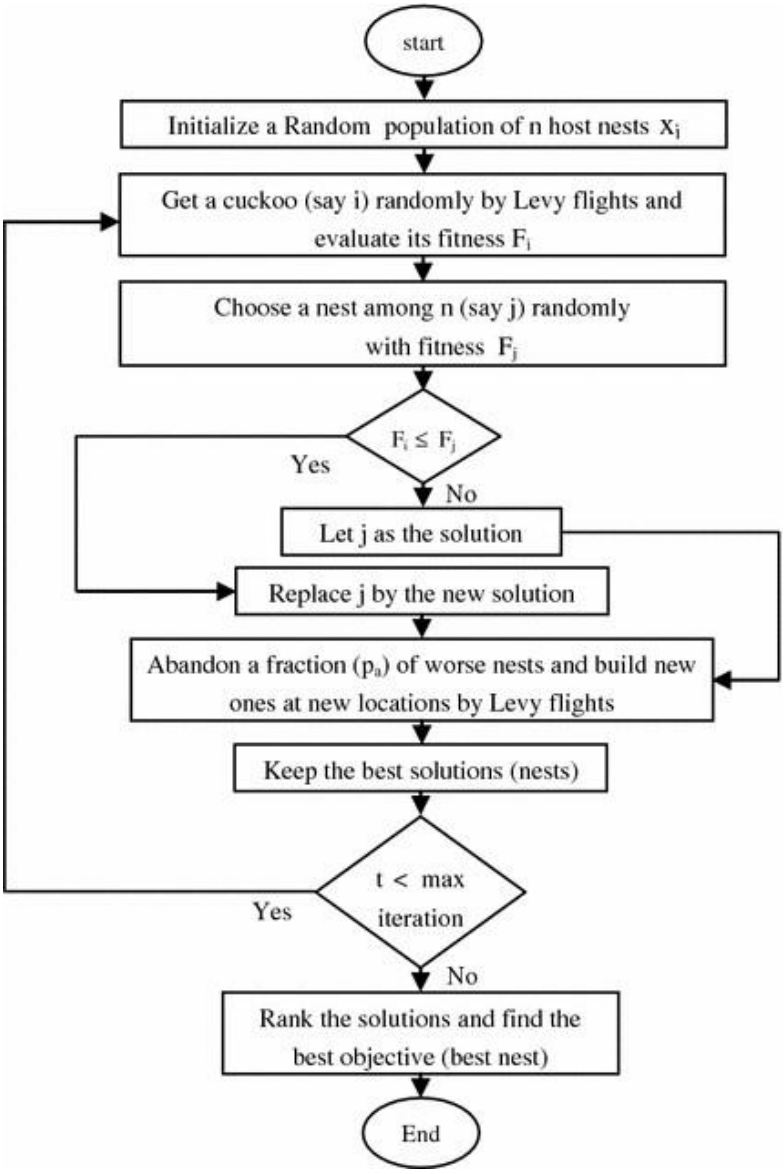
Κράτα τις καλύτερες λύσεις

Βρες την τρέχον βέλτιστη

end while

Επιστροφή της καλύτερης λύσης

Παρακάτω φαίνεται το διάγραμμά ροής του Cuckoo search algorithm.



Στη συνέχεια παρουσιάζονται τα βήματα που κάνει ο αλγόριθμος που υλοποιήθηκε στο MATLAB προκειμένου να επιλύσει το πρόβλημα βέλτιστου σχεδιασμού προϊόντων μιας γραμμής παραγωγής.

1. Αρχικά φορτώνονται όλα τα δεδομένα του προβλήματος στο MATLAB από τη δημοσίευση του Belloni et al. (2008).
2. Το 2^{ο} βήμα είναι να δημιουργηθούν τυχαίες αρχικές λύσεις, να υπολογιστεί το κέρδος της κάθε μιας από αυτές τις λύσεις και να ταξινομηθούν ανάλογα με τα κέρδη τους.
3. Επιλέγεται τυχαία μια από τις λύσεις και με χρήση της Levy flight παράγεται μια καινούργια λύση, της οποίας υπολογίζεται το κέρδος.
4. Γίνεται σύγκριση του κέρδους της νέας αυτής λύσης με μια από τις ήδη υπάρχουσες λύσεις. Αν η καινούργια λύση βελτιώνει το κέρδος τότε αντικαθιστά την παλιά λύση. Σε διαφορετική περίπτωση διαγράφεται η νέα λύση.
5. Διαγράφεται ένα ποσοστό (το οποίο έχει οριστεί στην αρχή του κώδικα) των χειρότερων λύσεων και αντικαθίσταται με νέες λύσεις οι οποίες δημιουργούνται πάλι με τη χρήση Levy flight.
6. Υπολογίζεται το κέρδος των νέων λύσεων, ταξινομούνται και αποθηκεύεται η καλύτερη λύση.
7. Επανάληψη βημάτων 3-6 μέχρι να εκπληρωθεί το κριτήριο τερματισμού (αριθμός επαναλήψεων που έχει οριστεί στην αρχή του κώδικα).
8. Εκτύπωση της καλύτερης λύσης και τερματισμός κώδικα.

3.2 Αλγόριθμος Bat

Ο αλγόριθμος της νυχτερίδας (Bat algorithm) είναι ένας από τους πιο πρόσφατους μεθευρετικούς αλγορίθμους. Αναπτύχθηκε από τον Xin-She Yang το 2010 και είναι εμπνευσμένος από το σύστημα ηχοεντοπισμού που χρησιμοποιούν οι νυχτερίδες κατά την αναζήτηση της τροφής τους. Έχει αναδειχθεί σε έναν ιδιαίτερα αποδοτικό πληθυσμιακό αλγόριθμο νοημοσύνης σμήνους, με την απόδοσή του να μπορεί να αυξάνεται περαιτέρω ‘έπειτα από κατάλληλη τροποποίηση ορισμένων παραμέτρων και ένταξη στοιχείων από άλλους μεθευρετικούς αλγορίθμους.

3.2.1 Το φαινόμενο του ηχοεντοπισμού

Ο ηχοεντοπισμός των νυχτερίδων είναι ένα από τα πιο ενδιαφέρον χαρακτηριστικά τους. Αυτό το χαρακτηριστικό εντοπίζεται και σε άλλα ζώα όπως οι φάλαινες και τα δελφίνια, αλλά οι νυχτερίδες το έχουν εξελίξει σε πιο υψηλό βαθμό από κάθε άλλο ζώο καθώς η επιβίωσή τους εξαρτάται από αυτή τους την ικανότητα. Χρησιμοποιούν ένα είδος σόναρ για να κινούνται στο σκοτάδι μέσα στις σπηλιές, να εντοπίζουν τη λεία τους αλλά και να αποφεύγουν εμπόδια και πιθανούς εχθρούς τους.

Ο τρόπος λειτουργίας αυτού του σόναρ είναι ο εξής. Οι νυχτερίδες εκπέμπουν ηχητικά κύματα από το στόμα ή τη μύτη τους (ανάλογα το είδος της νυχτερίδας). Αυτά τα κύματα όταν συναντήσουν κάποιο εμπόδιο παράγουν ηχώ. Αυτή η ηχώ έπειτα επιστρέφει στις νυχτερίδες οι οποίες μπορούν να την επεξεργαστούν και να καταλάβουν αναλυτικές πληροφορίες για το εμπόδιο όπως το μέγεθος, το σχήμα του, αν κινείται και πολλές ακόμα λεπτομερείς που τους επιτρέπουν να το αξιολογήσουν. Επιπλέον οι νυχτερίδες υπολογίζουν τη χρονική καθυστέρηση μεταξύ της εκπομπής του παλμού και της λήψης της ανάκλασής του, τη χρονική διαφορά μεταξύ των ήχων που φτιάνουν στα δύο αφτιά καθώς και τη διακύμανση της έντασης των ανακλάσεων του ήχου ώστε να δημιουργήσουν μία τρισδιάστατη εικόνα του χώρου που τις περιβάλλει.

3.2.2 Ακουστικές ιδιότητες του ηχοεντοπισμού

Η συχνότητα ενός ηχητικού παλμού που εκπέμπεται από μια νυχτερίδα συνήθως βρίσκεται στη περιοχή μεταξύ 25 KHz και 100 KHz. Παρόλα αυτά υπάρχουν μερικά είδη νυχτερίδων που μπορούν να εκπέμπουν συχνότητες έως 150 KHz. Ο παλμός που εκπέμπεται ανά πάση στιγμή εξαρτάται από πολλούς παράγοντες όπως το είδος της νυχτερίδας καθώς και η στρατηγική που χρησιμοποιεί καθώς κυνηγάει ένα θήραμα.

Από μελέτες έχει παρατηρηθεί ότι τα περισσότερα είδη εκπέμπουν σύντομους παλμούς διάρκειας 8 έως 10 ms, υπάρχουν όμως και κάποια είδη των οποίων η διάρκεια των παλμών μπορεί να φτάσει μέχρι και τα 100 ms. Όταν οι νυχτερίδες ξεκινήσουν να κυνηγάνε τη λεία τους εκπέμπουν λιγότερούς ηχητικούς παλμούς και αφήνουν μεγάλα χρονικά περιθώριά ανάμεσα στους παλμούς. Σε αυτό το σημείο εκπέμπουν περίπου 15 με 20 ηχητικούς παλμούς το δευτερόλεπτο. Όσο πλησιάζουν όμως το θήραμά τους τόσο αυξάνεται ο αριθμός των παλμών που εκπέμπουν και στο αποκορύφωμά τους μπορεί να φτάσουν μέχρι και τους 200 παλμούς ανά δευτερόλεπτο.

3.2.3 Περιγραφή Bat algorithm

Στις προσομοιώσεις χρησιμοποιούμε ψηφιακές νυχτερίδες. Πρέπει συνεπώς να οριστούν οι κανόνες με βάση τους οποίους ενημερώνονται οι θέσεις x_i και οι ταχύτητες v_i σε ένα χώρο αναζήτησης d διαστάσεων.

$$f_i = f_{min} + (f_{max} - f_{min)}\,\beta$$

$$v_i^t = v_i^{t-1} + (x_i^t - x_*)\, f_i$$

$$x_i^t = x_i^{t-1} + v_i^t$$

όπου:

β ένας τυχαίος αριθμός ο οποίος λαμβάνεται από μια ομοιόμορφη κατανομή στο διάστημα [0,1].

χ_* η τρέχουσα ολική βέλτιστη λύση η οποία εντοπίζεται μετά από σύγκριση όλων των θέσεων μεταξύ όλων των νυχτερίδων.

Για την τοπική αναζήτηση αφού επιλεγεί μια λύση μεταξύ των καλύτερων λύσεων της τρέχουσας χρονικής περιόδου, δημιουργείται μια νέα λύση για κάθε νυχτερίδα με τη χρήση τυχαίας διαδρομής.

$$x_{new} = x_{old} + \varepsilon A^t$$

όπου ε ένας τυχαίος αριθμός στο διάστημα [-1,1] και A^t η μέση τιμή των εντάσεων του ήχου για όλες τις νυχτερίδες στην επανάληψη t.

Όπως αναφέρθηκε και προηγουμένως η σωστή επιλογή παραμέτρων είναι πολύ σημαντική για την απόδοση του αλγορίθμου. Στον αλγόριθμο της νυχτερίδας οι παράμετροί αυτοί είναι το $\mathbf{f_{min}}$ και $\mathbf{f_{max}}$, η αρχική μέση ένταση του ήχου **A**, η ένταση του παλμού **r** και ένας δείκτης **ε** που βοηθάει στη παραγωγή νέων λύσεων στην φάση της τοπικής αναζήτησης.

Έπειτα από πολλές δοκιμές και σύγκριση της απόδοσης του αλγόριθμου της νυχτερίδας για διαφορετικές τιμές παραμέτρων επιλέχτηκαν οι εξής τιμές που παράγουν τα βέλτιστα αποτελέσματα στο πρόβλημά μας.

$$f_{min} = 0 \text{ , } f_{max} = 1 \text{ , } A = 0.5 \text{ , } r = 0.5 \text{ , } \varepsilon = 15$$

Ο ψευδοκώδικας του αλγορίθμου είναι ο εξής:

Αντικειμενική συνάρτηση f(x), $\chi = (\chi_1, \chi_2, \ldots, \chi_d)$

Δημιουργία ενός πληθυσμού από νυχτερίδες x_i $(i = 1,2,\ldots,n)$ και v_i

Αρχικοποίηση της συχνότητας f_i , της έντασης του παλμού (r_i) και της έντασης του ήχου (A_i)

While ο μέγιστος αριθμός των επαναλήψεων δεν έχει ολοκληρωθεί

Δημιούργησε νέες λύσεις με τυχαία βήματα και τοπική αναζήτηση

Υπολογισμός της αντικειμενικής συνάρτησης για κάθε υποψήφια νυχτερίδα

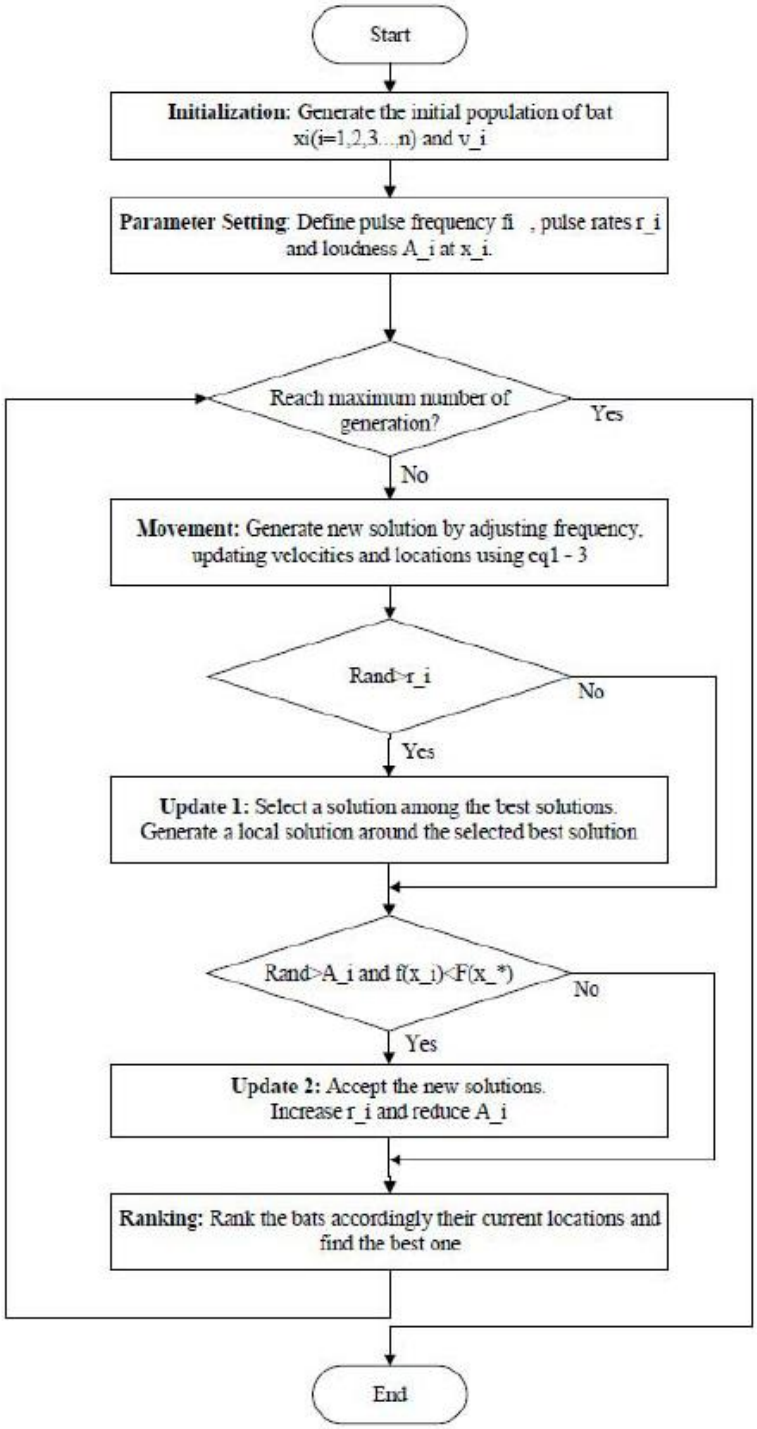
Ενημέρωση παραμέτρων ηχοεντοπισμού

Ταξινόμηση των νυχτερίδων και εύρεση της καλύτερης

end while

επιστροφή της καλύτερης λύσης

Στη συνέχεια φαίνεται το διάγραμμά ροής του Bat algorithm.



Τα βήματα του αλγορίθμου που υλοποιήθηκε για τον Bat algorithm στο MATLAB είναι τα εξής:

1. Αρχικά φορτώνονται όλα τα δεδομένα του προβλήματος στο MATLAB από τη δημοσίευση του Belloni et al. (2008).
2. Το 2^{ο} βήμα είναι να δημιουργηθούν τυχαίες αρχικές λύσεις, να υπολογιστεί το κέρδος της κάθε μιας από αυτές τις λύσεις και να ταξινομηθούν ανάλογα με τα κέρδη τους.
3. Για κάθε μια από τις αρχικές λύσεις υπολογίζεται μια νέα λύση χρησιμοποιώντας τους τύπους για τη ταχύτητα και τη θέση κάθε νυχτερίδας και υπολογίζεται το κέρδος της λύσης.
4. Αν το κέρδος της νέας λύσης είναι μεγαλύτερο από το κέρδος της ήδη υπάρχουσας λύσης και η νέα λύση δεν είναι πολύ θορυβώδης τότε αντικαθίσταται η παλιά λύση με τη νέα.
5. Υπολογίζεται το κέρδος των νέων λύσεων, ταξινομούνται και αποθηκεύεται η καλύτερη λύση.
6. Επανάληψη βημάτων 3-5 μέχρι να εκπληρωθεί το κριτήριο τερματισμού (αριθμός επαναλήψεων που έχει οριστεί στην αρχή του κώδικα).
7. Εκτύπωση της καλύτερης λύσης και τερματισμός κώδικα.

4. Περιγραφή του μοντέλου

Στόχος αυτής της εργασίας είναι η βελτιστοποίηση των κερδών από μια γραμμή πέντε και δέκα σακιδίων που πρόκειται να ενταχθούν στην αγορά. Για το σκοπό αυτό έγινε έρευνα αγοράς με τη μορφή ερωτηματολογίων, της οποίας τα αποτελέσματα θα χρησιμοποιηθούν για την επίλυση του προβλήματος. Οι 324 καταναλωτές κλήθηκαν να απαντήσουν ποια χαρακτηριστικά προτιμάνε σε μία τσάντα από τις επιλογές που τους δοθήκαν.

Κάθε σακίδιο χαρακτηρίζεται από δέκα χαρακτηριστικά. Το 1ο είναι η τιμή του προϊόντος που μπορεί να πάρει 7 τιμές (70$, 75$, 80$, 85$, 90$, 95$, 100$) και τα υπόλοιπα 9 παίρνουν τιμές ναι ή όχι (υπάρχει το χαρακτηριστικό ή δεν υπάρχει αντίστοιχα).

Στο πρόβλημα με τις 5 τσάντες οι πιθανοί συνδυασμοί διαφορετικών γραμμών παραγωγής που μπορούν να δημιουργηθούν είναι $4,9 \times 10^{15}$ ενώ για 10 τσάντες οι πιθανοί συνδυασμοί είναι $9,5 \times 10^{28}$. Όπως βλέπουμε οι πιθανοί συνδυασμοί που δημιουργούνται είναι πάρα πολλοί για να υπολογιστούν με το ανθρώπινο μυαλό αλλά και με συμβατικές υπολογιστικές μεθόδους. Σε αυτές τις περιπτώσεις οι γενετικοί και εξελικτικοί αλγόριθμοι δουλεύουν εξαιρετικά καλά και προσεγγίζουν τις λύσεις σε πολύ μικρό χρονικό διάστημα.

Για κάθε έναν από τους 324 καταναλωτές μπορούμε να υπολογίσουμε τη συνολική χρησιμότητα για κάθε μία από τις 3584 διαφορετικές τσάντες. Υπολογίζουμε λοιπόν τη χρησιμότητα για κάθε καταναλωτή για το προτεινόμενο προϊόν από τη δική μας γραμμή και τη χρησιμότητα των ανταγωνιστικών τσαντών. Συγκρίνουμε τις χρησιμότητες και καταλήγουμε στην επιλογή του καταναλωτή. Χρησιμοποιούμε τον κανόνα **πρώτης επιλογής ή μέγιστης χρησιμότητας** (First Choice / Maximum Utility rule). Το κέρδος για την πώληση μιας τσάντας ισούται με την τιμή πώλησής της μείον ένα σταθερό κόστος κατασκευής 35 δολάρια και μείον το κόστος των χαρακτηριστικών που διαθέτει. Για να υπολογίσουμε το τελικό κέρδος (που είναι η λύση του προβλήματος) αθροίζουμε το κέρδος από τις πωλήσεις κάθε σακιδίου της προτεινόμενης γραμμής.

Όπως προαναφέρθηκε η εργασία αυτή στηρίχθηκε στη δημοσίευση «*Optimizing Product Line Designs: Efficient Methods and Comparisons», των Alexandre Belloni, Robert Freund, Matthew Selove και Duncan Simester, 2008*. Χρησιμοποιήθηκαν λοιπόν οι μέθοδοι και οι πίνακες που αναπτυχθήκαν στα πλαίσια της δημοσίευσής, για τον υπολογισμό της αντικειμενικής συνάρτησης στο πρόβλημα μας, όπως φαίνονται παρακάτω.

- Η μέθοδος *« load_data»* η οποία παρέχει τα στοιχεία στους 4 επόμενους πίνακες:
    - Στον πίνακα *worths(324 x 10)* αποθηκεύονται οι μερικές αξίες κάθε χαρακτηριστικού, για κάθε προϊόν για κάθε έναν από τους καταναλωτές.
    - Στον πίνακα *bags(3584 x 10)* όπου αποθηκεύονται όλα τα διαφορετικά προϊόντα σε μορφή διανύσματος 10 χαρακτηριστικών.
    - Στον πίνακα *profits(1 x 10)* αποθηκεύεται το οριακό κόστος από την προσθήκη του κάθε χαρακτηριστικού.
    - Στον πίνακα *bagfeats(5 x 10)* αποθηκεύονται τα χαρακτηριστικά του προϊόντος της γραμμής παραγωγής.
- Ο πίνακας *bagprofits(3584 x 1)* όπου αποθηκεύονται και υπολογίζονται τα οριακά κέρδη κάθε τσάντας.
- Ο πίνακας *coop_bags(1 x 3)* με αποθηκευμένα τα 3 προϊόντα από την ανταγωνιστική γραμμή παραγωγής.
- Ο πίνακας *bagworths(3584 x 324)* όπου αποθηκεύεται η χρησιμότητα κάθε προϊόντος για κάθε έναν από τους καταναλωτές.
- Ο πίνακας *base_util(1 x 324)* αποθηκεύει την χρησιμότητα του προτιμητέου προϊόντος για κάθε καταναλωτή σε σχέση με την ανταγωνιστική γραμμή.
- Ο πίνακας *util(1 x 324)* αποθηκεύει την χρησιμότητα του προτιμητέου προϊόντος για κάθε καταναλωτή από τη δική μας γραμμή.

5. Αποτελέσματα

5.1 Bat Algorithm

Κατά τη διάρκεια της δημιουργίας του Bat Algorithm στα πλαίσια της παρούσας διπλωματικής υλοποιήθηκαν δύο μορφές του αλγορίθμου με διαφορετικούς τρόπους επεξεργασίας και αποθήκευσης των δεδομένων. Ο λόγος που έγινε αυτό είναι για να συγκριθούν μεταξύ τους και να δούμε ποια παράγει καλύτερα αποτελέσματα για το συγκεκριμένο πρόβλημα βελτιστοποίησης.

Η μια μορφή είναι να δουλεύει ο αλγόριθμός με διακριτές τιμές. Στην περίπτωση αυτή οι λύσεις αποθηκεύονται σε δισδιάστατους πίνακες και η κάθε λύση αντιπροσωπεύεται με έναν αριθμό τσάντας από το 1 έως το 3584, όσες είναι δηλαδή και οι διαφορετικές τσάντες που μπορούν να δημιουργηθούν.

Η δεύτερη μορφή του αλγορίθμου είναι να χρησιμοποιεί δυαδικές τιμές. Αυτή τη φορά, σε αντίθεση με την προηγούμενη μορφή, δεν χρησιμοποιούνται δισδιάστατοί πίνακες αλλά τρισδιάστατοι. Η κάθε λύση αποθηκεύεται ως ένας δισδιάστατος πίνακας 10 στηλών μέσα σε ένα τρισδιάστατο πίνακα, οπού η $1^{η}$ στήλη αντιπροσωπεύει τη τιμή του προϊόντος και παίρνει τιμές από 0 έως 6 και οι υπόλοιπες 9 στήλες είναι τα χαρακτηριστικά της τσάντας και παίρνουν τιμή 0 ή 1.

5.1.1 5 τσάντες για πραγματικά δεδομένα (Bat Algorithm)

Αρχικά θα δούμε τα αποτελέσματα που παράγει ο Bat algorithm για 5 τσάντες σε σύγκρισή με τους αλγόριθμους που είχαν υλοποιηθεί για την δημοσίευση των Belloni και Freund. Για να μπορεί να γίνει σύγκριση μεταξύ των αποτελεσμάτων χρησιμοποιήθηκε για όλους τους αλγορίθμους ο ίδιος αριθμός αξιολόγησης της αντικειμενικής συνάρτησης. Τρέξαμε τον κάθε αλγόριθμο 10 φορές και κρατήσαμε την καλύτερη λύση καθώς και το μέσο όρο των λύσεων.

Τα αποτελέσματα φαίνονται στον *Πίνακα 1.*

	Καλύτερη λύση	Ποσοστό καλύτερης λύσης %	Μέσο κέρδος	Ποσοστό καλύτερης λύσης Μ.Ο.	Πλήθος αξιολογήσεων αντικειμενικής
Bat (Binary)	12175	99.58	12028	98.38	500.000
Bat (Discrete)	11811	96.60	11276	92.22	500.000
Coordinate ascent(one-opt)	11980	97.98	11238	91.91	500.000
Coordinate ascent(two-opt)	12021	98.32	11874	97.12	500.000
Coordinate ascent(three-opt)	12056	98.60	11986	98.03	500.000
Genetic	12202	99.80	12038	98.46	500.000
Divide and conquer	12175	99.58	12153	99.40	500.000
Greedy heuristic	12035	98.43	12035	98.43	500.000
Simulated annealing	12226	100.00	12220	99.95	500.000
Product swapping	12219	99.94	12218	99.93	500.000
DP heuristic	11882	97.18	11498	94.04	500.000
Beam search	12035	98.43	11528	94.29	500.000
Nested partitions	12035	98.43	11781	96.36	500.000

Πίνακας 1

Παρατηρούμε ότι ο δυαδικός αλγόριθμος Bat βγάζει καλύτερα αποτελέσματα από τον διακριτό. Επιπλέον είναι φανερό ότι ο αλγόριθμος Bat δεν καταφέρνει να βγάλει καλύτερα αποτελέσματα από τον Genetic και τον Simulated annealing, οι οποίοι θεωρούνται ιδανικοί για τέτοιου είδους προβλήματα, πλησιάζει όμως σε πολύ υψηλό ποσοστό την βέλτιστη λύση (99,58%) και έχει μεγάλό ποσοστό μέσης λύσης (98.38%). Ο Bat Algorithm προσεγγίζει καλύτερα τη βέλτιστη λύση από τους περισσοτέρους αλγορίθμους με τους οποίους συγκρίνεται.

5.1.2 10 τσάντες για πραγματικά δεδομένα (Bat Algorithm)

Στη προηγούμενη ενότητα δείξαμε τα αποτελέσματα για μια γραμμή παραγωγής 5 τσαντών. Τώρα θα τρέξουμε τα ίδια δεδομένα από την δημοσίευση των Belloni και Freund αλλά αυτή τη φορά θέλουμε να δημιουργήσουμε μια γραμμή που αποτελείται από 10

τσάντες. Ο κάθε αλγόριθμος έτρεξε πάλι 10 φορές και καταγράφτηκε το μέσο όρο καθώς και η καλύτερη λύση που βρέθηκε.

Τα αποτελέσματα φαίνονται στο *Πίνακα 2.*

	Καλύτερη λύση	**Ποσοστό καλύτερης λύσης %**	**Μέσο κέρδος**	**Ποσοστό καλύτερης λύσης Μ.Ο.**	**Πλήθος αξιολογήσεων αντικειμενικής**
Bat (Binary)	13484	98.04	13217	96.10	500.000
Bat (Discrete)	12624	91.79	12284	89.31	500.000
Coordinate ascent(one-opt)	13095	95.21	12938	94.07	500.000
Coordinate ascent(two-opt)	13594	98.84	13410	97.50	500.000
Genetic	13636	99.14	13534	98.40	500.000
Divide and conquer	13649	99.24	13594	98.84	500.000
Greedy heuristic	13381	97.29	13381	97.29	500.000
Simulated annealing	13753	100.00	13748	99.96	500.000
Product swapping	13733	99.85	13731	99.84	500.000
Nested partitions	13414	97.53	13026	94.71	500.000

Πίνακας 2

Από το *Πίνακα 2* φαίνεται ότι τα ποσοστά εύρεσης της καλύτερης λύσης των αλγορίθμων μειωθήκαν στο πρόβλημα για 10 τσάντες σε σύγκριση με το πρόβλημα 5 τσαντών. Αυτό συμβαίνει διότι με την αύξηση του αριθμού των προϊόντων της γραμμής παραγωγής που θέλουμε να δημιουργήσουμε αυξήθηκε κατά πολύ η πολυπλοκότητα του προβλήματος.

5.1.3 Δοκιμασία ευστάθειας (Bat algorithm)

Έως τώρα υποθέταμε ότι οι μερικές αξίες που αποδίδουν σε κάθε προϊόν οι καταναλωτές είναι σωστές. Σε πραγματικά προβλήματα όμως κάτι τέτοιο δεν ισχύει και για αυτό το λόγο πρέπει να υπολογιστεί και το σφάλμα που θα υπάρχει κατά τον προσδιορισμό των μερικών αξιών από τους καταναλωτές. Για το σκοπό αυτό έχουν αναπτυχθεί διάφορες στατιστικές τεχνικές που μετράνε την απόκλιση του δείγματος από τις πραγματικές προτιμήσεις.

Στο συγκεκριμένο πρόβλημα δημιουργούμε ένα *δείγμα σύγχυσης* το οποίο διαταράσσει το δείγμα με τις καταναλωτικές προτιμήσεις που χρησιμοποιήθηκε στις προηγούμενες

μεθόδους. Η συγκεκριμένη μέθοδος θεώρει ότι στο δείγμα υπάρχει ένα σφάλμα εκτίμησης ίσο με ε. Έχουμε λοιπόν,

$$u'_{i,j} = u_{i,j} + \varepsilon_{i,j} \quad ,\text{οπού:}$$

- $u_{i,j}$ η πραγματική προτίμηση του καταναλωτή i για το χαρακτηριστικό j.
- $\varepsilon_{i,j}$ σφάλμα μέτρησης, το οποίο είναι μια ανεξάρτητη μεταβλητή που ακολουθεί την κανονική κατανομή.
- $u'_{i,j}$ η νέα προτίμηση του καταναλωτή με το σφάλμα εκτίμησης.

Τρέχουμε πάλι τους αλγορίθμους 10 φορές με τα νέα δεδομένα για το πρόβλημα με τις 5 τσάντες. Καταγράφουμε το μέσο όρο και τις καλύτερες λύσεις.

Τα αποτελέσματα φαίνονται στο *Πίνακα 3*

	Καλύτερη λύση	**Ποσοστό καλύτερης λύσης %**	**Μέσο κέρδος**	**Ποσοστό καλύτερης λύσης Μ.Ο.**	**Πλήθος αξιολογήσεων αντικειμενικής**
Bat (Binary)	11795	96.47	11458	93.71	500.000
Bat (Discrete)	11318	92.57	11022	90.15	500.000
Coordinate ascent(one-opt)	10955	89.60	10792	88.27	500.000
Coordinate ascent(two-opt)	11560	94.55	11318	92.57	500.000
Coordinate ascent(three-opt)	11780	96.35	11624	95.07	500.000
Genetic	11812	96.61	11747	96.08	500.000
Divide and conquer	11784	96.38	11759	76.20	500.000
Greedy heuristic	11232	91.86	11232	91.86	500.000
Simulated annealing	11849	96.91	11836	96.81	500.000
Product swapping	11780	96.35	11761	96.19	500.000
DP heuristic	11551	94.47	11218	91.17	500.000
Beam search	11418	93.39	10875	88.94	500.000
Nested partitions	11232	91.86	10740	87.84	500.000

Πίνακας 3

5.1.4 Δεδομένα προσομοίωσης

Σε αυτό το σημείο θα επιλύσουμε το πρόβλημα βέλτιστου σχεδιασμού γραμμής προϊόντων χρησιμοποιώντας προσομοιωμένα δεδομένα. Σε αντίθεση με το τρόπο που λύναμε έως τώρα τα προβλήματα, δηλαδή με πραγματικά δεδομένα που είχαν αποκτηθεί μετρά από έρευνα αγοράς με τη χρήση ερωτηματολόγιών, αυτή τη φορά θα επιλύσουμε το πρόβλημα για διαφορετικά μεγέθη. Πιο συγκεκριμένα θα έχουμε 4 μεγέθη τα οποία θα αλλάζουν και θα δημιουργούν 12 διαφορετικά σετ δεδομένων. Αυτά είναι:

- Αριθμός τσαντών με επιλογές 3 ή 4
- Αριθμός καταναλωτών με επιλογές 50 ή 100
- Επίπεδα χαρακτηριστικών με επιλογές 2,3,5 ή 8
- Χαρακτηριστικά προϊόντων με επιλογές 3,5 ή 7

Δημιουργούνται τα εξής 12 προβλήματα:

Αριθμός τσαντών	Αριθμός καταναλωτών	Επίπεδα χαρακτηριστικών	Χαρακτηριστικά προϊόντων
4	50	5	3
4	100	5	3
4	50	3	5
4	100	3	5
4	50	2	7
4	100	2	7
3	50	8	3
3	100	8	3
3	50	5	5
3	100	5	5
3	50	3	7
3	100	3	7

Τρέχουμε πάλι τους αλγορίθμους 10 φορές με τα νέα δεδομένα για το πρόβλημα με τα δεδομένα προσομοίωσης. Καταγράφουμε το μέσο όρο και τις καλύτερες λύσεις. Σε αυτή τη περίπτωση τρέχουμε μόνο τη διακριτή έκδοση του Bat algorithm καθώς ο δυαδικός δε βγάζει καθόλου καλά αποτελέσματα για simulated data.

Τα αποτελέσματα φαίνονται στον *Πίνακα 4.*

	Μέσο ποσοστό καλύτερης λύση	**Συχνότητα εύρεσης >95% της καλύτερης λύσης**	**Πλήθος αξιολογήσεων αντικειμενικής**
Bat (Discrete)	96.37	94.48	100.000
Coordinate ascent(one-opt)	94.52	67.49	100.000
Coordinate ascent(two-opt)	95.88	88.28	100.000
Coordinate ascent(three-opt)	96.27	91.41	100.000
Genetic	99.82	100.00	100.000
Divide and conquer	97.17	93.10	100.000
Greedy heuristic	95.74	91.21	100.000
Simulated annealing	100.00	100.00	100.000
Product swapping	97.31	92.84	100.000
DP heuristic	95.27	88.72	100.000
Beam search	98.40	95.11	100.000
Nested partitions	93.59	58.23	100.000

Πίνακας 4

5.2 Cuckoo Search Algorithm

Όπως και στο Bat algorithm έτσι και για το Cuckoo Search Algorithm υλοποιήθηκαν δύο μορφές του αλγορίθμου με διαφορετικούς τρόπους επεξεργασίας και αποθήκευσης των δεδομένων. Η φιλοσοφία είναι η ίδια όπως και προηγούμενος δηλαδή να συγκριθούν μεταξύ τους αυτές οι δυο μορφές του αλγορίθμου και να δούμε ποια παράγει καλύτερα αποτελέσματα για το συγκεκριμένο πρόβλημα βελτιστοποίησης.

Η μια μορφή είναι να δουλεύει ο αλγόριθμός με διακριτές τιμές. Στην περίπτωση αυτή οι λύσεις αποθηκεύονται σε δισδιάστατους πίνακες και η κάθε λύση αντιπροσωπεύεται με έναν αριθμό τσάντας από το 1 έως το 3584, όσες είναι δηλαδή και οι διαφορετικές τσάντες που μπορούν να δημιουργηθούν.

Η δεύτερη μορφή του αλγορίθμου είναι να χρησιμοποιεί δυαδικές τιμές. Αυτή τη φορά, σε αντίθεση με την προηγούμενη μορφή, δεν χρησιμοποιούνται δισδιάστατοί πίνακες αλλά τρισδιάστατοι. Η κάθε λύση αποθηκεύεται ως ένας δισδιάστατος πίνακας 10 στηλών μέσα σε ένα τρισδιάστατο πίνακα, οπού η $1^{η}$ στήλη αντιπροσωπεύει τη τιμή του προϊόντος και παίρνει τιμές από 0 έως 6 και οι υπόλοιπες 9 στήλες είναι τα χαρακτηριστικά της τσάντας και παίρνουν τιμή 0 ή 1.

5.2.1 5 τσάντες για πραγματικά δεδομένα (Cuckoo search)

Αρχικά θα δούμε τα αποτελέσματα που παράγει ο Cuckoo Search algorithm για 5 τσάντες σε σύγκρισή με τους αλγόριθμους που είχαν υλοποιηθεί για την δημοσίευση των Belloni και Freund. Για να μπορεί να γίνει σύγκριση μεταξύ των αποτελεσμάτων χρησιμοποιήθηκε για όλους τους αλγορίθμους ο ίδιος αριθμός αξιολόγησης της αντικειμενικής συνάρτησης. Τρέξαμε τον κάθε αλγόριθμο 10 φορές και κρατήσαμε την καλύτερη λύση καθώς και το μέσο όρο των λύσεων.

Τα αποτελέσματα φαίνονται στον *Πίνακα 5*

	Καλύτερη λύση	Ποσοστό καλύτερης λύσης %	Μέσο κέρδος	Ποσοστό καλύτερης λύσης Μ.Ο.	Πλήθος αξιολογήσεων αντικειμενικής
Cuckoo search (Binary)	11543	94.41	11097	90.76	500.000
Cuckoo search (Discrete)	12056	98.60	11823	96.70	500.000
Coordinate ascent(one-opt)	11946	97.70	11248	92.00	500.000
Coordinate ascent(two-opt)	12021	98.32	11829	96.75	500.000
Coordinate ascent(three-opt)	12056	98.60	11988	98.05	500.000
Genetic	12202	99.80	12051	98.56	500.000
Divide and conquer	12175	99.58	12155	99.41	500.000
Greedy heuristic	12035	98.43	12035	98,43	500.000
Simulated annealing	12226	100.00	12221	99,95	500.000
Product swapping	12219	99.94	12217	99.92	500.000
DP heuristic	11882	97.18	11440	93.57	500.000
Beam search	12035	98.43	11588	94.78	500.000
Nested partitions	12035	98.43	11787	96.40	500.000

Πίνακας 5

Παρατηρούμε ότι ο διακριτός αλγόριθμος Cuckoo search βγάζει καλύτερα αποτελέσματα από τον δυαδικό. Επιπλέον μπορούμε να δούμε και σε αυτή τη περίπτωση ότι ο αλγόριθμος Cuckoo search δεν καταφέρνει να βγάλει καλύτερα αποτελέσματα από τον Genetic και τον Simulated annealing, οι οποίοι θεωρούνται ιδανικοί για τέτοιου είδους προβλήματα, πλησιάζει όμως σε μεγάλο ποσοστό την βέλτιστη λύση (98,60%) και αποδίδει καλύτερα από τους περισσοτέρους αλγορίθμους με τους οποίους συγκρίνεται.

5.2.2 10 τσάντες για πραγματικά δεδομένα (Cuckoo search)

Τώρα θα τρέξουμε τα ίδια δεδομένα για μια γραμμή παραγωγής 10 τσαντών. Ο κάθε αλγόριθμος έτρεξε 10 φορές και καταγράφτηκε το μέσο όρο καθώς και η καλύτερη λύση που βρέθηκε.

Τα αποτελέσματα φαίνονται στον *Πίνακα 6*.

	Καλύτερη λύση	Ποσοστό καλύτερης λύσης %	Μέσο κέρδος	Ποσοστό καλύτερης λύσης Μ.Ο.	Πλήθος αξιολογήσεων αντικειμενικής
Cuckoo search (Binary)	12133	88.22	11583	84.22	500.000
Cuckoo search (Discrete)	13381	97.29	13079	95.09	500.000
Coordinate ascent(one-opt)	13052	94.90	12911	93.87	500.000
Coordinate ascent(two-opt)	13571	98.67	13438	97.70	500.000
Genetic	13642	99.19	13526	98.34	500.000
Divide and conquer	13649	99.24	13571	98.67	500.000
Greedy heuristic	13381	97.29	13381	97.29	500.000
Simulated annealing	13753	100.00	13749	99.97	500.000
Product swapping	13733	99.85	13729	99.82	500.000
Nested partitions	13398	97.41	13056	94.93	500.000

Πίνακας 6

5.2.3 Δοκιμασία ευστάθειας (Cuckoo search)

Στο συγκεκριμένο πρόβλημα δημιουργούμε ένα *δείγμα σύγχυσης* το οποίο διαταράσσει το δείγμα με τις καταναλωτικές προτιμήσεις που χρησιμοποιήθηκε στις προηγούμενες μεθόδους. Η συγκεκριμένη μέθοδος θεώρει ότι στο δείγμα υπάρχει ένα σφάλμα εκτίμησης ίσο με ε. Έχουμε λοιπόν,

$$u'_{i,j} = u_{i,j} + \varepsilon_{i,j} \quad \text{, οπού:}$$

- $u_{i,j}$ η πραγματική προτίμηση του καταναλωτή i για το χαρακτηριστικό j.
- $\varepsilon_{i,j}$ σφάλμα μέτρησης, το οποίο είναι μια ανεξάρτητη μεταβλητή που ακολουθεί την κανονική κατανομή.

- $u'_{i,j}$ η νέα προτίμηση του καταναλωτή με το σφάλμα εκτίμησης.

Τρέχουμε πάλι τους αλγορίθμους 10 φορές με τα νέα δεδομένα για το πρόβλημα με τις 5 τσάντες. Καταγράφουμε το μέσο όρο και τις καλύτερες λύσεις.

Τα αποτελέσματα φαίνονται στον *Πίνακα 7*.

	Καλύτερη λύση	**Ποσοστό καλύτερης λύσης %**	**Μέσο κέρδος**	**Ποσοστό καλύτερης λύσης Μ.Ο.**	**Πλήθος αξιολογήσεων αντικειμενικής**
Cuckoo search (Binary)	11120	90.95	10972	89.74	500.000
Cuckoo search (Discrete)	11551	94.47	11273	92.20	500.000
Coordinate ascent(one-opt)	10955	89.60	10726	87.73	500.000
Coordinate ascent(two-opt)	11560	94.55	11341	92.76	500.000
Coordinate ascent(three-opt)	11780	96.35	11647	95.26	500.000
Genetic	11836	96.81	11738	96.00	500.000
Divide and conquer	11784	96.38	11761	96.19	500.000
Greedy heuristic	11232	91.86	11232	91.86	500.000
Simulated annealing	11849	96.91	11832	96.77	500.000
Product swapping	11780	96.35	11763	96.21	500.000
DP heuristic	11524	94.25	11201	91.61	500.000
Beam search	11418	93.39	10947	89.53	500.000
Nested partitions	11270	92.18	10714	87.63	500.000

Πίνακας 7

Παρατηρούμε ότι το σφάλμα των καταναλωτικών προτιμήσεων επηρεάζει σε ένα βαθμό τις λύσεις και καθιστά αδύνατο την εύρεση της βέλτιστης λύσης, παρόλα αυτά οι αλγόριθμοι εξακολουθούν να πλησιάζουν σε ένα ικανοποιητικό βαθμό την βέλτιστη λύση. Σε αυτή την περίπτωση όπως και στις προηγούμενες ο διακριτός Cuckoo search algorithm πλησιάζει καλύτερη τη βέλτιστη λύση από τον δυαδικό και σε υψηλό ποσοστό (σχεδόν 95%) της καλύτερης λύσης.

5.2.4 Δεδομένα προσομοίωσης

Σε αυτό το σημείο θα επιλύσουμε το πρόβλημα βέλτιστου σχεδιασμού γραμμής προϊόντων χρησιμοποιώντας προσομοιωμένα δεδομένα. Σε αντίθεση με το τρόπο που λύναμε έως τώρα τα προβλήματα, δηλαδή με πραγματικά δεδομένα που είχαν αποκτηθεί μετρά από έρευνα αγοράς με τη χρήση ερωτηματολόγιών, αυτή τη φορά θα επιλύσουμε το πρόβλημα για διαφορετικά μεγέθη. Πιο συγκεκριμένα θα έχουμε 4 μεγέθη τα οποία θα αλλάζουν και θα δημιουργούν 12 διαφορετικά σετ δεδομένων. Αυτά είναι:

- Αριθμός τσαντών με επιλογές 3 ή 4
- Αριθμός καταναλωτών με επιλογές 50 ή 100
- Επίπεδα χαρακτηριστικών με επιλογές 2,3,5 ή 8
- Χαρακτηριστικά προϊόντων με επιλογές 3,5 ή 7

Δημιουργούνται τα εξής 12 προβλήματα:

Αριθμός τσαντών	Αριθμός καταναλωτών	Επίπεδα χαρακτηριστικών	Χαρακτηριστικά προϊόντων
4	50	5	3
4	100	5	3
4	50	3	5
4	100	3	5
4	50	2	7
4	100	2	7
3	50	8	3
3	100	8	3
3	50	5	5
3	100	5	5
3	50	3	7
3	100	3	7

Τρέχουμε πάλι τους αλγορίθμους 10 φορές με τα νέα δεδομένα για το πρόβλημα με τα δεδομένα προσομοίωσης. Καταγράφουμε το μέσο όρο και τις καλύτερες λύσεις.

Τα αποτελέσματα φαίνονται στον *Πίνακα 8*.

	Μέσο ποσοστό καλύτερης λύση	**Συχνότητα εύρεσης >95% της καλύτερης λύσης**	**Πλήθος αξιολογήσεων αντικειμενικής**
Cuckoo search (Discrete)	95.93	90.64	100.000
Coordinate ascent(one-opt)	94.52	67.49	100.000
Coordinate ascent(two-opt)	95.88	88.28	100.000
Coordinate ascent(three-opt)	96.27	91.41	100.000
Genetic	99.82	100.00	100.000
Divide and conquer	97.17	93.10	100.000
Greedy heuristic	95.74	91.21	100.000
Simulated annealing	100.00	100.00	100.000
Product swapping	97.31	92.84	100.000
DP heuristic	95.27	88.72	100.000
Beam search	98.40	95.11	100.000
Nested partitions	93.59	58.23	100.000

Πίνακας 8

6. Επίλογος

6.1 Συμπεράσματα

Στην παρούσα εργασία έγινε πλήρης έλεγχος της αποτελεσματικότητας δυο νέων μεθευρετικών αλγορίθμων σε ένα πρόβλημα βελτιστοποιησης μιας γραμμής παραγωγής. Είναι η πρώτη φορά στη βιβλιογραφία που εφαρμόστηκαν ο αλγόριθμος Cuckoo search και ο Bat algorithm σε ένα τέτοιο πρόβλημα. Τα αποτελέσματα είναι αρκετά καλά και για τους δυο αλγορίθμους και ειδικά στη περίπτωση του Bat algorithm η βέλτιστη λύση προσεγγίστηκε σε πολύ υψηλό βαθμό.

Παρατηρούμε ότι για τον Bat algorithm η υλοποίηση που δουλεύει σε δυαδικές τιμές παράγει αρκετά καλύτερα αποτελέσματα από την έκδοση που δουλεύει με διακριτές τιμές για όλες τις περιπτώσεις των πραγματικών δεδομένων. Μόνη εξαίρεση είναι τα προσομοιωμένα δεδομένα οπού ο διακριτός αλγόριθμος δίνει καλύτερα αποτελέσματα.

Ο Cuckoo search algorithm σε αντίθεση με τον Bat algorithm παράγει καλύτερα αποτελέσματα για την διακριτή υλοποίηση σε όλες τις περιπτώσεις των πραγματικών δεδομένων αλλά και στα δεδομένα προσομοίωσης.

Παρόλα αυτά οι αλγόριθμοι στην παρούσα μορφή τους δε κατάφεραν να ξεπεράσουν σε αποτελεσματικότατά τον genetic algorithm και τον Simulated annealing που αποδίδουν ιδανικά σε τέτοιους είδους προβλήματα, για αυτό αλώστε είναι και οι πιο συχνά χρησιμοποιούμενοι σε αυτή τη κατηγορία προβλημάτων.

6.2 Μελλοντική έρευνα

Στη συνέχεια θα δούμε κάποιες εναλλακτικές μορφές των αλγορίθμων που έχουν προταθεί, καθώς και κάποιους υβριδικούς αλγορίθμους που περιέχουν τον Cuckoo search algorithm και Bat algorithm, οι οποίοι αξίζει να ερευνηθούν και ενδεχόμενος να παρέχουν καλύτερες λύσεις στο πρόβλημα βελτιστοποίησης γραμμής προϊόντων.

6.2.1 Hybrid bat algorithm with artificial bee colony

Ο Υβριδικός αυτός αλγόριθμος βασίζεται στον Bat algorithm(BA) που δημοσίευσε ο Xin-She Yang το 2010 και στον Artificial bee colony algorithm(ABC) που προτάθηκε από τον Karaboga το 2008. Η ιδέα του βασίζεται στην αντικατάσταση των χειρότερων λύσεων του ενός αλγόριθμου με καλύτερες λύσεις από τον άλλον οι οποίοι δουλεύουν παράλληλα. Ο πληθυσμός των λύσεων διαιρείται σε υποομάδες πληθυσμών και κατασκευάζεται έτσι ένας αλγόριθμος παράλληλης επεξεργασίας. Κάθε μια από τις υποομάδες πληθυσμού εξελίσσεται αυτόνομα από τις υπόλοιπες ενώ γίνονται οι επαναλήψεις. Η μόνη περίπτωση στην οποία ανταλλάσσονται πληροφορίες από τις υποομάδες είναι όταν ενεργοποιείται μια στρατηγική επικοινωνίας. Αυτό έχει ως αποτέλεσμα, ο αλγόριθμος παράλληλης επεξεργασίας να εκμεταλλεύεται τα δυνατά σημεία του κάθε αλγορίθμου και να αντικαθιστά τις αδύναμες λύσεις του ενός με τις καλύτερες λύσεις του άλλου.

Τα βήματα που ακολουθεί ο προτεινόμενος αλγόριθμος είναι τα εξής:

1. **Αρχικοποίηση:** Παραγωγή ξεχωριστού πληθυσμού λύσεων για τον Bat algorithm και για τον Artificial Bee colony. Ορίζεται η επανάληψη R στην οποία εκτελείται η στρατηγική επικοινωνίας.

2. **Αξιολόγηση:** Αξιολογούνται οι λύσεις του πληθυσμού για τον BA και ABC. Οι αξιολογήσεις του ενός αλγορίθμου εκτελούνται ανεξάρτητα από τις αξιολογήσεις του άλλου.

3. **Ενημέρωση**: Ενημερώνονται οι θέσεις κάθε νυχτερίδας και η καλύτερη πηγή φαγητού για κάθε μέλισσά.

4. **Στρατηγική Επικοινωνίας**: Μετανάστευση των καλύτερων νυχτερίδων στον πληθυσμό του ABC και αντικατάσταση των χειρότερων artificial agents. Αντιστοίχως, μετανάστευση των καλύτερων artificial agents στον πληθυσμό του BA και αντικατάσταση των χειρότερων νυχτερίδων.

5. **Τερματισμός**: Επανάληψη των βημάτων 2 έως 5 μέχρι να εκπληρωθεί το κριτήριο τερματισμού. Καταγραφή των καλύτερων λύσεων και θέσεων.

6.2.2 Hybrid cuckoo search algorithm with Nelder mead

Ο προτεινόμενος αλγόριθμος Hybrid cuckoo search algorithm with Nelder mead(HCSNM) είναι ένας υβριδικός αλγόριθμος που δημιουργείται συνδυάζοντας τον κλασσικό αλγόριθμο Cuckoo search με την μέθοδο Nelder mead. Σκοπός του είναι να ξεπεράσει την αργή σύγκλιση του Cuckoo search. Η μέθοδος Nelder mead, η οποία προτάθηκε από τον John Nelder το 1965, επιταχύνει την έρευνα της βέλτιστης λύσης του αλγορίθμου, βελτιώνει τη σύγκλιση του και εξασφαλίζει ότι ο αλγόριθμος Cuckoo search δε θα συνεχίσει να κάνει επαναλήψεις χωρίς να βελτιώνει τις λύσεις του.

Για να γίνει πλήρως κατανοητό πως δουλεύει ο προτεινόμενος αλγόριθμος, θα δείξουμε πρώτα τα βήματα του κάθε αλγορίθμου που θα χρησιμοποιηθεί για να δημιουργηθεί ο HCSNM ξεχωριστά και συνέχεια θα αναφερθεί πως συνδυάζονται για να δημιουργηθεί ο υβριδικός αλγόριθμος.

Τα βήματα που ακολουθεί η μέθοδος Nelder Mead:

1. Let x_i denote the list of vertices in the current simplex, $i = 1, \ldots, n+1$.
2. Order. Order and re-label the $n+1$ vertices from lowest function value $f(x_1)$ to highest function value $f(x_{n+1})$ so that $f(x_1) \leq f(x_2) \leq \ldots \leq f(x_{n+1})$.
3. Reflection. Compute the reflected point x_r by
$x_r = \bar{x} + \rho(\bar{x} - x_{(n+1)})$, where $\bar{x}$ is the centroid of the n best points,
$\bar{x} = \sum(x_i/n), i = 1, \ldots, n$.
if $f(x_1) \leq f(x_r) < f(x_n)$ **then**
Replace x_{n+1} with the reflected point x_r and go to Step 7.
end if
4. Expansion.
if $f(x_r) < f(x_1)$ **then**
Compute the expanded point x_e by $x_e = \bar{x} + \chi(x_r - \bar{x})$.
end if
if $f(x_e) < f(x_r)$ **then**
Replace x_{n+1} with x_e and go to Step 7.
else
Replace x_{n+1} with x_r and go to Step 7.
end if
5. Contraction.
if $f(x_r) \geq f(x_n)$ **then**
Perform a contraction between $\bar{x}$ and the best among x_{n+1} and x_r.
end if
if $f(x_n) \leq f(x_r) < f(x_{n+1})$ **then**
Calculate $x_{oc} = \bar{x} + \tau(x_r - \bar{x})$ {*Outside contract.*}
end if
if $f(x_{oc}) \leq f(x_r)$ **then**
Replace x_{n+1} with x_{oc} and go to Step 7.
else
Go to Step 6.
end if
if $f(x_r) \geq f(x_{(n+1)}$ **then**
Calculate $x_{ic} = \bar{x} + \tau(x_{n+1} - \bar{x})$. {*Inside contract*}
end if
if $f(x_{ic}) \geq f(x_{(n+1)}$ **then**
Replace x_{n+1} with x_{ic} and go to Step 7.
else
Go to Step 6.
end if
6. Shrink. Evaluate the n new vertices
$x' = x_1 + \phi(x_i - x_1), i = 2, \ldots, n+1$.
Replace the vertices $x_2, \ldots, x_{n+1}$ with the new vertices $x'_2, \ldots, x'_{n+1}$.
7. Stopping Condition. Order and re-label the vertices of the new simplex as $x_1, x_2, \ldots, x_{n+1}$ such that $f(x_1) \leq f(x_2) \leq \ldots \leq f(x_{n+1})$
if $f(x_{n+1}) - f(x_1) < \epsilon$ **then**
Stop, where $\epsilon > 0$ is a small predetermined tolerance.
else
Go to Step 3.
end if

Τα βήματα του Cuckoo search όπως τα είδαμε και προηγουμένως:

1: Set the initial value of the host nest size n, probability $p_a \in [0,1]$ and maximum number of iterations Max_{itr}.
2: Set $t := 0$. {Counter initialization}
3: **for** $(i = 1 : i \leq n)$ **do**
4: Generate initial population of n host $x_i^{(t)}$. {n is the population size}
5: Evaluate the fitness function $f(x_i^{(t)})$.
6: **end for**
7: **repeat**
8: Randomly generate a new solution (Cuckoo) $x_i^{(t+1)}$ by Lévy flight.
9: Evaluate the fitness function of a solution $x_i^{(t+1)}$ $f(x_i^{(t+1)})$
10: Randomly choose a nest x_j among n solutions.
11: **if** $(f(x_i^{(t+1)}) > f(x_j^{(t)}))$ **then**
12: Replace the solution x_j with the solution $x_i^{(t+1)}$
13: **end if**
14: Abandon a fraction p_a of worse nests.
15: Build new nests at new locations using Lévy flight a fraction p_a of worse nests
16: Keep the best solutions (nests with quality solutions)
17: Rank the solutions and find the current best solution
18: Set $t = t + 1$.
19: **until** $(t \geq Max_{itr})$. {Termination criteria are satisfied}
20: Produce the best solution.

Τα βήματα του HCSNM είναι τα ιδιά βήματα που ακολουθεί ο Cuckoo search μέχρι το βήμα 19 και έπειτα εφαρμόζεται η μέθοδος Nelder Mead ως μια διαδικασία εντατικοποίησης προκειμένου να βελτιωθεί η βέλτιστη λύση που βρέθηκε στα προηγούμενα βήματα. Ο HCSNM αν και δημοσιεύτηκε μόλις πρόσφατα είναι πολλά υποσχόμενος καθώς έχει τη δυνατότητα να λύνει προβλήματα βελτιστοποίησης πιο γρήγορα και αποτελεσματικά από τους περισσότερους αλγορίθμους.

Βιβλιογραφία

[1] Alexandre Belloni, Robert Freund, Matthew Selove, Duncan Simester: **Optimizing Product Line Designs: Efficient Methods and Comparisons (2008)**

[2] Xin-She Yang: **A New Metaheuristic Bat-Inspired Algorithm (2010)**

[3] Ιωάννης Μαρινάκης, Μαγδαληνή Μαρινάκη, Νικόλαος Ματσατσίνης, Κωνσταντίνος Ζοπουνίδης: **Μεθευρετικοί και Εξελικτικοί Αλγόριθμοι σε Προβλήματα Διοικητικής Επιστήμης (2011)**

[4] Στέλιος Τσαφαράκης, Νικόλαος Ματσατσίνης: **Designing Optimal Products: Algorithms and Systems (2010)**

[5] Ιωάννης Μαρινάκης, Αθανάσιος Μυγδαλάς: **Σχεδιασμός και Βελτιστοποίηση της Εφοδιαστικής Αλυσίδας (2008)**

[6] Xin-She Yang, Suash Deb: **Cuckoo Search via Lévy flights (2009)**

[7] Amir Hossein, Gandomi, Xin-She Yanh, Amir Hossein Alavi: **Cuckoo search algorithm: a metaheuristic approach to solve structural optimization problems (2011)**

[8] Paul Green, Abba Krieger: **A Consumer-Based Approach to Designing Product Line Extensions (1987)**

[9] M. Mareli, B. Twala: **An adaptive Cuckoo search algorithm for optimization (2017)**

[10] Xin-She Yang: **Cuckoo Search via Levy Flights (2009)**

[11] Dimitris Bertsimas: **Exact First-Choice Product Line Optimization (2017)**

[12] Xin-She Yang: **Exact First-Choice Product Line Optimization (2010)**

[13] Xin-She Yang, Amir Hossein Gandomi; **Bat algorithm: a novel approach for global engineering optimization (2011)**

[14] George Batlas, Peter Doyle: **Random utility models in marketing research: a survey (2001)**

[15] Emile Aarts, Jan Lenstra: **Local Search in Combinatorial Optimization (1997)**

[16] Dimitris Bertsimas, John Tsiltsiklis: **Introduction to linear Optimization (1997)**

[17**]** Paul Green, Abba Krieger: **An application of a product positioning model to pharmaceutical products (1992)**

[18] Γεώργιος Μπαλαράς: **Βέλτιστος Σχεδιασμός Γραμμής Προϊόντων με χρήση του Tabu Search (2015)**

[19] Gregory Dobson: **Heuristics for Pricing and Positioning a Product-Line Using Conjoint and Cost Data (1993)**

[20] Paul Green, Abba Krieger: **Choice rules and sensitivity analysis in conjoint simulators (1988)**

[21] Paul Green, Abba Krieger: **Conjoint analysis with product positioning applications (1993)**

[22] Paul Green, Abba Krieger, Robert Zelnio: **A componential segmentation model with optimal product design features (1989)**

[23] Ramin Rajabioun: **Cuckoo Optimization Algorithm (2011)**

[24] E. Taillard: ***Some efficient heuristic methods for the flow shop sequencing problem (1990)***

[25] Pierre Hansen, Nenad Mladenociv: ***Variable neighborhood search: Principles and applications (1999)***

[26] Βασιλική Νταμαδάκη: **Εφαρμογή αλγορίθμων τεχνητού ανοσοποιητικού συστήματος για την επίλυση του προβλήματος του βέλτιστου σχεδιασμού γραμμής προϊόντων (2015)**

[27] Δημήτριος Γονιδάκης: **Εφαρμογές του αλγορίθμου νυχτερίδας σε πολυκριτηριακά προβλήματα βελτιστοποίησης (2014)**

[28] M. Arias: **Echolocation in bats (2003)**

[29] J. R Koza: Genetic Programming: **On the Programming of Computers by Means of Natural Selection**

[30] D. Bertsekas**: Nonlinear programming (1999)**

[31] Allan Shocker, V. Srinivasan: **A Consumer-Based Methodology for the identification of New Product Ideas (1974)**

[32] Georgia Alexouda, Konstantinos Paparrizos: **A genetic algorithm approach to the product line design problem using the seller's return criterion: An extensive comparative computational study (2001)**

[33] Bernd Stauß, Wolfgang Gaul: **Product Line Optimization as a Two Stage Problem (2004)**

[34] Ahmed Ali, Mohamed Tawhid: **A hybrid cuckoo search algorithm with Nelder Mead method for solving global optimization problems (2016)**

[35] J. Lesourne: **A theory of the individual for economic analysis (1977)**