ΕΛΛΗΝΙΚΟ ΑΝΟΙΧΤΟ ΠΑΝΕΠΙΣΤΗΜΙΟ

ΣΧΟΛΗ ΘΕΤΙΚΩΝ ΕΠΙΣΤΗΜΩΝ ΚΑΙ ΤΕΧΝΟΛΟΓΙΑΣ

Πρόγραμμα σπουδών:

Μεταπτυχιακή Εξειδίκευση στα Πληροφοριακά Συστήματα

Διπλωματική Εργασία:

Εφαρμογή σύγχρονων τεχνικών υπολογιστικής νοημοσύνης Cat Swarm Optimization (CSO) για την αποδοτική επίλυση του προβλήματος Curriculum based Course Timetabling

Όνομα: Παρθενία

Επώνυμο: Τσόγκα

Α.Μ.: 131071

Επιβλέπων Καθηγητής:

Γρηγόριος Μπεληγιάννης

Πάτρα, Σεπτέμβριος 2021

Εφαρμογή σύγχρονων τεχνικών υπολογιστικής νοημοσύνης Cat Swarm Optimization (CSO) για την αποδοτική επίλυση του προβλήματος Curriculum based Course Timetabling.

Τσόγκα Παρθενία

Peny_tsoga@hotmail.com

Επιτροπή Επίβλεψης Πτυχιακής / Διπλωματικής Εργασίας

Επιβλέπων Καθηγητής: Γρηγόριος Μπεληγιάννης

Συν-Επιβλέπων Καθηγητής: Αλεφραγκής Παναγιώτης

Πάτρα, Σεπτέμβριος 2021

ΕΥΧΑΡΙΣΤΙΕΣ

Θα ήθελα να ευχαριστήσω τον επιβλέποντα καθηγητή μου κύριο Γρηγόριο Μπεληγιάννη, για την ανάθεση του θέματος και την εμπιστοσύνη που έδειξε στο πρόσωπό μου καθώς και για την βοήθεια και υποστήριξη που μου παρείχε καθ' όλη την διάρκεια της εκπόνησης της παρούσας εργασίας .

Αφιερώνεται στην οικογένεια μου.

Περίληψη

Ο τίτλος της παρούσας διπλωματικής εργασίας είναι «Εφαρμογή σύγχρονων τεχνικών υπολογιστικής νοημοσύνης Cat Swarm Optimization (CSO) για την αποδοτική επίλυση του προβλήματος Curriculum based Course Timetabling και σκοπός της είναι να σχεδιαστεί, υλοποιηθεί και αξιολογηθεί ένας αλγόριθμος Υπολογιστικής Νοημοσύνης ο οποίος θα παράγει ποιοτικά ωρολόγια προγράμματα. Ο αλγόριθμος που χρησιμοποιήθηκε είναι σμήνους και πιο συγκεκριμένα ο Cat Swarm Optimization όπως αυτός παραγοντοποιήθηκε για να παραχθεί το καλύτερο δυνατό αποτέλεσμα. Ο CSO προσαρμόστηκε στο πρόβλημα του Curriculum Based Course Timetabling και βελτιστοποιήθηκε περαιτέρω με διάφορες τεχνικές.

Υλοποιήθηκε σε περιβάλλον Matlab και εφαρμόστηκε πάνω σε δεδομένα που προέρχονται από το διεθνή διαγωνισμό International Timetabling Competition που διεξήχθη το 2007.

Η αξιολόγηση των ωρολογίων προγραμμάτων που παρήχθησαν πραγματοποιήθηκε χρησιμοποιώντας μετρήσιμα κριτήρια και ο αλγόριθμος CSO που προτάθηκε επιλύει το πρόβλημα και προσεγγίζει αρκετά τη βέλτιστη λύση, παράγοντας σε ικανοποιητικό χρονικό διάστημα ποιοτικά ωρολόγια προγράμματα, όπου υπερτερούν συγκριτικά με εκείνα που παράγουν άλλοι παρόμοιοι αλγόριθμοι.

Λέξεις κλειδιά: Τεχνητή νοημοσύνη, Υπολογιστική νοημοσύνη, Αλγόριθμοι, αλγόριθμος σμήνους, Cat swarm Optimization, πρόβλημα Timetabling, ωρολόγια προγράμματα τμημάτων Πανεπιστημίων

Περιεχόμενο: Κείμενο, Πίνακες, Εικόνες, screenshots, Βιβλιογραφική αναφορά

Summary

The title of this dissertation is "Application of modern computational intelligence techniques" Cat Swarm Optimization (CSO) to efficiently solve the Curriculum based Course Timetabling problem and its purpose is to be designed, to be implemented and to be evaluated a Computational Intelligence algorithm that will produce quality educational timetables. The algorithm used is swarms and more specifically the Cat Swarm Optimization as it was configured to produce the best possible result. The CSO was adapted to the Curriculum Based Course Timetabling problem and further optimized with various techniques.

It was implemented in Matlab environment and applied on data coming from the International Timetabling Competition held in 2007.

The evaluation of the program schedules produced was carried out using measurable criteria and the proposed CSO algorithm solves the problem and is quite close to the optimal solution, producing in a satisfactory period of time quality program programs, which are superior to those produced by other similar algorithms.

Keywords: Artificial Intelligence, Computational Intelligence, Algorithms, Swarm Algorithm, Cat Swarm Optimization, Timetabling Problem, University Department Schedules

Content: Text, Tables, Images, screenshots, Bibliographic reference.

Περιεχόμενα

1ο ΚΕΦΑΛΑΙΟ

1.1 Εισαγωγή

Ένα από τα σημαντικότερα προβλήματα που καλούνται να αντιμετωπίσουν οι μεγάλες επιχειρήσεις και οι οργανισμοί όσον αφορά το προσωπικό που εργάζεται σε αυτούς, και για να μπορούν να λειτουργούν ομαλά και αρμονικά, είναι η δημιουργία ενός χρονοδιαγράμματος στο οποίο θα κατανέμονται οι ώρες εργασίας του προσωπικού. Ο χρονοπρογραμματισμός αυτός ονομάζεται time scheduling και είναι αρκετά δύσκολο να πραγματοποιηθεί.

Η δημιουργία ενός ωρολογίου προγράμματος (timetable) είναι ένα αρκετά δύσκολο πρόβλημα, γιατί οι προϋποθέσεις και οι παράμετροι που πρέπει να λάβει υπ' όψιν, η κάθε εταιρεία ή οργανισμός είναι πολλές και τις περισσότερες φορές ανταγωνιστικές μεταξύ τους. Υπάρχουν για παράδειγμα αρκετοί οργανισμοί που λόγω των απαιτήσεων που έχουν θα πρέπει να λειτουργούν εικοσιτέσσερις ώρες το εικοσιτετράωρο και να απασχολούν προσωπικό σε διαφορετικές ειδικότητες και αρκετές φορές όχι ίσης και σταθερής διάρκειας σε ώρες. Επίσης το ωράριο του προσωπικού ενδέχεται να είναι διαφορετική ημέρα και ώρα μέσα στην εβδομάδα ή στο μήνα. Η διαθεσιμότητα του προσωπικού μπορεί να μην είναι

Οι παράμετροι του προβλήματος αυτού το καθιστούν αρκετά δύσκολο ώστε να κατασκευαστεί ωρολόγιο πρόγραμμα.

Ένα αντίστοιχο παράδειγμα ενός οριοθετημένου και θεσμοθετημένου οργανισμού που είναι απαραίτητη η ύπαρξη ωρολογίου προγράμματος είναι τα εκπαιδευτικά ιδρύματα. Στις περιπτώσεις αυτές θα πρέπει να κατανεμηθούν σε αίθουσες και σε ώρες διδασκαλίας τόσο οι φοιτητές όσο και οι εκπαιδευτικοί, ενώ θα πρέπει να ληφθούν υπόψιν οι διάφοροι περιορισμοί που υφίστανται από τους εκπαιδευτικούς, από τη χωρητικότητα των τάξεων, από τους φοιτητές ή και από την φύση των μαθημάτων.

Η κατασκευή αυτών των ωρολογίων προγραμμάτων κατά το παρελθόν είχε ανατεθεί σε κάποιους υπαλλήλους των εκπαιδευτικών ιδρυμάτων όπου είχαν επιβαρυνθεί με την εργασία αυτή από την διοίκηση του εκάστοτε ιδρύματος. Ωστόσο επειδή δεν είχαν στην

διάθεσή τους κάποιο λογισμικό, αναγκάζονταν να κάνουν το χρονοπρογραμματισμό με το χέρι.

Οι περιορισμοί που έπρεπε να λάβουν υπ' όψιν τους δυσκόλευε πολύ την εργασία τους, ο χρόνος που δαπανούσαν σ' αυτήν ήταν αρκετά μεγάλος και τα λάθη έκαναν συχνά την εμφάνιση τους. Αρκετές φορές μάλιστα το αποτέλεσμα δεν ικανοποιούσε το προσωπικό του οργανισμού, καθώς κάποιες από τις απαιτήσεις και τους περιορισμούς που είχαν θέσει δεν ικανοποιούνταν.

Επίσης, τα τελικά ωρολόγια προγράμματα έπρεπε να αλλάζουν και να τροποποιούνται κάθε φορά που υπήρχαν μεταβολές στο προσωπικό όπως για παράδειγμα άδειες ή μεταβολές στην λειτουργία του εκπαιδευτικού ιδρύματος γενικότερα.

Τα τελευταία χρόνια τα πράγματα άλλαξαν θεαματικά προς το καλύτερο με την βοήθεια της επιστήμης της Πληροφορικής και την αξιοποίηση λογισμικών. Σήμερα σχεδόν όλες οι επιχειρήσεις ή οργανισμοί, διαθέτουν έναν σύγχρονο υπολογιστή εξοπλισμένο με το αντίστοιχο λογισμικό. Το οποίο είναι σε θέση να κατασκευάσει σε αρκετά μικρό χρονικό διάστημα το ωρολόγιο πρόγραμμά του οργανισμού κατά σχεδόν βέλτιστο τρόπο, ακόμα και όταν οι απαιτήσεις και οι περιορισμοί που υπάρχουν είναι πολλοί.

Σήμερα υπάρχουν στη διάθεση των ενδιαφερομένων αρκετοί αποτελεσματικοί αλγόριθμοι επίλυσης του σχετικού προβλήματος, και έτσι είναι δυνατή η δημιουργία ποιοτικών ωρολογίων προγραμμάτων σε γρήγορο χρονικό διάστημα.

Πολλές επιστημονικές ομάδες σε παγκόσμιο επίπεδο εργάζονται για να βρούνε λύσεις σε αντίστοιχα προβλήματα και δημοσιεύουν επιστημονικά άρθρα με βελτιώσεις σε αλγόριθμους, ενώ αρκετές φορές προτείνουν και καινοτόμες λύσεις.

Υπάρχουν πολλά επιστημονικά περιοδικά και δημοσιεύσεις πάνω στο συγκεκριμένο πρόβλημα ενώ συγχρόνως διοργανώνονται διεθνή συνέδρια και παγκόσμιοι διαγωνισμοί σχετικοί με το θέμα αυτό.

Η σχεδίαση και η δημιουργία ενός ωρολογίου προγράμματος για μία επιχείρηση ή έναν οργανισμό εντάσσεται στα προβλήματα βελτιστοποίησης, αφού αντικείμενο τους είναι να βρεθούν βέλτιστες λύσεις, οι οποίες θα καλύπτουν τους περιορισμούς των προβλημάτων. Με αυτού του είδους τα προβλήματα ασχολείται η επιχειρησιακή έρευνα μια επιστήμη που

εφαρμόζει προηγμένες αναλυτικές μεθόδους για τη λήψη βέλτιστων αποφάσεων. Ένας άλλος κλάδος που ασχολείται με τέτοιου είδους προβλήματα είναι η Υπολογιστική Νοημοσύνη την οποία θα αναλύσουμε στο δεύτερο κεφάλαιο της παρούσας εργασίας.

1.2 Θεματικό Πεδίο Διπλωματικής

Ένα αρκετά δύσκολο προβλήματα που έχει να επιλύσει κάθε τμήμα πανεπιστημίου κατά την αρχή κάθε εξαμήνου είναι η δημιουργία του ωρολογίου προγράμματος. Θεωρείται δύσκολο γιατί πρέπει να ληφθούν υπόψιν αρκετοί περιορισμοί όπως είναι το πλήθος των αιθουσών και οι διαλέξεις που πρέπει να πραγματοποιηθούν σε αυτές. Άλλος ένας περιορισμός για παράδειγμα είναι η χωρητικότητα των αιθουσών. Υπάρχουν περιορισμοί που είναι αναγκαίο να τηρούνται, και κάποιο άλλοι που μπορεί τελικά να μην ικανοποιηθούν.

Είναι λοιπόν εμφανές ότι η σχεδίαση ενός ποιοτικού ωρολογίου προγράμματος είναι δύσκολη και απαιτεί αρκετό χρόνο. Ο υπάλληλος που είναι επιβαρυμένος με αυτή την δύσκολη εργασία έχει πλέον τη δυνατότητα να την ολοκληρώσει χωρίς να αντιμετωπίσει ιδιαίτερα προβλήματα χρησιμοποιώντας τον Ηλεκτρονικό Υπολογιστή μπορεί να απλοποιήσει αυτή τη διαδικασία, εφόσον βέβαια πάντα χρησιμοποιηθεί για την επίλυσης του προβλήματος κάποιος αλγόριθμος, ο οποίος θα έχει υλοποιηθεί έτσι ώστε να υπολογίζει και να καλύπτει τους περισσότερους περιορισμούς.

Σημαντικό επίσης είναι ότι η ποιότητα των δημιουργηθέντων ωρολογίων προγραμμάτων μπορεί να αξιολογηθεί, άρα υπάρχει η δυνατότητα σύγκρισης των αποτελεσμάτων των αλγόριθμων που χρησιμοποιούνται.

Στη βιβλιογραφία υπάρχουν αλγόριθμοι , οι οποίοι προσεγγίζουν τη βέλτιστη λύση του προβλήματος της κατασκευής ωρολογίων προγραμμάτων για τα πανεπιστήμια και τις πολυτεχνικές σχολές.

Είναι κατά κύριο λόγο αλγόριθμο Υπολογιστικής Νοημοσύνης, οι οποίοι έχουν την ιδιότητα να μεταβάλλουν την συμπεριφορά τους με βάση με τις πληροφορίες που είναι διαθέσιμες την δεδομένη στιγμή που εκτελούνται, και χρησιμοποιούν διάφορες τεχνικές για τη δημιουργία των αποτελεσμάτων τους.

Αντικείμενο της παρούσας Διπλωματικής Εργασίας είναι ο σχεδιασμός και η υλοποίηση ενός τέτοιου αλγόριθμου, ο οποίος βασίζεται στην λογική σμήνους και είναι ο Cat Swarm

Optimization (CSO) με τον οποίο θα επιλύσουμε το πρόβλημα της κατασκευής ωρολογίων προγραμμάτων για την τριτοβάθμια εκπαίδευση. Επίσης θα γίνει σύγκριση των αποτελεσμάτων του αλγορίθμου CSO με τα αντίστοιχα αποτελέσματα διαφόρων άλλων αλγόριθμων,

Η ποιοτική αξιολόγηση των αποτελεσμάτων η οποία θα αναλυθεί στο πέμπτο κεφάλαιο της διπλωματικής εργασίας γίνεται με απόλυτα μετρήσιμα κριτήρια και με βάση πραγματικά αρχεία εισόδου.

1.3 Δομή της Διπλωματικής Εργασίας

Η παρούσα διπλωματική εργασία με τίτλο «Εφαρμογή σύγχρονων τεχνικών υπολογιστικής νοημοσύνης Cat Swarm Optimization (CSO) για την αποδοτική επίλυση του προβλήματος Curriculum based Course Timetabling» αποτελείται από πέντε κεφάλαια στα οποία θα κάνουμε μια σύντομη αναφορά

Κεφάλαιο 1: Αναφέρεται στη δυσκολία που αντιμετωπίζουν οι διάφοροι οργανισμοί στο να δημιουργήσουν ωρολόγια προγράμματα για το προσωπικό τους και δίνεται έμφαση στα εκπαιδευτικά ιδρύματα και κυρίως στα Πανεπιστήμια. Επίσης περιγράφεται το θεματικό πεδίο της διπλωματικής εργασίας και η δομής της. Γίνεται μια σύντομη αναφορά στα πρόβλημα Timetabling και μια ιστορική αναδρομή στη διεθνής βιβλιογραφία για την επίλυση του από αλγορίθμους που έχουν δημιουργηθεί μέχρι και σήμερα.

Κεφάλαιο 2 : Περιγράφεται ο όρος της Τεχνητής Νοημοσύνης και καθώς και το πέρασμα στην Υπολογιστική νοημοσύνη. Γίνεται μια σύντομη περιγραφή των Ασαφών Συστημάτων, των Τεχνητών Νευρωνικών Δικτύων, του Εξελικτικού υπολογισμού, των Τεχνητών Ανοσοποιητικών Συστημάτων και της Νοημοσύνη Σμήνους. Αναλύεται ο ψευδοκώδικας του αλγορίθμου Cat swarm optimization και περιγράφονται τα βήματα για την εκτέλεση του κώδικα σε περιβάλλον Matlab.

Κεφάλαιο 3: Περιγράφεται η έννοια της πολυπλοκότητας και οι κλάσεις στις οποίες κατηγοριοποιείται ένα πρόβλημα. Πιο συγκεκριμένα αναλύονται η κλάση P, η κλάση NP και η κλάση NP-complete. Το πρόβλημα Timetabling είναι είναι NP-complete, έτσι στη συνέχεια του κεφαλαίου θα περιγραφεί λεπτομερώς και θα δοθεί έμφαση στους περιορισμούς του.

Κεφάλαιο 4 : Γίνεται η ανάλυση και η περιγραφή του προτεινόμενου αλγορίθμου όπως αυτός παραμετροποιήθηκε για το πρόβλημα Curriculum Based Course Timetabling. Έτσι, παρουσιάζεται η αναπαράσταση και η αρχικοποίηση του αλγορίθμου καθώς και η κωδικοποίηση των cats.. Επίσης, γίνεται αξιολόγηση του προτεινόμενου αλγόριθμου με βάση τα αποτελέσματα των ωρολόγιων προγραμμάτων που παρήγαγε συγκριτικά με αυτά άλλων αλγορίθμων που έχουν υλοποιηθεί για το ίδιο πρόβλημα.

Κεφάλαιο 5 : Αναλύεται η εξαγωγή των συμπερασμάτων της παρούσας διπλωματικής εργασίας και αναφέρονται κάποιες παρατηρήσεις. Στο κεφάλαιο αυτό εντάσσεται και η βιβλιογραφία που χρησιμοποιήθηκε όπου γίνεται αναφορά σε επιστημονικές δημοσιεύσεις, βιβλία και πηγές του διαδικτύου, τα οποία χρησιμοποιήθηκαν για την εκπόνηση της παρούσας διπλωματικής Διατριβής.

1.4 Το πρόβλημα του timetabling

Το πρόβλημα του timetabling, έχει απασχολήσει την επιστημονική κοινότητα εδώ και αρκετά χρόνια και έχουν βρεθεί πολλοί αλγόριθμοι που προσπαθούν να προσεγγίσουν τη βέλτιστη λύση. Το συναντάμε κυρίως στον ακαδημαϊκό χώρο και διακρίνεται στο University Course Timetabling, στο Exam Timetabling και στο School Timetabling,

1.4.1 School Timetabling

Το πρόβλημα school timetabling αφορά τη κατασκευή του ωρολόγιου προγράμματος των σχολείων και είναι αποδεδειγμένο ότι ανήκει στην κλάση των προβλημάτων NP – Complete. Στην αρχή της σχολικής χρονιάς στα σχολεία γίνεται η κατανομή των ωρών των μαθημάτων με βάση τα τμήματα και τις τάξεις των σχολείων καθώς και η κατανομή των ωρών ανά εκπαιδευτικό. Σημαντικό σε κάθε σχολείο είναι να συνυπολογιστεί και ο αριθμός των αιθουσών, αλλά και η χωρητικότητα της κάθε τάξης. Με βάση αυτά τα παραπάνω δεδομένα συντάσσεται το ωρολόγιο πρόγραμμα το οποίο θα πρέπει να αναδιαμορφώνεται συνεχώς μέσα στη σχολική χρονιά όπως για παράδειγμα λόγω των προσλήψεων των εκπαιδευτικών που συνεχίζονται κατά τη διάρκεια της σχολικής χρονιάς, των αδειών των εκπαιδευτικών που μπορεί να προκύψουν αλλά και άλλων καταστάσεων όπου ενδεχομένως να χρειαστεί να αναδιαμορφωθεί το πρόγραμμα. Άλλος ένας βασικός περιορισμός που θα πρέπει να ικανοποιείται είναι ότι ένας αριθμός εκπαιδευτικών διατίθεται και σε άλλα σχολεία κάποιες μέρες της εβδομάδας.

Με αποτέλεσμα η δημιουργία του ωρολόγιου προγράμματος να είναι μία από τις πιο δύσκολες διαδικασίες για την ομαλή λειτουργία του σχολείου.

Λόγω του ότι το ωρολόγιο πρόγραμμα αναδιαμορφώνεται αρκετές φορές κατά την διάρκεια της σχολικής χρονιάς και κάθε φορά πρέπει να αντιμετωπισθούν οι ίδιες δυσκολίες και περιορισμοί απαιτείται πολύς χρόνος και κόπος από τον εκπαιδευτικό που το έχει αναλάβει.

Πιο αναλυτικά ως οντότητες ορίζονται: οι εκπαιδευτικοί, τα τμήματα, τα μαθήματα και οι χρονικές περίοδοι. Δηλαδή, οι εκπαιδευτικοί πρέπει να διδάσκουν συγκεκριμένα μαθήματα, σε συγκεκριμένες τάξεις και σε συγκεκριμένες χρονικές περιόδους.

Θα πρέπει κατά την δημιουργία του προγράμματος να ικανοποιούνται όσον το δυνατό περισσότεροι περιορισμοί.

Οι περιορισμοί, διακρίνονται σε δύο κατηγορίες: τους ελαστικούς και τους ανελαστικούς περιορισμούς. Το ωρολόγιο πρόγραμμα χαρακτηρίζεται θεωρείται αποδεκτό, όταν ικανοποιούνται όλοι οι ανελαστικοί περιορισμοί.

Όσο το δυνατόν λιγότεροι ελαστικοί περιορισμοί παραβιάζονται στο ωρολόγιο πρόγραμμα τόσο πιο αποτελεσματικό θεωρείται. Άρα πρωταρχικός σκοπός είναι η κατασκευή εφικτών ωρολόγιων προγραμμάτων και δευτερεύον στόχος η δημιουργία ποιοτικών ωρολογίων προγραμμάτων.

1.4.2 Exam Timetabling Problem

Το πρόβλημα του Exam Timetabling είναι επίσης και αυτό ένα NP – Complete πρόβλημα το οποίο αφορά τον προγραμματισμό της εξεταστικής περιόδου των τμημάτων των πανεπιστημίων. Δηλαδή εξετάζονται οι παρακάτω περιορισμοί: ο συνολικός αριθμός των αιθουσών όπου πραγματοποιούνται οι εξετάσεις, ο συνολικός αριθμός των εξεταστέων μαθημάτων και τέλος, η συνολική προτίμηση κατανομής των εξετάσεων. Θα πρέπει οι εξετάσεις να κατανεμηθούν σε έναν περιορισμένο αριθμό χρονικών περιόδων και αιθουσών και να ικανοποιείται επιπλέον και μία σειρά από άλλους περιορισμούς. Οι περιορισμοί αυτοί είναι ελαστικοί και ανελαστικοί. Ανελαστικοί περιορισμοί για παράδειγμα είναι ο περιορισμός χωρητικότητας των αιθουσών και ότι οι φοιτητής δε μπορούν να δίνουν ταυτόχρονα δύο μαθήματα. Θα πρέπει απαραιτήτως να ικανοποιούνται όλοι οι ανελαστικοί περιορισμοί για να είναι αποδεκτό το ωρολόγιο πρόγραμμα.

Οι ελαστικοί περιορισμοί από την άλλη πλευρά δεν είναι απαραίτητο να ικανοποιούνται όλοι αλλά όσον το δυνατόν περισσότεροι ώστε να έχουμε ένα ποιοτικό ωρολόγιο πρόγραμμα.

Ένας ελαστικός περιορισμός είναι η σωστή κατανομή των μαθημάτων που δίνουν οι φοιτητές κατά την διάρκεια της εξεταστικής περιόδου

Έτσι για να έχουμε ένα αποδοτικό πρόγραμμα εξετάσεων θα πρέπει να καλύπτονται και να ικανοποιούνται όλοι οι ανελαστικοί περιορισμοί και όσοι περισσότεροι ελαστικοί περιορισμοί για να έχουμε και ποιοτικά προγράμματα.

1.4.3 University Course timetabling

Η επιστημονική κοινότητα όσον αφορά το πρόβλημα του timetabling έχει δώσει τη μεγαλύτερη έμφαση στο university Course Timetabling δηλαδή στη δημιουργία ωρολόγιων προγραμμάτων για τα πανεπιστήμια και τις πολυτεχνικές σχολές. Το συγκεκριμένο πρόβλημα αναφέρεται στον προγραμματισμό των διαλέξεων των μαθημάτων στη τριτοβάθμια εκπαίδευση σε ένα Πανεπιστήμιο ή ένα Πολυτεχνείο. Ορίζεται ως η ανάθεση συγκεκριμένων διαλέξεων σε έναν συγκεκριμένο αριθμό χρονοθυρίδων (Timeslots) και αιθουσών, με βάση τους ανελαστικούς και ελαστικούς περιορισμούς . Ως ελαστικοί περιορισμοί ορίζονται αυτοί που είναι υποχρεωτικοί και πρέπει οπωσδήποτε να ικανοποιούνται και ως ανελαστικοί ορίζονται αυτοί που δεν είναι απαραίτητο να ικανοποιούνται όλοι αλλά όσον το δυνατό περισσότεροι έτσι ώστε να παράγονται ποιοτικά προγράμματα.
Οι περιορισμοί ποικίλλουν από πανεπιστήμιο σε πανεπιστήμιο με βάση τις ανάγκες που έχει καθένα από αυτά με αποτέλεσμα να μην είναι όλοι ίδιοι.
Το πρόβλημα του University Course Timetabling διακρίνεται σε δυο κατηγορίες στο Curriculum based Course Timetabling και στο Post Enrollment Based Course Timetabling .
Το πρόβλημα του Post Enrollment Based Course Timetabling σχετίζεται με την δημιουργία ωρολόγιου προγράμματος με βάση την επιθυμία των φοιτητών για τη δήλωση διαλέξεων, δηλαδή οι φοιτητές δηλώνουν αρχικά τις διαλέξεις που θέλουν να παρακολουθήσουν και έπειτα παράγεται το πρόγραμμα στηριζόμενο σε αυτές.

Συμπερασματικά οι ανελαστικοί περιορισμοί του συγκεκριμένου προβλήματος αφορούν τις δηλώσεις των φοιτητών στις διαλέξεις όπου θα πρέπει υποχρεωτικά να μην υπάρχουν δυο ταυτόχρονα την ίδια χρονική περίοδο, τη χωρητικότητα των αιθουσών, τον προγραμματισμό

των διαλέξεων σε αίθουσες , τη διαθεσιμότητα προγραμματισμού διαλέξεων σε συγκεκριμένες περιόδους και τη σειρά με την οποία θα γίνεται η πραγματοποίηση των διαλέξεων.

Επιπλέον οι ελαστικοί περιορισμοί αφορούν την επιθυμία των φοιτητών και την κάλυψη των αναγκών τους όσον αφορά τις διαλέξεις. Για παράδειγμα θα επιθυμούσαν να μην έχουν κάποια διάλεξη την τελευταία περίοδο της ημέρας ή να μην έχουν συνεχόμενες διαλέξεις στη διάρκεια της ημέρας (Lewis κ.ά., 2007).

Για το προβλήματος Post Enrollment Based Course Timetabling έχουν χρησιμοποιηθεί αρκετοί αλγόριθμοι οι οποίοι δίνουν μια αποδοτική λύση όπως ο Simulated Annealing, ο Ant Colony Optimization, ο Tabu Search και άλλοι.

Το πρόβλημα Curriculum based Course Timetabling αφορά το πρόγραμμα σπουδών που βασίζεται στον προγραμματισμό των διαλέξεων των πανεπιστημιακών μαθημάτων ενός δεδομένου αριθμού αιθουσών και χρονικών περιόδων, όπου οι συγκρούσεις μεταξύ των μαθημάτων καθορίζονται σύμφωνα με τα προγράμματα σπουδών που δημοσιεύει το Πανεπιστήμιο και όχι βάσει των δεδομένων εγγραφής των φοιτητών. Το Curriculum based Course Timetabling θα αναλυθεί λεπτομερώς στα επόμενα κεφάλαια.

2ο ΚΕΦΑΛΑΙΟ

2.1 Τεχνητή νοημοσύνη

Ο όρος Τεχνητή Νοημοσύνη-ΤΝ (Artificial Intelligence-AI) διατυπώθηκε για πρώτη φορά σε ένα επιστημονικό συνέδριο ερευνητών που πραγματοποιήθηκε στο Dartmouth College.Το συνέδριο αφορούσε τους κλάδους των Μαθηματικών, της Ηλεκτρονικής και της Ψυχολογίας με θέμα τη μελέτη δυνατοτήτων χρήσης των υπολογιστών ως προς την προσομοίωση της ανθρώπινης νοημοσύνης . Στην πραγματικότητα όμως ως έννοια είχε ήδη κάνει την εμφάνισή της το 1950, σε μελέτη του Άλαν Τούρινγκ (1912-1954), όπου ο φημισμένος Άγγλος μαθηματικός έθετε το ερώτημα: *«Μπορούν οι μηχανές να σκεφτούν;»*. Το ερώτημα αυτό του Τούρινγκ ακόμη και σήμερα δεν μπορεί να απαντηθεί με βεβαιότητα.

Ιστορική αναδρομή της Τεχνητής Νοημοσύνης

- Το 1943 Οι McCulloch και Pitts προτείνουν ένα μοντέλο τεχνητών νευρώνων που έχει τη δυνατότητα να μαθαίνει και να υπολογίζει κάθε υπολογίσιμη συνάρτηση.
- Το 1950 Ο Alan Turing, που θεωρείται ο πατέρας της ΤΝ, εμπνέεται το τεστ της μίμησης (τεστ Τούρινγκ) για την αναγνώριση ευφυών μηχανών.
- Το 1951 Οι Minsky και Edmonts δημιουργούν το πρώτο υπολογιστή νευρωνικού δικτύου, το SNARC (*Stochastic Neural Analog Reinforcement Calculator*), το οποίο έχει 40 νευρώνες και χρησιμοποιεί 3000 λυχνίες.
- Το 1956 πραγματοποιείται συνάντηση ερευνητών στο Dartmouth College από το χώρο των Μαθηματικών, της Ηλεκτρονικής και Ψυχολογίας (McCarthy, Allen Newell, Herbert Simon, Marvin Minsky) με κοινό στόχο τη μελέτη δυνατοτήτων χρήσης των υπολογιστών για την προσομοίωση της ανθρώπινης νοημοσύνης.
- Το 1958 δημιουργείται η γλώσσας Lisp από τον McCarthy.
- Το 1959 ιδρύεται το εργαστήριο τεχνητής νοημοσύνης MIT AI Lab
- Το 1966 ο Weizenbaum δημιουργεί το ELIZA.
- Το 1972 δημιουργείται η γλώσσας λογικού προγραμματισμού PROLOG.
- Το 1977 δημιουργούνται τα πρώτα έμπειρα συστήματα: DENDRAL, MYCIN, Prospector .
- Το 1981 οι Ιάπωνες φτιάχνουν το πρόγραμμα 5ης γενιάς χρησιμοποιώντας τη γλώσσα PROLOG.

- Το 1986 οι Rumelhart and McClelland περιγράφουν τη δημιουργία προσομοιώσεων της αντίληψης στον υπολογιστή.
- Το 1987 πραγματοποιείται το 1ο Διεθνές Συνέδριο για τα Νευρωνικά Δίκτυα του IEEE (Institute of Electrical and Electronics Engineers).
- Το 1992 πραγματοποιείται το 1ο Συνέδριο του IEEE για τα Ασαφή Σύνολα.
- Το 1997 το Deep Blue της IBM κερδίζει τον παγκόσμιο πρωταθλητή σκακιού Gary Kasparov.
- Το 1999 η Sony δημιουργεί το πρώτο αυτόνομο κατοικίδιο το σκυλάκι AIBO.
- Το 2009 στο πανεπιστήμιο της Οξφόρδης δημιουργείται ο Adam, ο πρώτος επιστήμονας ρομπότ.

Έτσι σήμερα η Τεχνητή νοημοσύνη μελετά τη σχεδίαση και την υλοποίηση ευφυών πρακτόρων οι οποίοι μπορούν να εκτελούν πολύ γρήγορα πολλές εργασίες που για τον άνθρωπο απαιτείται πολύ κόπος και χρόνος.

Ένας ορισμός που θα μπορούσε να δοθεί για την Τεχνητή νοημοσύνη είναι ο ακόλουθος

«Τεχνητή Νοημοσύνη είναι η επιστήμη που επιδιώκει να κατασκευάσει μηχανές οι οποίες όχι μόνο θα επιδεικνύουν ανθρώπινη συμπεριφορά, αλλά θα μπορούν, επίσης, να προσαρμόζονται στο περιβάλλον τους με τρόπο παρόμοιο με αυτόν των ανθρώπων».

Η τεχνητή νοημοσύνη διακρίνεται σε δύο κλάδους:

- Την συμβολική Τεχνητή Νοημοσύνη (symbolic AI) η οποία βασίζεται στη κατανόηση των νοητικών διεργασιών και ασχολείται με την προσομοίωση της ανθρώπινης νοημοσύνης με αλγόριθμους και συστήματα που βασίζονται στη γνώση Χρησιμοποιώντας ως δοκιμές μονάδες τα σύμβολα (συλλογισμοί με βάση τα σύμβολα).
- Υπολογιστική νοημοσύνη βασίζεται στη μίμηση της βιολογικής λειτουργίας του εγκεφάλου όπως στην διαδικασία της εξέλιξης ή λειτουργία του εγκεφάλου. Όπως για παράδειγμα τα νευρωνικά δίκτυα και οι γενετικοί αλγόριθμοι.

2.2 Υπολογιστική Νοημοσύνη

Ο όρος υπολογιστική νοημοσύνη εισήχθη για πρώτη φορά στο Παγκόσμιο Συνέδριο World Congress computational intelligence to 1994. Από τη στιγμή που έκανε την εμφάνισή της η υπολογιστική νοημοσύνη έχουν δημοσιευτεί πολλά επιστημονικά άρθρα καθώς και πάρα πολλές εργασίες. Ωστόσο μέχρι και σήμερα δεν έχει δοθεί ένας ακριβής ορισμός. Ένας καλός ορισμός είναι ο ακόλουθος:

«Υπολογιστική Νοημοσύνη είναι ο επιστημονικός χώρος που προσφέρει τις τεχνικές για την επίλυση δύσκολων προβλημάτων, με τη μηχανή να μιμείται απλώς, βιολογικές διεργασίες, χωρίς να είναι απαραίτητο να επιδεικνύει γενική νοημοσύνη. ».

Ο Engelbrecht αναφέρει πως η υπολογιστική νοημοσύνη είναι ένας κλάδος της τεχνητής νοημοσύνης ο οποίος ασχολείται με την έρευνα και τη μελέτη προσαρμοστικών μηχανισμών οι οποίοι δίνουν έμφαση στην ευφυή συμπεριφορά σε περίπλοκα και δυναμικά περιβάλλοντα.

Τέτοιου είδους μηχανισμοί είναι η ικανότητα μάθησης ,η προσαρμογή σε νέες καταστάσεις Καθώς επίσης και η ικανότητα γενίκευσης, αφαίρεσης , ανακαλύψεις και συσχέτισης. Οι τεχνικές και μέθοδοι οι οποίες ανήκουν στον κλάδο της υπολογιστικής νοημοσύνης μπορούν να επιλύουν σύνθετα προβλήματα που μέχρι σήμερα ήταν πάρα πολύ δύσκολο να επιλυθούν. Όπως αναφέρει ο Engelbrecht οι μέθοδοι της υπολογιστικής νοημοσύνης διακρίνονται στις παρακάτω κατηγορίες:

- Ασαφής συστήματα
- Τεχνητά νευρωνικά δίκτυα
- Εξελικτικός υπολογισμός
- Τεχνητά ανοσοποιητικά συστήματα
- Νοημοσύνη σμήνους

2.2.1 Ασαφής Συστήματα

Ο όρος ασαφής λογικής (fuzzy logic) χρησιμοποιήθηκε για πρώτη φορά το 1965 από τον Lotfi Aliasker Zadeh, έναν καθηγητή του Πανεπιστημίου Μπέρκλεϋ στην Καλιφόρνια . Η ασαφής λογική βασίζεται στη θεωρία των ασαφών συνόλων (fuzzy set theory). Ο όρος

ασάφεια (fuzziness) στα μαθηματικά σημαίνει πολλαπλότητα τιμών. Στην ασαφή λογική μια πρόταση μπορεί να είναι αληθής μέχρι ένα βαθμό αληθείας. Η ασαφής λογική λέει ότι τα πράγματα δεν είναι μόνο άσπρο ή μαύρο, αλλά μπορούν να είναι και αποχρώσεις του γκρι . Έτσι ξεφύγαμε από το 0 ή 1 και το true ή false (Boolean λογική). Η ασαφής λογική δίνει τη δυνατότητα στα συστήματα να μπορούν σε συνθήκες αβεβαιότητας να λειτουργούν και να εξάγουν συμπεράσματα . Να μπορούν δηλαδή να δίνουν λύσεις σε προβλήματα τα οποία δεν είναι δυνατόν να αναπαρασταθούν με μαθηματικές εξισώσεις .Ο όρος αυτής της αβεβαιότητας διατυπώθηκε από τον Zadeh ως μια αρχή, την οποία ονόμασε "αρχή του ασυμβίβαστου". Η αρχή του ασυμβίβαστου λέει ότι: «Καθώς η πολυπλοκότητα ενός συστήματος αυξάνει, η ικανότητα μας να προβαίνουμε σε ακριβείς και σημαντικές δηλώσεις για τη συμπεριφορά του μειώνεται μέχρι που να φθάσουμε σε ένα όριο πέρα από το οποίο η ακρίβεια και η σημαντικότητα (ή σχετικότητα) καθίστανται σχεδόν αμοιβαίως αποκλειόμενα χαρακτηριστικά».

Εφαρμογές της ασαφούς λογικής συναντάμε σε πάρα πολλές εφαρμογές, όπως παρακάτω:

- Αυτόματα Κιβώτια Ταχυτήτων Αυτοκινήτων
- Ελεγκτές Ταξιδίου (GPS) Αυτοκινήτων και κινητών
- Ανελκυστήρες
- Αυτόματα Τρένα
- Βιολογικός Καθαρισμός Νερού
- Εναέρια Κυκλοφορία και πλοήγηση και πολλές άλλες εφαρμογές.

2.2.2 Τεχνητά Νευρωνικά δίκτυα

Τα νευρωνικά δίκτυα ANN (Artificial Neural Networks) είναι ένα σύνολο από πολύ ισχυρά μαθηματικά εργαλεία, τα οποία στηρίζονται στη λειτουργία του ανθρώπινου εγκεφάλου. Ο ανθρώπινος εγκέφαλος μπορεί να μαθαίνει από το περιβάλλον , να αποθηκεύει τις πληροφορίες στη μνήμη του και στη συνέχεια να τις χρησιμοποιεί σε νέες καταστάσεις. θα μπορούσαμε να πούμε ότι είναι ένα είδος μαύρου κουτιού αφού μπορεί να μοντελοποιήσει μη γραμμικά συστήματα χρησιμοποιώντας μόνο δεδομένα εισόδου και εξόδου. Ένας ορισμός που έχει δοθεί για τα νευρωνικά δίκτυα από τον Hyakin είναι ο παρακάτω:

Νευρωνικό δίκτυο ονομάζουμε έναν μαζικά κατανεμημένο παράλληλα επεξεργαστή, αποτελούμενο από απλές υπολογιστικές μονάδες που έχει την ικανότητα να αποθηκεύει εμπειρική γνώση και να την κάνει διαθέσιμη για χρήση. Μοιάζει με τον ανθρώπινο εγκέφαλο σε δυο σημεία:

1. *Η γνώση αποκτάται από το περιβάλλον του δικτύου μέσα από μια διαδικασία μάθησης.*
2. *Η «ισχύς» των συνάψεων ανάμεσα στους νευρώνες, γνωστή και ως συναπτικά βάρη, χρησιμοποιείται για να αποθηκευτεί η αποκτούμενη γνώση.*

Τα τεχνητά νευρωνικά δίκτυα κατασκευάζονται από μεμονωμένα κύτταρα του ανθρώπινου εγκεφάλου. Ο ανθρώπινος εγκέφαλος είναι ένας υπολογιστής που αποτελείται από δισεκατομμύρια νευρώνες διασυνδεδεμένους μεταξύ τους, θα μπορούσαμε να το χαρακτηρίσουμε ως ένα δίκτυο νευρώνων το οποίο διακρίνεται σε τρία επίπεδα. Παρακάτω απεικονίζεται ένα σχήμα επιπέδων αυτών:

Τα βέλη από τα δεξιά προς τα αριστερά δείχνουν την προώθηση της πληροφορίας προς το σύστημα ενώ τα βέλη από τα αριστερά προς τα δεξιά δείχνουν την ανατροφοδότηση του συστήματος. Τα τρία επίπεδα που διακρίνουμε είναι τα εξής

- **Νευρωνικό δίκτυο:** Ο εγκέφαλος που είναι το κεντρικό σύστημα αναπαριστάται ως ένα νευρωνικό δίκτυο που δέχεται πληροφορίες τις οποίες επεξεργάζεται και στη συνέχεια παίρνει αποφάσεις.
- **Δέκτες:** Οι οποίοι μετατρέπουν τα ερεθίσματα που δέχονται από το ανθρώπινο σώμα και το εξωτερικό περιβάλλον, σε ηλεκτρικούς παλμούς οι οποίοι στη συνέχεια μεταφέρουν τις πληροφορίες στο νευρωνικό δίκτυο.

- **Ενεργοποιητές:** Οι οποίοι είναι υπεύθυνοι για τη μετατροπή των ηλεκτρικών παλμών που παράγονται από το νευρωνικό δίκτυο σε αντιδράσεις και είναι η έξοδος του συστήματος.

Ο εγκέφαλος είναι η κεντρική μονάδα του νευρικού συστήματος ο οποίος αποτελείται από νευρωνικά δίκτυα. Τα νευρωνικά δίκτυα των ζωντανών οργανισμών είναι αυτά πού στηρίζουν εξειδικευμένες λειτουργίες όπως είναι η όραση η, μνήμη, η μάθηση κτλ. Κάθε νευρωνικό δίκτυο αποτελείται από νευρώνες οι οποίοι με τη σειρά τους αποτελούνται από τους δενδρίτες, το κυρίως σώμα καθώς επίσης και τον άξονα. Οι νευρώνες είναι μεμονωμένα κύτταρα τα οποία επεξεργάζονται μικρές πληροφορίες και έπειτα ενεργοποιούν άλλους νευρώνες για να συνεχίσουν αυτή τη διαδικασία.

Η μετάβαση της πληροφορίας γίνεται μέσω των δενδριτών και στα σημεία που τέμνονται δημιουργείται μία σύναψη(σύνδεση). Υπάρχουν περίπου 10 δισεκατομμύρια νευρώνες στον ανθρώπινο εγκέφαλο και 60 τρισεκατομμύρια συνδέσεις.

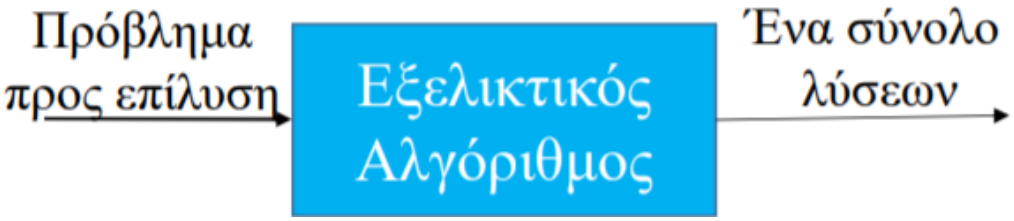
Τομείς που χρησιμοποιούνται τα νευρωνικά δίκτυα:

- Τα νευρωνικά δίκτυα μπορούν να χρησιμοποιηθούν στις τραπεζικές συναλλαγές. Χαρακτηριστικό είναι το πρόγραμμα Νέστωρ το οποίο καθορίζει την επικινδυνότητα μιας νέας αίτησης δανείου και αποφασίζει εάν θα πρέπει να εγκριθεί ή όχι το δάνειο, με ποσοστό επιτυχίας μεγαλύτερο συγκριτικά με άλλες μεθόδους.
- Στις τηλεπικοινωνίες όπου έχει δημιουργηθεί ένα φίλτρο το οποίο μειώνει τα λάθη κατά τη μετάδοση καθώς και τον εξωτερικό θόρυβο.
- Στα αεροδρόμια χημείας για τον έλεγχο αποσκευών.
- Στην ιατρική στην ανάγνωση και ανάλυση ακτινών Χ, για να προβλεφθούν αντιδράσεις του οργανισμού από τη λήψη φαρμακευτικών ουσιών.

2.2.3. Εξελεκτικός Υπολογισμος

Ο όρος του εξελικτικού υπολογισμού εμφανίστηκε για πρώτη φορά το 1950. Ο εξελικτικός υπολογισμός είναι ένα σύνολο αλγόριθμων και ο καθένας από αυτούς βασίζεται σε έναν πληθυσμό υποψήφιων λύσεων. Η κάθε υποψήφια λύση ονομάζεται άτομο (individual) (του πληθυσμού.) Το Βασικό χαρακτηριστικό του εξελικτικού υπολογισμού σε αντίθεση με τους κλασικούς αλγορίθμους είναι ότι η αναζήτηση γίνεται παράλληλα στο χώρο και οι

υποψήφιες λύσεις του πληθυσμού δοκιμάζονται ταυτόχρονα στο πρόβλημα δηλαδή η βελτιστοποίηση πραγματοποιείται στο χώρο αναζήτησης σε πολλά σημεία, ενώ στους κλασικούς αλγορίθμους η βελτιστοποίηση γίνεται σημείο προς σημείο στο χώρο αναζήτησης. Έτσι επιτυγχάνεται μια καλύτερη εξερεύνηση (exploration) και μια εκμετάλλευση (exploitation) του χώρου αναζήτησης ώστε να βρεθεί η βέλτιστη λύση. Σε κάθε μια επανάληψη που γίνεται μεταξύ των ατόμων του πληθυσμού εφαρμόζονται κατάλληλοι μηχανισμοί που ονομάζονται τελεστές (operators) οι οποίοι προέρχονται από τη Δαρβινική θεωρία της εξέλιξης έτσι ώστε να μπορούν να παράγονται οι καλύτερες δυνατές λύσεις για κάθε επανάληψη. Μετά από έναν αριθμό επαναλήψεων παρατηρείται πως ολόκληρος ο πληθυσμός τείνει προς μία βέλτιστη λύση. Οι τελεστές αποσκοπούν στην παραγωγή καλύτερων λύσεων αλλά και στην εξάλειψη των χειρότερων λύσεων .Σε κάθε επανάληψη που πραγματοποιείται υπολογίζεται ένας καινούριος πληθυσμός ατόμων ο οποίος περιέχει καλύτερες λύσεις έτσι ώστε να βρεθεί το ολικό βέλτιστο του προβλήματος. Ένα πρόβλημα που εμφανίζεται στους αλγορίθμους εξελικτικού υπολογισμού είναι η πρόωρη σύγκλιση (premature conevrgence). Το πρόβλημα αυτό δημιουργείται όταν ο αλγόριθμος προσεγγίζει αρκετά γρήγορα μία λύση, δηλαδή όταν οι λύσεις του πληθυσμού αποκτούν όλες την ίδια τιμή με αποτέλεσμα να μην μπορούν να δημιουργηθούν διαφορετικές λύσεις και να μην εξερευνάται όλος ο χώρος αναζήτησης των εφικτών λύσεων. Ένας τρόπος για να αποφύγουμε το πρόβλημα αυτό είναι να οριστεί η διαφορετικότητα των ατόμων του πληθυσμού και έτσι να επιτύχουμε την εφαρμογή των τελεστών. Ο εξελικτικός υπολογισμός συμπεριλαμβάνει μια κατηγορία αλγορίθμων βελτιστοποίησης οι οποίοι ονομάζονται Εξελικτικοί Αλγόριθμοι.



Χαρακτηριστικά εξελικτικών αλγορίθμων:

Οι εξελικτικοί αλγόριθμοι είναι :

- Είναι γενικοί και χρησιμοποιούνται σε πολλά προβλήματα.
- Εξελίσσουν τον πληθυσμό εφικτών λύσεων.

- Είναι στοχαστικοί γιατί χρησιμοποιούν πιθανοτικούς κανόνες και τελεστές.
- Χρησιμοποιούν τις αρχές της από τη βιολογικής εξέλιξης.
- Αξιοποιούνται για τις λύση στατικών και δυναμικών προβλημάτων.

Στατικά προβλήματα είναι τα προβλήματα στα οποία η βέλτιστη λύση δεν αλλάζει με το χρόνο ενώ δυναμικά είναι τα προβλήματα τα οποία η βέλτιστη λύση αλλάζει με το χρόνο.

Τα πλεονεκτήματα εξελικτικών αλγόριθμων είναι:

- Η απλότητα
- Αξιολογούνται μόνο οι λύσεις οι οποίες βρέθηκαν.
- Μπορούν να παράγουν και άλλες εναλλακτικές λύσεις.
- Ο υβριδισμός.
- Ο Παραλληλισμός.
- Έχουν εφαρμοστεί με επιτυχία στην πράξη.
- Έχουν υψηλή απόδοση.

Τα μειονεκτήματα εξελικτικών αλγόριθμων είναι:

- δεν εγγυώνται πάντα για την ποιοτική λύση.
- Το υπολογιστικό φορτίο.
- Η ανάγκη να ρυθμίζονται οι παράμετροι του αλγορίθμου.

Οι εξελικτικοί αλγόριθμοι διακρίνονται στις παρακάτω κατηγορίες:

- Στους γενετικούς αλγόριθμους οι οποίοι είναι οι πιο γνωστοί εξελικτικοί αλγόριθμοι και η μοντελοποίησή τους γίνεται με βάση τη γενετική εξέλιξη των διάφορων ειδών
- Στο γενετικό προγραμματισμό ο οποίος αποτελεί μία ειδική εφαρμογή των γενετικών αλγορίθμων στον οποίο τα άτομα είναι προγράμματα και η αναπαράσταση τους γίνεται με δέντρα.
- Στις Εξελικτικές Στρατηγικές οι οποίες προσανατολίζονται στη μοντελοποίηση των στρατηγικών παραμέτρων και ελέγχουν την μεταβολή της εξέλιξης.
- Στον εξελικτικός προγραμματισμό ο οποίος έχει τον παρόμοιο στόχο με τον γενετικό προγραμματισμό και τους γενετικούς αλγορίθμους. Η διαφορά του είναι ότι εστιάζει στην ανάπτυξη μοντέλων συμπεριφοράς και όχι σε γενετικά μοντέλα.

- Διαφορική εξέλιξη η οποία είναι σχεδόν ίδια με τους γενετικούς αλγορίθμους, αλλά έχει διαφορετικό μηχανισμό στην αναπαραγωγή.

2.2.4 Τεχνητά ανοσοποιητικά συστήματα

Το βιολογικό ανοσοποιητικό σύστημα το οποίο αποτελείται από εξειδικευμένους ιστούς, κύτταρα, όργανα και χημικά μόρια είναι ένα αρκετά πολύπλοκο δίκτυο και είναι υπεύθυνο για την άμυνα του οργανισμού. Έχει την ικανότητα να διακρίνει τα ξένα (αντιγόνα)από τα φυσιολογικά κύτταρα τα οποία εισέρχονται στον οργανισμό και η ιδιαιτερότητα αυτή ονομάζεται ανοσία. Όταν το φυσικό ανοσοποιητικό σύστημα έρχεται σε επαφή με ένα αντιγόνο η προσαρμοστική του φύση ενεργοποιεί και απομνημονεύει τη δομή του αντιγόνου για πιο αποτελεσματική απόκριση στο μέλλον. Το φυσικό ανοσοποιητικό σύστημα έχει τέσσερα μοντέλα:

- Το κλασικό μοντέλο όπου το κύρια χαρακτηριστικό του γνωρίσματα με βάση το κλασσικό μοντέλο του φυσικού ανοσοποιητικού συστήματος είναι η ικανότητά που έχει να ξεχωρίζει τα ξένα κύτταρα από τα φυσιολογικά κύτταρα με τη χρήση λεμφοκυττάρων που παράγονται στα λεμφοειδή όργανα.
- Η θεωρία επιλογής κλώνων το οποίο είναι ένα ενεργό Β-κύτταρο που παράγει αντισώματα και η εξάπλωση του γίνεται μέσω μιας διαδικασίας κλωνοποίησης. Οι κλώνοι που παράγονται περνούν και αυτοί μέσα από μία διαδικασία μετάλλαξης.
- Η Θεωρία Κινδύνου, στην οποία το ανοσοποιητικό σύστημα μπορεί να ξεχωρίζει τα επικίνδυνα από τα ακίνδυνα αντιγόνα.
- Η θεωρία Δικτύου όπου τα Β-κύτταρα δημιουργούν ένα δίκτυο κυττάρων και όταν ένα Β-κύτταρο του δικτύου αντιδράσει σε ένα αντιγόνο, ενεργοποιεί και διεγείρει όλα τα Β-κύτταρα που είναι συνδεδεμένο.

Έτσι το τεχνητό ανοσοποιητικό σύστημα (artificial immune system, AIS) μιμείται το φυσικό ανοσοποιητικό σύστημα καθώς και τις βασικές του λειτουργίες του οι οποίες αφορούν τους παθογόνους εισβολείς όπως:

- η αναγνώριση
- η καταστροφή
- η εκμάθηση και
- η μνήμη

Τα τεχνητά ανοσοποιητικά συστήματα εφαρμόζονται για την επίλυση των προβλημάτων αναγνώρισης προτύπων, στο τομέα της ασφάλειας στα πληροφοριακά συστήματα ,στην ανίχνευση σφαλμάτων και στη διάγνωσης βλαβών.

2.2.5 Νοημοσύνη Σμήνους

Η υπολογιστική νοημοσύνη σμήνους(swarm optimization intelligence SI) τα τελευταία χρόνια αναπτύσσεται δυναμικά. Αρχικά αναπτύχθηκαν οι τρεις παρακάτω αλγόριθμοι:

- Ο αλγόριθμος αναζήτησης στοχαστικής διάχυσης (Stochastic Diffusion Search (SDS), ο οποίος δημοσιεύθηκε το 1989 από τον Biston και είναι ο πρώτος μεταευρητικός αλγόριθμος που χρησιμοποιεί τον παράγοντα των πιθανοτήτων στη βελτιστοποίηση αναζήτησης.
- Ο Αλγόριθμος βελτιστοποίησης αποικίας μυρμηγκιών (Ant Colony Optimization ACO) Ο οποίος είναι εμπνευσμένος από τις ενέργειες μιας πραγματικής αποικίας μυρμηγκιών. Το πρωτότυπο ACO δημοσιεύθηκε το 1992 από τον Dorigo στη διδακτορική διατριβή του. Το κύριο χαρακτηριστικό του είναι μια τεχνική πιθανότητας βελτιστοποίησης για την εύρεση των καλύτερων διαδρομών στα γραφήματα
- Ο αλγόριθμος βελτιστοποίησης σμήνους σωματιδίων (PSO) που δημοσιεύθηκε το 1995 και είναι εμπνευσμένος από την κοινωνική συμπεριφορά οργανισμών όπως τα πουλιά (ένα σμήνος πουλιών) ή τα ψάρια (ένα κοπάδι ψαριών).

Η υπολογιστική νοημοσύνη σμήνους χρησιμοποιείται για τη μοντελοποίηση και την έξυπνη συλλογική συμπεριφορά των σμηνών στη φύση. Μια απλή συμπεριφορά είναι η παρατήρηση συγκεκριμένων πρακτόρων και η οργανωμένη αλληλεπίδραση μεταξύ τους φύση, π.χ. κοπάδια ψαριών, σμήνη πουλιών, αποικίες μυρμηγκιών. Αυτές οι συμπεριφορές είναι εμπνευσμένες για την ανάπτυξη «τεχνητών αποικιών πρακτόρων» που είναι σε θέση να επιλύσουν δύσκολα προβλημάτων βελτιστοποίησης. Η έννοια υπολογιστικού σμήνους νοημοσύνης μπορεί να περιγραφεί ως εξής:

«Η απλή συμπεριφορά των ατόμων η οποία είναι σχετικά απλή στη δομή της συν τις αλληλεπιδράσεις μεταξύ των ατόμων του σμήνους με την πάροδο του χρόνου είναι ίσο με μια πολύ περίπλοκη συλλογική συμπεριφορά »

Με βάση τους αλγορίθμους υπολογιστικής νοημοσύνης σμήνους, έχουν αναπτυχθεί πολλοί αλγόριθμοι μέχρι σήμερα. Οι αλγόριθμοι SI αντιπροσωπεύουν τεχνικές βελτιστοποίησης εμπνευσμένους από υποοικογένειες της φύσης .Πέρα από τις παραδοσιακές τεχνικές βελτιστοποίησης οι αλγόριθμοι SI έχουν ένα βασικό πλεονεκτήματα καθώς ξεκινούν με έναν πληθυσμό πιθανών λύσεων και όχι από ένα σημείο και οι λύσεις μπορούν να συνεργαστούν μεταξύ τους για να μοιραστούν γνώσεις.

Χαρακτηριστικά SI

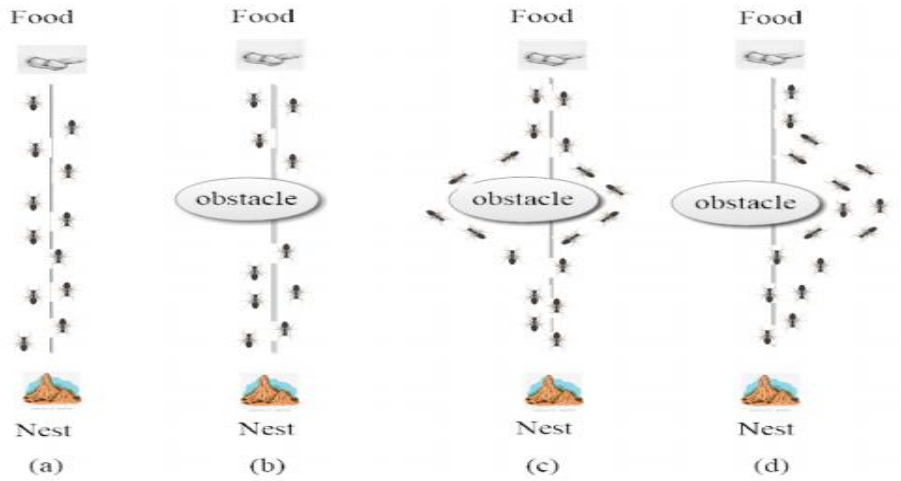
- Αποτελείται από πλήθος ατόμων.
- Τα άτομα είναι ομοιογενή, δηλαδή είναι όμοια ή διαφέρουν πολύ λίγο.
- Οι αλληλεπιδράσεις ανάμεσα στα άτομα στηρίζονται σε απλούς κανόνες συμπεριφοράς οι οποίες προέρχονται από τοπικές πληροφορίες που τα άτομα αποκτούν από μόνα τους ή από το περιβάλλον τους.
- Η συλλογική συμπεριφορά του συστήματος προέρχεται από τις αλληλεπιδράσεις μεταξύ των ατόμων αλλά και με του περιβάλλον τους , δηλαδή συμπεριφορά του συστήματος αυτοοργανώνεται.

Χαρακτηριστικό γνώρισμα ενός συστήματος ευφυΐας σμήνους είναι η δράση γίνεται με συντονισμένο τρόπο χωρίς τη παρουσία κεντρικού μηχανισμού συντονισμού.

Αλγόριθμοι υπολογιστικής νοημοσύνης σμήνους:

2.2.5.1 Αλγόριθμος αποικίας μυρμηγκιών (ant colony optimization ACO)

Τα μυρμήγκια είναι κοινωνικά πλάσματα που προτιμούν να ζουν σε ομάδες. Για την εύρεση της τροφής τους χρησιμοποιούν πάντα τη συντομότερη διαδρομή από τη φωλιά τους προς την πηγή τροφίμων. Στις αποικίες μυρμηγκιών, κάθε μυρμήγκι εκτελεί πολύ απλές ομαδικές ενέργειες χωρίς να γνωρίζει τι κάνουν τα άλλα μυρμήγκια. Ωστόσο, όλοι γνωρίζουν ότι το αποτέλεσμα είναι εξαιρετικά κοινωνικό και δομημένο. Μπορούν εύκολα να βρουν τη επόμενη συντομότερη διαδρομή του τελικού τους στόχου , ακόμη και αν υπάρχει ένα εμπόδιο. Αυτό το φαινόμενο παρουσιάζεται στο παρακάτω σχήμα:



α) τα μυρμήγκια ταξιδεύουν σε ευθεία γραμμή ώστε να ακολουθήσουν τη συντομότερη διαδρομή.

β) ένα εμπόδιο σπάει το ταξίδι τους και χωρίζει το μονοπάτι.

γ) τα μυρμήγκια αναζητούν ξανά την επόμενη πιο σύντομη διαδρομή

δ) τελικά, τα μυρμήγκια είναι σε θέση να εξασφαλίσουν τη συντομότερη διαδρομή.

Οι εθνολόγοι έχουν βρει ότι η ικανότητα αυτή των μυρμηγκιών οφείλεται σε ένα φαινόμενο που ονομάζεται μονοπάτι φερομόνης. Αυτό βοηθά τα μυρμήγκια να επικοινωνούν μεταξύ τους και μα ακλουθούν τον ίδιο δρόμο ή να αλλάξουν διαδρομή για την εύρεση τροφής. Αρχικά, τα μυρμήγκια εξερευνούν τυχαία την περιοχή γύρω από τη φωλιά τους για αναζήτηση τροφής. Αφού βρούνε τροφή επιστρέφουν στη φωλιά τους και μεταφέρουν κάποιο φαγητό. Κατά την επιστροφή αφήνουν μια οργανική ένωση στο έδαφος γνωστή ως φερομόνη και όσο μεγαλύτερη είναι η ποσότητα του φαγητού τόσο περισσότερη φερομόνη αφήνουν στο έδαφος. Έτσι και τα υπόλοιπα μυρμήγκια της φωλιάς ανιχνεύουν και ακολουθούν το μονοπάτι με τη πιο υψηλή φερομόνη.

Ο αλγόριθμος ACO είναι εμπνευσμένος από τη συμπεριφορά των μυρμηγκιών η οποία μοντελοποιήθηκε μαθηματικά με μία σειρά εξισώσεων για την εύρεση βέλτιστης λύσης σε συνδυαστικά προβλήματα βελτιστοποίησης και έχουν επιλυθεί αρκετά προβλήματα με τη χρήση αυτής της μεθόδου. Χρησιμοποιείται κυρίως για εύρεση βέλτιστων μονοπατιών σε γράφους. Ο πρώτος ο οποίος αναφέρθηκε σε αυτόν τον αλγόριθμο ήταν ο Marco Dorigo το 1992 μέσω της διατριβής του. Ο αλγόριθμος αυτός αποσκοπεί στην αναζήτηση μιας βέλτιστης διαδρομής σε ένα γράφο με βάση την συμπεριφορά των μυρμηγκιών όπου

αναζητούν μια διαδρομή από την φωλιά τους προς την τροφή. Η ιδέα αυτή διαφοροποιήθηκε και επεκτάθηκε έτσι ώστε να χρησιμοποιηθεί για την επίλυση μιας μεγαλύτερης κατηγορίας υπολογιστικών προβλημάτων τα οποία στηρίζονται στη συμπεριφορά των μυρμηγκιών. Στον αλγόριθμο CSO τα ψηφιακά μυρμήγκια είναι ένα γράφημα κόμβων και συνδέσεων που μιμείται το μονοπάτι από τη φωλιά προς τη τροφή των φυσικών μυρμηγκιών και η φερομόνη αντιστοιχεί σε διάφορες θέσεις με αριθμητικές π πληροφορίες όπου μας δίνει όλα τα πιθανά μονοπάτια του συστήματος Ο ACO μπορεί να βρίσκει τις πιο καλές λύσεις που δημιουργήθηκαν από προηγούμενες επαναλήψεις. Σε σε κάθε σύνδεση του γράφου αντιστοιχεί μια μεταβλητή (τεχνητή φερομόνη) και ανάλογα με το πόσο καλύτερη είναι η λύση, τόσο περισσότερη είναι η τεχνητή φερομόνη. Έτσι όταν ολοκληρωθεί θα έχουμε όλες τις πιθανές λύσεις και θα μπορούμε να δούμε την ποσότητα της τεχνητής φερομόνης έτσι ώστε να βρούμε τη βέλτιστη λύση.

Υπάρχουν όμως κάποια στοιχεία που δεν συναντούμε στα φυσικά μυρμήγκια , αλλά υπάρχουν στα ψηφιακά αυτά είναι: οι περισσότεροι αισθητήρες, η μνήμη, ο χρόνος και ο διακριτός χώρος, η απόθεση φερομόνης σε διαφορετικές χρονικές στιγμές, (αφού τα ψηφιακά μυρμήγκια αφήνουν φερομόνη όταν ολοκληρώσουν τις κινήσεις τους ενώ τα φυσικά την αφήνουν συνεχώς) και οι αλγόριθμοι για τη βελτίωση της απόδοσης του συστήματος.

2.2.5.2 Αλγόριθμος σμήνους Σωματιδίων (particle swarm optimization PSO)

Ο αλγόριθμος βελτιστοποίησης σμήνους σωματιδίων (PSO) εισήχθη το 1995 από τους James Kennedy και Russell Eberhart . Είναι και αυτός αλγόριθμος βελτιστοποίησης υπολογιστικής νοημοσύνης και αναπτύχθηκε από τη συμπεριφορά που παρουσιάζει ένα σμήνος έμβιων όντων όπως για παράδειγμα πτηνών ή ψαριών. Στον αλγόριθμο βελτιστοποίησης PSO υπάρχει ένα σύνολο από σωματίδια τα οποία αναζητούν για τη βέλτιστη λύση σε ένα πρόβλημα βελτιστοποίησης. Το κάθε σωματίδιο αποτελεί και αυτόνομα τη λύση στο πρόβλημα βελτιστοποίησης με βάσει τα ερεθίσματα τα οποία δέχεται από τα γειτονικά σωματίδια ή και από την ίδια την εμπειρία του έτσι επιλέγει τον τρόπο με τον οποίο θα κινηθεί στο χώρο. Ο συγκεκριμένος αλγόριθμος έχει την ικανότητα να θυμάται και να

μπορεί να συγκρατήσει λύσεις που έχει συναντήσει στο παρελθόν τις οποίες αξιοποιεί ξανά όταν εμφανιστεί παρόμοιο πρόβλημα.

Κατά την φάση της προετοιμασίας το κάθε σωματίδιο του συστήματος έχει μια τυχαία αρχική θέση και μια τυχαία ταχύτητα, και έτσι η αρχική του θέση είναι μια πιθανή λύση στο πρόβλημα και παίρνει μια τιμή που ορίζεται στη συνάρτηση του συστήματος. Το σωματίδιο κινείται σε ένα πολυδιάστατο χώρο που έχει οριστεί και κάθε φορά αποθηκεύει τη βέλτιστη λύση.

Στην πορεία μέσα από τα αποτελέσματα πειραμάτων προστέθηκαν και άλλες εκδοχές του αλγορίθμου όπως το global PSO(καθολικό) και το local PSO(τοπικό).

2.2.5.3 Αλγόριθμος αποικίας σμήνους μελλισσων (Artificial Bee Colony)

Η Τεχνητή Αποικία Μελισσών (ABC) είναι εξέλιξη του The bees Algorithm από τους Karaboga, Dervis, και Bahriye Basturk, όπου πρόσθεσαν άλλον ένα εξελικτικό αλγόριθμο βασισμένο στην λειτουργία σμήνους μελισσών.

Ο Αλγόριθμος ονομάστηκε (ABC) και έχει την ίδια συμπεριφορά με τον BA όμως η μεθοδολογία που ακολουθεί για να βρεθεί η βέλτιστη λύση είναι διαφορετική. Στην προσέγγιση του ABC υπάρχουν τρείς κατηγορίες μελισσών:

- Οι εργάτες (employed bees) είναι οι μέλισσες οι οποίες επισκέπτονται μια πηγή τροφής που την έχουν επισκεφθεί ξανά στο παρελθόν
- οι θεατές (onlookers) που είναι οι μέλισσες οι οποίες αναμένουν στη περιοχή χορού
- οι ανιχνευτές (scouts). που είναι οι μέλισσες οι οποίες ψάχνουν τυχαία τον χώρο για να βρούνε νέες λύσεις (πηγές τροφής).

. Θεωρούμε ότι μισό μέρος του πληθυσμού αποτελείται από εργάτες και το υπόλοιπο από θεατές. Για κάθε πηγή τροφής υπάρχει μόνο ένας εργάτης.

αριθμός πηγών = αριθμός εργατών

Η κάθε επανάληψη που πραγματοποιείται αποτελείται από τρία βήματα. Στη πρώτη φάση επιλέγονται τυχαίες πηγές στον χώρο και οι εργάτες ξεκινούν να συλλέξουν το νέκταρ

(απόδοση λύσης) το οποίο αξιολογείται κατά την επιστροφή τους στην φωλιά. Έπειτα, κάθε εργάτης επιστρέφει πίσω στην πηγή τροφής για να διαλέξει μια νέα γειτονική της προηγούμενης. Τέλος, στη τελευταία φάση οι θεατές διαλέγουν μία πηγή τροφής βάσει της απόδοσής της. Όσο μεγαλύτερη είναι η ποσότητα νέκταρ που προέρχεται από μία πηγή τροφής, τόσο μεγαλύτερη είναι και η πιθανότητα να την επιλέξουν οι θεατές. Συγκεκριμένα, στην αρχή, ο αλγόριθμος παράγει τον πληθυσμό των λύσεων που είναι κατανεμημένος τυχαία στον χώρο. Κάθε λύση είναι ένα διάνυσμα διαστάσεων δηλαδή ο αριθμός παραμέτρων του προβλήματος.

Μετά από την πραγματοποίηση αρκετών επαναλήψεων παράγονται νέες λύσεις που είναι είτε καλύτερες είτε χειρότερες. Οι εργάτες δεν θυμούνται τα αποτελέσματα των παλαιότερων λύσεων αλλά έχουν την ικανότητα να θυμούνται μόνο την καλύτερη που προέκυψε μέχρι εκείνη τη στιγμή.. Η επιλογή του σημείου τροφοληψίας αποφασίζεται βάσει μιας πιθανοτήτων.

2.2.5.4. Αλγόριθμος Σμήνους Γατών (Cat swarm optimization CSO)

Ο αλγόριθμος CSO σύμφωνα με τον Adam Slowik είναι εμπνευσμένος από τη συμπεριφορά των γατών. Ο αλγόριθμος χρησιμοποιείται για τη βελτιστοποίηση της αναζήτησης μιας λύσης σε έναν χώρο M διαστάσεων και αναπαρίσταται με μια κατάλληλη συνάρτηση. (fitness function).

Τα σύνολα λύσεων απεικονίζονται από γάτες. Η κάθε γάτα χαρακτηρίζεται:

- από έναν αριθμό διαστάσεων,
- ταχύτητες για την κάθε διάσταση
- μια σημαία(flag) που περιγράφει το mode της γάτας και το fitness value.Το mode της γάτας μπορεί να είναι εάν η γάτα βρίσκεται σε κατάσταση αναζήτησης(seeking mode) ή σε λειτουργία ανίχνευσης(tracing mode)

Η καλύτερη λύση δίνεται από τη γάτα που έχει το μεγαλύτερο fitness value.

Το seeking mode εκφράζει τη κατάσταση κατά την οποία η γάτα είναι σε θέση ανάπαυσης κοιτάζει γύρω και αναζητά την επόμενη θέση για να μετακινηθεί ενώ το tracing mode εκφράζει την κατάσταση στην οποία η γάτα ανιχνεύει στόχους.

Στη λογοτεχνία υπάρχουν πολλές τροποποιήσεις του αλγορίθμου CSO όπως:

- CSO for clustering ,
- parallel CSO
- Crazy CSO
- Multi-Objective Binary Cat Swarm Optimization (MOBCSO)
- Harmonious Cat Swarm Optimization (HCSO)
- Discrete Binary Cat Swarm Optimization (DBCSO)]
- Quantum Cat Swarm Optimization (QCSOC) αλγόριθμος

2.2.5.4.1 Αλγόριθμος CSO

Παρακάτω παρουσιάζεται ο ψευδοκώδικας του αλγορίθμου CSO. Ο αλγόριθμος λαμβάνει ως είσοδο τις ακόλουθες παραμέτρους:

- M: τον αριθμό των διαστάσεων του χώρου αναζήτησης.
- $[p_{i,j}^{min}, p_{i,j}^{max}]$ το εύρος μεταβλητότητας για θέσεις των γατών.
- MR (Mixture Ratio): Το ποσοστό που περιγράφει πόσες γάτες βρίσκονται σε seeking mode και πόσες γάτες βρίσκονται σε tracing mode.
- SMP: η αναζήτηση μνήμης.
- SRD: το εύρος αναζήτησης της επιλεγμένης διάστασης.
- CDC :η μέτρηση της διάστασης που πρέπει να αλλάξει.
- SPC: η θεώρηση της αυτοθέσης.
- N:ο συνολικός αριθμός των γατών.
- max: ο μέγιστος αριθμός επαναλήψεων
- c1: μια σταθερά που χρησιμοποιείται για την ενημέρωση της ταχύτητας των γατών όταν βρίσκονται σε κατάσταση seeking mode.
- Output Gbest: Η έξοδος αντιπροσωπεύεται από την καλύτερη θέση που μπορεί να επιτευχθεί από μια γάτα.

Στο παρακάτω αλγόριθμο CSO υποθέτουμε ότι η σημαία(flag) = 0, το SPC = 0 και το σημείο για μετάβαση στο seeking mode αντλείται τυχαία από τα αντίγραφα SMP, κάθε αντίγραφο έχει την ίδια πιθανότητα να επιλεχθεί.

Αρχικά (βήμα 1) δημιουργείται τυχαία ο αρχικός πληθυσμός N γατών. Κάθε γάτα αναπαρίσταται από μία διανυσματική τιμή M. Η μεταβλητή Gbest, ενημερώνεται λαμβάνοντας υπόψη την καλύτερη απόδοση τιμής των γατών (βήμα 3). Για όσο δεν ικανοποιείται το κριτήριο της συνθήκης while δηλαδή για όσο ο αριθμός των επαναλήψεων είναι μικρότερος από το μέγιστο αριθμό επαναλήψεων η διαδικασία θα επαναλαμβάνεται. Σύμφωνα με το MR ορισμένες γάτες θα είναι σε λειτουργία ανίχνευσης, ενώ οι υπόλοιπες θα βρίσκονται σε κατάσταση αναζήτησης (βήμα 5). Η μεταβλητή Gbest θα ενημερώνεται από την γάτα με την πιο υψηλή απόδοση (βήμα 6) και στη συνέχεια για κάθε γάτα η αποτίμηση των τιμών για τις νέες θέσεων λαμβάνεται υπόψη αν η γάτα είναι σε κατάσταση seeking mode ή tracing mode. Εάν η γάτα βρίσκεται σε κατάσταση seeking mode αναζήτησης, τότε δημιουργούνται αντίγραφα SMP, οι θέσεις των αντιγράφων ενημερώνονται λαμβάνοντας υπόψη την αξία του SRD και η νέα θέση μετακίνησης επιλέγεται τυχαία. Στη κατάσταση tracing mode ενημερώνεται πρώτα η ταχύτητα (βήμα 15) και η νέα θέση καθορίζεται χρησιμοποιώντας την ενημερωμένη ταχύτητα (βήμα 17). Τέλος, στο βήμα 21 ενημερώνεται η μεταβλητη Gbest όπου επιστρέφει το τελικό αποτέλεσμα του αλγόριθμου.

1. Δώσε τυχαία τυχαία τον πληθυσμό των N γατών X_i (i = 1, 2, ..., N) κάθε γάτα παίρνει μια τιμή M
2. Όρισε μία μεταβλητή Gbest στην οποία θα καταχωρηθεί η τιμή της καλύτερης απόδοσης
3. Καταχώρισε την καλύτερη X_i τιμή στο Gbest
4. **while** I< Imax **do**
5: θέσε τις γάτες σε tracing mode ή seeking mode σύμφωνα με τον MR
6: Υπολόγισε τις τιμές των γατών και ενημερώστε το Gbest
7: **for** i = 1: N **do**
8: **if** X_i είναι seeking mode **then**
9: δημιούργησε αντίγραφα SMP
10: ενημέρωσε τη θέση κάθε αντιγράφου χρησιμοποιώντας τον τύπο
11: $X_{cn} = X_c \times (1 \pm SRD \times R)$
12. επέλεξε τυχαία μια θέση για να μετακινηθεί από το σύνολο των αντιγράφων SMP
13: **else**
14: ενημέρωσε την ταχύτητα της γάτας χρησιμοποιώντας τον τύπο
15: $V_{i,d} = V_{i,d} + R \times C1 \times (X_{best,d} - X_{i,d})$

16: ενημέρωσε τη θέση της γάτας χρησιμοποιώντας τον τύπο

17: $X_{i,\,d,\,new} = X_{i,\,d,old} + V_{i,d}$

18**:** **end if**

19: **end for**

20: **end while**

21: επέστρεψε το Gbest ως αποτέλεσμα

Ο αλγόριθμος ξεκινά καθορίζοντας τις παραμέτρους και στη συνέχεια δημιουργούνται τυχαία οι αρχικές θέσεις και οι ταχύτητες των γατών. Οι γάτες βρίσκονται σε κατάσταση seeking mode ή tracing mode σύμφωνα με την τιμή της παραμέτρου που ονομάζεται MR (Mutation Ratio). Οι τιμές απόδοσης(fitness value) των γατών υπολογίζονται χρησιμοποιώντας μια αντικειμενική συνάρτηση και η καλύτερη λύση είναι η θέση της γάτας (X_{best}) με την καλύτερη απόδοση. Στη συνέχεια εφαρμόζεται η κατάσταση seeking mode και tracing mode και με βάση τις τιμές των flags και εάν πληρούνται τα κριτήρια τερματισμού τότε η διαδικασία τερματίζεται, διαφορετικά ακολουθούνται τα βήματα μετά την αρχικοποίησης του προβλήματος.

2.2.5.4.2 Κατάσταση seeking mode

Η κατάσταση seeking mode περιγράφει τη λειτουργία όταν η γάτα ξεκουράζεται. Όταν η γάτα ανιχνεύει έναν κίνδυνο τότε κινείται με προσοχή και με αργό ρυθμό. Στη κατάσταση αυτή η γάτα παρατηρεί τον M χώρο λύσεων για να αποφασίσει ποιά είναι η επόμενη κίνηση. Η γάτα γνωρίζει:

- το περιβάλλον της.
- τη δική της κατάσταση.
- τις επιλογές που μπορεί να κάνει.

Στον αλγόριθμο CSO αυτά τα γεγονότα αντιπροσωπεύονται από τις ακόλουθες παραμέτρους:

- αναζήτηση μνήμης (SMP).
- αναζήτηση εύρους επιλεγμένων διαστάσεων (SRD).
- πλήθος διαστάσεων που μεταβάλλονται (CDC) - ο αριθμός των διαστάσεων.
- αυτοθέση (SPC).

Η διαδικασία τρόπου αναζήτησης περιγράφεται στα επόμενα βήματα:

1. Για κάθε γάτα X_i δημιούργησε αντίγραφα SMP και εάν το flag SPC είναι αληθής τότε η τρέχουσα θέση της γάτας θεωρείται ως ένα από αυτά τα αντίγραφα SMP.
2. Σύμφωνα με το CDC υπολόγισε τη νέα θέση για κάθε γάτα χρησιμοποιώντας την παρακάτω εξίσωση:

$$\mathbf{X_{cn} = X_c \times (1 \pm SRD \times R)}$$

όπου το X_{cn} είναι η νέα θέση, το X_c είναι η τρέχουσα θέση και το R είναι ένα τυχαίος αριθμός από το διάστημα [0, 1]

3. για κάθε θέση υπολόγισε την απόδοση τιμής (fitness value) των νέων θέσεων και εάν όλες οι τιμές απόδοσης είναι ίσες, όρισε την πιθανότητα για όλα τα υποψήφια σημεία με 1, διαφορετικά χρησιμοποίησε την εξίσωση στο παρακάτω βήμα.
4. επέλεξε το σημείο για να μετακινηθείς τυχαία από το σύνολο των υποψηφίων πόντων και αντικαταστήστε τη θέση της γάτας X_i χρησιμοποιώντας τη ακόλουθη εξίσωση:

$$Pi = \frac{FSi - FSb}{FSmax - FSmin}$$

όπου 0 <i <j, P_i είναι η πιθανότητα της υποψήφιας γάτας X_i, FS_i είναι η απόδοσης τιμής (fitness value) των γατών X_i, FS_{max} είναι η μέγιστη απόδοση τιμής της συνάρτησης ,FS_{min} είναι η ελάχιστη απόδοση τιμής της συνάρτησης.FS_{max}=FS_b για προβλήματα μεγιστοποίησης και FS_b= FS_{min} για προβλήματα ελαχιστοποίησης.

2.2.5.4.3 Κατάσταση tracing mode

Η κατάσταση tracing mode προσομοιώνει το κυνήγι ενός θηράματος από μια γάτα. Οι εξισώσεις για τις ενημερώσεις ταχύτητας και θέσεων των γατών είναι η παρακάτω:

$$V_{k,d} = V_{k,d} + R \times c1 \times (X_{best,d} - X_{k,d})$$

και

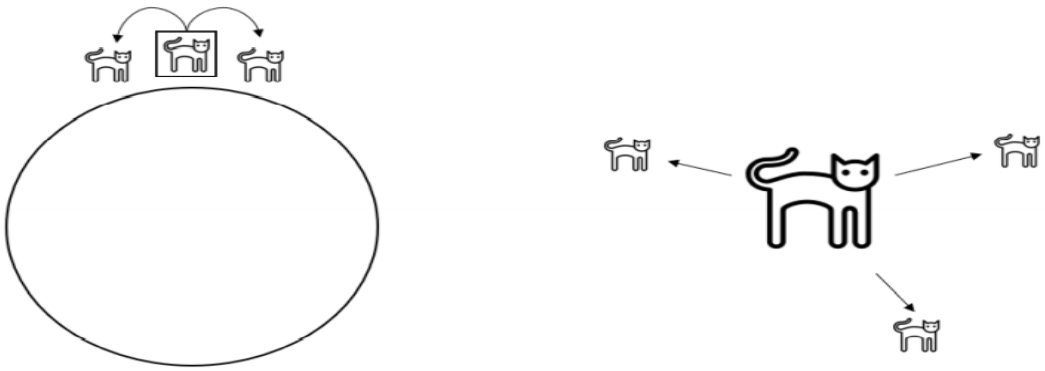
$$X_{k,d,new} = X_{k,d,old} + V_{k,d}$$

όπου το c1 είναι μια σταθερά της οποίας η βέλτιστη τιμή καθορίζεται λαμβάνοντας υπόψιν τις ιδιαιτερότητες του προβλήματος βελτιστοποίησης που θα λυθεί και δεν έχει ένα συγκεκριμένο εύρος μεταβλητότητας. Η τιμή του καθορίζεται στις περισσότερες περιπτώσεις που μέσο μιας διαδικασία δοκιμής και σφάλματος, το R είναι μια τυχαία τιμή από το διάστημα [0, 1], $V_{k,d}$ είναι η ταχύτητα της γάτας k για τη διάσταση d, $X_{k,d}$ είναι η θέση της γάτας k για τη διάσταση d, $X_{best,\ d}$ είναι η θέση της γάτας που έχει το τη καλύτερη λύση για τη διάσταση d, $X_{k,\ d,\ new}$ είναι η νέα τιμή της θέσης του cat k για τη διάσταση d και $X_{k,d,old}$ είναι η τρέχουσα θέση της γάτας k για τη διάσταση d.

2.5.4.4 Ο αλγόριθμος CSO σε matlab

Παρακάτω παρουσιάζεται ο πηγαίος κώδικας για την αντικειμενική συνάρτηση που θα βελτιστοποιηθεί από τον αλγόριθμο CSO. Στη συνάρτηση OF(P), η παράμετρος εισόδου είναι η θέση μιας γάτας C. Η αντικειμενική συνάρτηση δίνεται από τον τύπο, όπου:

$P_{i,\ j}$, $1 \le j \le M$, είναι η θέση της γάτας C_i για τη διάσταση j.



(a) (b)

Στο παραπάνω σχήμα απεικονίζεται

α) Τοπολογία δακτυλίου που λαμβάνει υπόψη την αριστερή και τη δεξιά γειτονική γάτα

β) τοπολογία αστέρα που απεικονίζει τη μητέρα γάτα στο κέντρο

$$OF(P) = \sum_{j=1}^{M} P_{i,j}^2 \qquad \text{where } -5.12 \leqslant P_{i,j} \leqslant 5.12$$

.

Ορισμός αντικειμενικής συνάρτησης OF(P) στο Matlab

```
1:  function [ out ]=OF(P)
2:  [ x , y ]= s i z e (P) ;
3:  out=zeros ( x , 1 ) ;
4:  for i =1:x
5     for j =1:y
6     out ( i , 1 )=out ( i , 1 )+P( i , j ) ^ 2 ;
7     end
8 end
```

Ορισμός shuffle συνάρτησης OF(P) στο matlab

Είναι μία συνάρτηση που χρησιμοποιείται για την τυχαία μετατροπή των αρχικών τιμών από τον πίνακα που δίνεται ως είσοδος.

```
1:  function [ out ]= shuffle ( x )
2:  out=x (randperm( length ( x ) ) ) ;
```

3: end

Κώδικα του CSO σε Matlab.

```
1: %−−− definition of the parameters of CSO algorithm
2: M=5; Pmin=zeros ( 1 ,M) ; Pmax=zeros ( 1 ,M) ; N=6; MAX=100 ; MR= 0 . 2 ;
3: SMP=5; SRD= 0 . 2 ; CDC=3; c1 = 0 . 5 ;
4: %−−− definition of the constraints value
5: Pmin ( 1 , : ) = −5.12; Pmax ( 1 , : ) = 5 . 1 2 ;
6: VMAX=10; VMIN=−10;
7: %−−− definition of arrays required by CSO algorithm
8: copies=zeros (SMP, M); numberOfCopies=SMP;
9: P=zeros (N,M) ; V=zeros (N,M) ;
10: Eva lPbest=zeros (N, 1 ) ; Gbest=zeros ( 1 ,M) ;
11: flag=zeros (N, 1 ) ; dimensionsToChange=zeros ( 1 ,M) ; Iter =0;
12: %−−− population of cats is randomly created
13: P=P+(Pmax ( 1 , : )−Pmin ( 1 , : ) ) . ∗ rand (N,M)+Pmin ( 1 , : ) ;
14: Pbest=P ;
15: %−−− population of cat evaluated
16: Eval=OF(P) ;
17: EvalPbest ( : , 1 )=Eva l ( : , 1 ) ;
18: %−−− seeking the best cat
19: [Y, I ]=min( Eval ( : , 1 ) ) ;
20: TheBest=I ;
21: EvalGbest=Y ;
22: %−−− t h e main l o o p o f t h e CSO a l g o r i t hm
23: while ( Iter <MAX)
24: Iter=I t e r +1;
25: catsInSeek in gMode=(1−MR)∗N ;
26: for i =1:N
27: if i < catsInSeekingMode
28: flag ( i , 1 ) =0;
29: else
30: flag ( i , 1 ) =1;
31: end
32: end
33: flag= shuffle ( flag ) ;
34: for i = 1:N
35: if flag ( i , 1 )==0
36 : dimensionsToChange ( 1 , : ) =0;
```

```
37:             dimensionsToChange ( 1 , 1 :CDC) =1;
38:             for c =1: numberOfCopies
39:                 copies ( c , : )=P( i , : ) ;
40:             end
41:             for  c =1: numberOfCopies
42:                 dimensionsToChange = shuffle ( dimensionsToChange ) ;
43:                 for j =1:M
44:                     if dimensionsToChange ( 1 , j )==1
45:                         if rand ( ) >0.5
46:                             copies ( c , j ) = copies ( c , j )+SRD∗ copies ( c , j ) ∗rand ( ) ;
47:                         else
48:                             copies ( c , j )= copies ( c , j )− SRD ∗ copies ( c , j ) ∗rand ( ) ;
49:                         end
50:                     end
51:                 end
52:             end
53:             selectedCopy = randi (SMP) ;
54:             for j =1:M
55:                 P( i , j )=copies ( selectedCopy , j ) ;
56:                 i f P( i , j )>Pmax( j ) P( i , j )=Pmax( j ) ; end
57:                 i f P( i , j )<Pmin ( j ) P( i , j )=Pmin ( j ) ; end
58:             end
59:         else
60:             for j =1:M
61:                 V( i , j )=V( i , j ) + rand ( ) ∗ c1 ∗(P( TheBest , j )−P( i , j ) ) ;
62 :                if  V( i , j )>VMAX     V( i , j )=VMAX; end
63:                 if  V( i , j )<VMAX     V( i , j )=VMIN ; end
64:                 P( i , j )=P( i , j )+V( i , j ) ;
65:                 if P( i , j )>Plax( j )      P( i , j )=Plax( j ) ; end
66:                 if P( i , j )<Pmin ( j )    P( i , j )=Pmin ( j ) ; end
67:             end
68:         end
69:     end
70:     TheBest =1;
71:     for  i =1:N
72:         Eval ( i , 1 )=OF(P( i , : ) ) ;
73:         if  Eval ( i , 1 )<EvalPbest ( i , 1 )
74:             Pbest ( i , : )=P( i , : ) ;
75:             EvalPbest ( i , 1 )=Eval ( i , 1 ) ;
76:         end
```

```
77:          if Eval ( i , 1 )<Eval ( TheBest , 1 )
78:            TheBest= i ;
79:          end
80:       end
81:       if Eval ( TheBest , 1 )<EvalGbest
82:          Gbest ( 1 , : )=P( TheBest , : ) ;
8:3          EvalGbest=Eval ( TheBest , 1 ) ;
84:       end
85: end
86: disp( EvalGbest ) ;
```

3ο Κεφάλαιο

3.1 Πολυπλοκότητα αλγορίθμου.

Η θεωρία της πολυπλοκότητας είναι η μελέτη των υπολογιστικών προβλημάτων και η κατάταξή τους σε κλάσεις σύμφωνα με τους υπολογιστικούς πόρους που χρησιμοποιούν και απαιτούνται για να επιλυθούν αλγοριθμικά. Αποδοτικός αλγόριθμος όσον αφορά τη χρονική διάρκεια εκτέλεσης πάντα είναι ο αλγόριθμος στον οποίο η χρονική του πολυπλοκότητα είναι πολυωνυμική.

Υπάρχουν προβλήματα όπως για παράδειγμα αυτό της εύρεσης ενός Hamiltonian κυκλώματος που μέχρι και σήμερα δεν έχει βρεθεί αποδοτικός αλγόριθμος. Οι αλγόριθμοι για το πρόβλημα αυτό που έχουν επινοηθεί μέχρι και τώρα δεν είναι αποδοτικοί διότι ο χρόνος εκτέλεσης τους είναι εκθετικός σε σχέση με το μέγεθος του γράφου εισόδου. Υπάρχουν αρκετά προβλήματα που φαίνεται να μην έχουν αποδοτική λύση.

Θα πρέπει στο σημείο αυτό να σημειωθεί ότι αναφορικά με τη πολυπλοκότητα του χρόνου δηλαδη πόσο χρόνο χρειάζεται για να τρέξει ένας αλγόριθμος θεωρούμε δύο μοντέλα στα οποία μπορούμε να στηριχθούμε και αναλύονται ως εξής:

- Τις **Ντετερμινιστικές μηχανές Turing** δηλαδή αλγορίθμους οι οποίοι λειτουργούν με ένα ακολουθησιακό τρόπο όπως για παράδειγμα τα προγράμματα C,Pascal κτλ και η διατύπωση ενός αλγορίθμου για μια ντετερμινιστική μηχανή Turing, διατυπώνει έναν ακολουθησιακό αλγόριθμο.
- Τις **μη Ντετερμινιστικές μηχανές Turing** αλγόριθμοι που θεωρούν ότι υπάρχει τέτοια μηχανή και μπορεί να χρησιμοποιηθεί για την επίλυση προβλημάτων.

Σημαντικό στη θεωρία της πολυπλοκότητας είναι η ταξινόμηση των προβλημάτων σε κλάσεις πολυπλοκότητας σύμφωνα με τη δυσκολία τους καθώς και η σχέση που υπάρχει μεταξύ των διαφορετικών κλάσεων πολυπλοκότητας.
Η σχέση αυτή φανερώνει βαθύτερες σχέσεις μεταξύ διαφορετικών μέσων υπολογισμού όπως για παράδειγμα τυχαίοι αλγόριθμοι σε σχέση με αιτιοκρατικούς αλγόριθμους. Ενώ συγχρόνως δείχνουν ένα βαθμό δυσκολίας των προβλημάτων που ανήκουν σε αυτές τις κλάσεις.

Οι πιο γνωστές κλάσεις είναι η κλάση ντετερμενιστικού πολυωνυμικού χρόνου P (Polynomial) και η κλάση μη ντετερμενιστικού πολυωνυμικού χρόνου NP (Non-deterministic Polynomial). Η κλάση P περιέχει όλα τα προβλήματα τα οποία μπορούν να επιλυθούν από έναν αιτιοκρατικό (deterministic) υπολογιστή σε πολυωνυμικό χρόνο.
Η δεύτερη κλάση περιέχει όλα εκείνα τα προβλήματα τα οποία μπορούν να λυθούν σε έναν μη ντετερμινιστικό (non-deterministic) υπολογιστή σε πολυωνυμικό χρόνο. Ένα τέτοιο παράδειγμα αποτελεί η εύρεση Hamiltonian κυκλωμάτων σε γραφήματα.
Ένα από τα πιο μεγάλα ανοικτά προβλήματα στην Επιστήμη των Υπολογιστών και των Μαθηματικά είναι αν P≠ N P.
Η πιο σημαντικότερη προσπάθεια η οποία έχει γίνει ως προς την κατεύθυνση επίλυσης αυτού του προβλήματος είναι η θεωρία NP-πληρότητας και γενικότερα η θεωρία ύπαρξης χαρακτηριστικών προβλημάτων για αρκετές κλάσεις πολυπλοκότητας.
Στο σημείο αυτό θα πρέπει να αναφερθεί ότι το σημαντικότερο εργαλείο για να αποδείξουμε ότι ένα πρόβλημα ανήκει σε μία κλάση πολυπλοκότητας είναι η αναγωγή.
Η αναγωγή αναφέρεται στην μετατροπή του στιγμιότυπου π1 ενός προβλήματος Π1 σε ένα στιγμιότυπο π2 του προβλήματος Π2 με τέτοιο τρόπο, ώστε η+ λύση στο π2 να δίνει την λύση και στο π1.

3.2 Κλάσεις πολυπλοκότητας

Η Κλάση P

Στην κλάση P συμπεριλαμβάνονται όλα εκείνα τα προβλήματα τα οποία μπορούν να επιλυθούν σε πολυωνυμικό χρόνο από Ντετερμινιστική μηχανή Turing. Βασικά υπάρχει μία σύνδεση της έννοιας και της αποδοτικότητας με την κλάση P.

Η σύνδεση αυτή στηρίζεται στην παρατήρηση ότι ένας πολυωνυμικός αλγόριθμος ενός προβλήματος βασίζεται σε κάποια βαθύτερη ιδιότητα του προβλήματος που του επιτρέπει τη γρήγορη επίλυσή του αποφεύγοντας έτσι την λύση της εξαντλητικής αναζήτησης (brute force) η οποία για τα τυπικά προβλήματα είναι εκθετικής πολυπλοκότητας. Για παράδειγμα, η εύρεση ενός ελάχιστου ζευγνύοντος δένδρου πραγματοποιείται σε πολυωνυμικό χρόνο, αφού μπορεί να εφαρμοστεί η μέθοδος της απληστίας. Υπάρχουν και κάποιες σπάνιες περιπτώσεις έως ανύπαρκτες μου μπορούμε να συναντήσουμε μη αποδοτικό αλγόριθμο ,με πολυπλοκότητα $O(n^{100})$ όπου δεν είναι αποδοτικός σε καμία περίπτωση. Η κλάση P ορίζεται ως εξής:

P= Σύνολο προβλημάτων που λύνονται σε πολυωνυμικό Ντετερμινιστικό χρόνο

Η Κλάση NP

Η κλάση πολυπλοκότητας NP αντιστοιχεί στα προβλήματα εκείνα για τα οποία υπάρχει πολυωνυμικός μη ντετερμινιστικός χρόνος, (non-deterministic) αλγόριθμος δηλαδή όλα τα προβλήματα που λύνονται από μη ντετερμινιστική μηχανή Turing και ομαδοποιούνται σε ένα σύνολο το οποίο ονομάζεται NP.

NP=Σύνολο προβλημάτων που λύνονται σε μη ντετερμινιστικό πολυωνυμικό χρόνο

Ένα πρόβλημα P υπάγεται στην κλάση NP αν έχει την ιδιότητα της πολυωνυμικής επαληθευσιμότητας. Για παράδειγμα, το πρόβλημα της ύπαρξης Hamiltonian κυκλώματος (HAMiltonian Circle - HAMC).

Συμπερασματικά μπορούμε να πούμε ότι η κλάση P περιέχει όλα εκείνα τα προβλήματα που λύνονται αποδοτικά, ενώ η κλάση NP περιέχει όλα εκείνα τα προβλήματα στα οποία η λύση μπορεί να επαληθευτεί αποδοτικά. Ένα μεγάλο πρόβλημα στην Επιστήμη των Υπολογιστών έχει να κάνει με το παρακάτω ερώτημα αν $P \neq NP$, αν δηλαδή η κλάση των προβλημάτων τα οποία λύνονται αποδοτικά συμπίπτει ή όχι με την κλάση των προβλημάτων των οποίων η λύση επαληθεύεται αποδοτικά. Είναι εμφανές ότι $P \subseteq NP$ αφού αν η λύση μπορεί να υπολογιστεί αποδοτικά, τότε μπορεί να επαληθευτεί και αποδοτικά από τον ίδιο τον αλγόριθμο ο οποίος βρίσκει τη λύση.

Υπάρχουν όμως και προβλήματα τα οποία δεν υπάγονται στην κλάση NP με αποτέλεσμα να μην επαληθεύονται αποδοτικά οι υποψήφιες λύσεις τους. Αυτού του είδους τα προβλήματα που κατά κάποιο τρόπο είναι συμπληρωματικά της κλάσης NP σχηματίζουν την κλάση NP-complete.

Η κλάση NP-complete

Η κλάση NPC (NP-Complete) είναι υποσύνολο της κλάσης NP και συμπεριλαμβάνει όλα τα NP-complete προβλήματα που είναι τα πιο δύσκολα της κλάσης NP. Για να θεωρηθεί ένα πρόβλημα NP-complete, πρέπει να ισχύουν τα παρακάτω:

1. $\Pi \in NP$

2. $\forall\ \Xi \in NP \Rightarrow \Xi \leq \Pi$

Ένα πρόβλημα Π για να θεωρηθεί NP-complete θα πρέπει να αποδειχθεί ότι ανήκει στην κλάση NP και αυτό επιτυγχάνεται κατασκευάζοντας έναν πολυωνυμικό επαληθευτή και έπειτα να αποδειχθεί ότι κάθε πρόβλημα στην κλάση NP ανάγεται στο Π. Κάτι που είναι αρκετά πολύπλοκο και για αυτό το λόγο αν αποδειχθεί ότι ισχύει για ένα πρόβλημα, τότε κάνοντας αναγωγές από αυτό θα αποδεικνύεται ότι και άλλα προβλήματα είναι N P-complete. Το θεώρημα των Cook-Levin δίνει το πρώτο τέτοιου είδους πρόβλημα που είναι NP-complete και είναι το SAT.Κάποια άλλα τέτοιου είδους προβλήματα είναι τα παρακάτω:

- SAT
- LONGEST PATH
- KNAPSACK
- INTEGER LINEAR PROGRAMMING
- TRAVELING SALESMAN PROBLEM
- RUDRATA PATH
- BALANCED CUT
- 3D MATCHING
- INDEPENDENT SET

Ένα ακόμα NP-complete πρόβλημα το οποίο μας απασχολεί και στη παρούσα εργασία, είναι το πρόβλημα του timetabling και συγκεκριμένα το πρόβλημα Curriculum based Course Timetabling το οποίο θα αναλυθεί στην επόμενη παράγραφο.

3.3 Curriculum based Course Timetabling

Όπως αναφέρθηκε και πιο πάνω το πρόβλημα Curriculum based Course Timetabling έχει να κάνει το πρόγραμμα σπουδών που βασίζεται στον προγραμματισμό των διαλέξεων των πανεπιστημιακών μαθημάτων ενός δεδομένου αριθμού αιθουσών και χρονικών περιόδων, όπου οι συγκρούσεις μεταξύ των μαθημάτων καθορίζονται σύμφωνα με τα προγράμματα σπουδών που δημοσιεύει το Πανεπιστήμιο και όχι βάσει των δεδομένων εγγραφής των φοιτητών.

Το πρόβλημα έχει απασχολήσει σε αρκετά μεγάλο βαθμό την επιστημονική κοινότητα και συγκεκριμένα το 2007 έλαβε χώρα ο Δεύτερος Διεθνής Διαγωνισμός (ITC 2007) ο οποίος αφορούσε το Timetabling. Ο Διαγωνισμός αποτελούντα από τρία σκέλη. Το πρώτο σκέλος εστίαζε στο Exam Timetabling, το δεύτερο σκέλος εστίαζε στο Post Enrollment Based Course Timetabling και το τρίτο σκέλος εστιάζει στο Curriculum based Course Timetabling.

Όπως έχει ήδη αναφερθεί και πιο πάνω το πρόβλημα Curriculum based Course Timetabling έχει να κάνει με την εύρεση του βέλτιστου ωρολογίου προγράμματος μαθημάτων για τμήματα πανεπιστημιακών και πολυτεχνικών σχολών με βάση το πρόγραμμα σπουδών των τμημάτων τους με βάση κάποιους περιορισμούς . Ενώ στο Post Enrollment Based Course Timetabling λαμβάνεται πρώτα υπόψιν η δήλωση μαθημάτων των φοιτητών και στη συνέχεια δημιουργείται το ωρολόγιο πρόγραμμα. Αναλυτικά θα περιγραφεί το πρόβλημα του Curriculum based Course Timetabling και θα αναλυθούν πο κάτω οι οντότητες όπως αυτές διατυπώθηκαν στο Δεύτερο Διεθνή Διαγωνισμό ITC-2007.

3.4Οντότητες του προβλήματος:

Οι οντότητες που συνιστούν το πρόβλημα του CB-CTT είναι οι εξής:

- Διδακτικές ημέρες (Days).Κάθε εβδομάδα αποτελείται από 5 έως 6 διδακτικές ημέρες ανάλογα πάντα με τη χώρα που βρίσκεται το κάθε πανεπιστήμιο για παράδειγμα στην Ελλάδα και στις περισσότερες χώρες οι διδακτικές μέρες ανέρχονται σε πέντε και είναι από Δευτέρα έως Παρασκευή.
- Χρονικά διαστήματα (Timeslots). Η κάθε διδακτική ημέρα χωρίζεται σε ίσα χρονικά διαστήματα τα οποία είναι ίδιες χρονικές στιγμές για κάθε διδακτική ημέρα.
- Περίοδοι (Periods). Ο ορισμός της περιόδου αντιστοιχεί σε ένα ζεύγος που αποτελείται από μια διδακτική ημέρα και ένα χρονικό διάστημα. Ο συνολικός αριθμός των περιόδων το σύνολο των διδακτικών ημερών επί το σύνολο των χρονικών διαστημάτων ανά ημέρα.
- Μαθήματα (Courses) και Καθηγητές (Teachers.). Κάθε μάθημα αποτελείται από ένα συγκεκριμένο αριθμό διαλέξεων οι οποίες προγραμματίζονται σε διαφορετικές περιόδους, παρακολουθείται από συγκεκριμένο αριθμό φοιτητών και διδάσκεται από ένα Καθηγητή. Επιπρόσθετα, για κάθε μάθημα υπάρχει ένας ελάχιστος αριθμός ημερών που μπορούν να πραγματοποιηθούν οι διαλέξεις και υπάρχουν και κάποιες

χρονικοί περίοδοι όπου δε μπορούν να πραγματοποιηθούν καθόλου οι διαλέξεις του συγκεκριμένου μαθήματος.

- Αίθουσες (Rooms). Κάθε αίθουσα έχει χωρητικότητα η οποία εκφράζεται με βάσει των αριθμό των θέσεων που έχει. Εδώ πρέπει να ληφθεί υπόψιν και το πόσο μεγάλη είναι η κάθε αίθουσα αν είναι εξίσου παρόμοιες όσον αφορά τις διαθέσιμες θέσεις.
- Προγράμματα Σπουδών (Curricula). Το curriculum είναι μια ομάδα μαθημάτων στην οποία κάθε ζεύγος μαθημάτων έχει κοινούς φοιτητές. Με βάση τα προγράμματα σπουδών υπάρχουν συγκρούσεις (Conflicts) μεταξύ των μαθημάτων.

Η λύση του προβλήματος είναι η ανάθεση μιας περιόδου (ημέρα και χρονική περίοδος) και μιας αίθουσας σε όλες τις διαλέξεις του κάθε μαθήματος.

3.5 Ανελαστικοί και ελαστικοί περιορισμοί

Όπως αναφέρθηκε και στο πρώτο κεφάλαιο τα προβλήματα αυτού του τύπου διέπονται από περιορισμούς οι οποίοι διακρίνονται σε ανελαστικούς και σε ελαστικούς. Οι ανελαστικοί περιορισμοί ή Hard Constraints είναι οι περιορισμοί που πρέπει να υποχρεωτικά να καλυφθούν ενώ οι ελαστικοί περιορισμοί ή soft Constraints είναι οι περιορισμοί εκείνοι που είναι επιθυμητοί να καλυφθούν αλλά όχι υποχρεωτικοί. Παρακάτω θα αναλυθούν οι περιορισμοί του προβλήματος Curriculum based Course Timetabling όπως αυτοί διατυπώθηκα στον διαγωνισμό ITC-2007.Επίσης θα πρέπει να επισημανθεί πως οι παραπάνω πειρασμοί δεν είναι πάντα σταθεροί και ίδιοι για όλα τα πανεπιστήμια, ανάλογα τις ανάγκες κάθε ιδρύματος αλλάζουν και τροποποιούνται.

Ανελαστικοί περιορισμοί ή Hard constraints:

- H1 Διαλέξεις (Lectures). Όλες οι διαλέξεις ενός μαθήματος πρέπει να προγραμματιστούν σε διαφορετικές χρονικές περιόδους, δεν γίνεται να συμπίπτουν τις ίδιες χρονικές στιγμές.
- H2 Χωρητικότητα Αιθουσών (RoomOccupancy). Δεν μπορούν να πραγματοποιηθούν δύο διαλέξεις ταυτόχρονα στη ίδια αίθουσα.
- H3 Συγκρούσεις (Conflicts). Τα μαθήματα τα οποία διδάσκονται από τον ίδιο καθηγητή πρέπει να διδάσκονται σε διαφορετικές χρονικές περιόδους. Είναι ένας σκληρός περιορισμός που δε πρέπει να παραβιαστεί καθώς ένας καθηγητής δε

μπορεί να βρίσκεται συγχρόνως σε δύο διαλέξεις οπότε θα πρέπει οπωσδήποτε να ικανοποιείται.

- Η4 Διαθεσιμότητα (Availability). Εάν ένας καθηγητής δεν μπορεί να διδάξει μια συγκεκριμένη χρονική περίοδο τότε δημιουργείται ένας περιορισμός διαθεσιμότητας όπου θα πρέπει να μην προγραμματιστεί καμία διάλεξη για την χρονική περίοδο που δεν είναι διαθέσιμος ο καθηγητής.

Ελαστικοί περιορισμοί (Soft Constraints)

Όπως αναφέρθηκε και προηγουμένως οι ελαστικοί περιορισμοί είναι εκείνοι οι οποίοι δεν είναι απαραίτητο να ικανοποιούνται όλοι αλλά όσον το δυνατόν περισσότεροι παρακάτω αναλύονται για το το πρόβλημα Curriculum based Course Timetabling.

- S1 Χωρητικότητα Αιθουσών (RoomCapacity). Για κάθε διάλεξη, ο αριθμός των φοιτητών που παρακολουθούν το μάθημα πρέπει να είναι μικρότερος ή ίσος από τον αριθμό των θέσεων της κάθε αίθουσας.
- S2 Ελάχιστος Αριθμός Ημερών (MinimumWorkingDays). Οι διαλέξεις κάθε μαθήματος πρέπει να κατανέμονται σε έναν ελάχιστο αριθμό ημερών.
- S3 Πιεσμένο Πρόγραμμα Σπουδών (CurriculumCompactness). Οι διαλέξεις που ανήκουν σε ένα πρόγραμμα σπουδών πρέπει να γειτνιάζουν μεταξύ τους (δηλαδή, σε διαδοχικές περιόδους). Για ένα δεδομένο πρόγραμμα σπουδών, θεωρούμε παραβίαση κάθε φορά που υπάρχει μια διάλεξη που δεν βρίσκεται δίπλα σε οποιαδήποτε άλλη διάλεξη εντός της ίδιας ημέρας.

- S4 Σταθερότητα αιθουσών (RoomStability). Όλες οι διαλέξεις ενός μαθήματος πρέπει να γίνονται όσο αυτό είναι εφικτό στην ίδια αίθουσα.

Όπως αναφέρθηκε στο κεφάλαιο δύο και με βάσει τους παραπάνω περιορισμούς το πρόβλημα του Curriculum based Course Timetabling ανήκει στα προβλήματα της κλάσης NP–Hard διότι ανήκει στο πρόβλημα χρωματισμού γράφου.(Graph Coloring).

3.6 Μαθηματική διατύπωση του προβλήματος πρόβλημα Curriculum based Course Timetabling.

Το πρόβλημα Curriculum based Course Timetabling. CB-CTT αποτελείται από ένα σύνολο n μαθημάτων C = {c1, c2,. . . , cn} που πρέπει να προγραμματιστεί λαμβάνοντας υπόψιν το σύνολο p περιόδων T = {t1, t2,. . . , tp} και το σύνολο m αιθουσών R = {r1, r2,. . . , rm}. Κάθε μάθημα ci, αποτελείται από li διαλέξεις που πρέπει να προγραμματιστούν. Για να είναι πιο απλό και για να μην υπάρχει μπέρδεμα δεν γίνεται διαχωρισμός μεταξύ του μαθήματος, της αίθουσας και του ονόματος της αίθουσας στο ακόλουθο περιεχόμενο. Η περίοδος είναι ένα ζεύγος που αποτελείται από μια διδακτική ημέρα και ένα χρονικό διάστημα, οι περίοδοι p κατανέμονται σε d ημέρες ανά εβδομάδα και h ημερήσια χρονικά διαστήματα (timeslots), δηλαδή p=d*h. Επιπρόσθετα υπάρχει ένα σύνολο s από προγράμματα σπουδών που ορίζεται ως CR = {cr1, cr2,. . . , crs}, όπου κάθε πρόγραμμα σπουδών Crk είναι μια ομάδα από μαθήματα που διαμοιράζονται σε κοινούς φοιτητές υποψήφια λύση αναπαρίσταται από ένα πίνακα X διαστάσεων p*m, όπου xi,j αντιστοιχεί στην ετικέτα του μαθήματος, που αντιστοιχεί στην περίοδο ti και στην αίθουσα rj. Εάν δεν έχει ανατεθεί μάθημα στην στην περίοδο ti και στην αίθουσα rj, τότε το xi,j παίρνει την τιμή «-1». Με αυτή την αναπαράσταση είναι βέβαιο ότι δε θα υπάρχουν περισσότερα από ένα μαθήματα που θα αντιστοιχούν σε κάθε αίθουσα σε οποιαδήποτε περίοδο. Με αυτόν τον τρόπο ο δεύτερος ανελαστικός περιορισμός (H2) θα ικανοποιείται πάντα.

Το συγκεκριμένο μαθηματικό μοντέλο του CB-CTT, παρουσιάζεται στην αναφορά των Lü & Hao (2010), και στον παρακάτω πίνακα παρουσιάζεται ένας αριθμός από σύμβολα και ορισμούς μεταβλητών, που χρησιμοποιήθηκαν για τη διατύπωση του μαθηματικού μοντέλου.

Σύμβολα	Περιγραφή
n	Συνολικός αριθμός μαθημάτων
m	Ο αριθμός αιθουσών
d	Αριθμός διδακτικών ημερών ανά εβδομάδα
h	Αριθμός χρονικών διαστημάτων ανά διδακτική ημέρα
p	Ο αριθμός περιόδων, p=d*h

s	ο αριθμός προγραμμάτων σπουδών (curricula)
C	Το σύνολο των μαθημάτων (|C| = n)
R	Το σύνολο των αιθουσών (|R| = m)
T	Το σύνολο των περιόδων (|T| = p)
CR	Το σύνολο των προγραμμάτων (curricula) (|CR| = s)
Crk	Το kth curriculum περιλαμβάνειi μια ομάδα μαθημάτων
li	Ο αριθμός των διαλέξεων του μαθήματος c_i
l	Συνολικός αριθμός διαλέξεων όλων των μαθημάτων ($l = \sum_1^n l\, i$
std_i	Ο αριθμός των φοιτητών που παρακολουθούν το μάθημα c_i
tc_i	Ο καθηγητής που διδάσκει το μάθημα ci
md_i	Ο αριθμός των ελαχίστων διδακτικών ημερών για τη κατανομή του μαθήματος ci
cap_j	Η χωρητικότητα της αίθουσας r_j
$uav_{i,j}$	Εάν το μάθημα c_i δεν είναι διαθέσιμο την περίοδο t_j, τότε το $uav_{i,j} = 1$, αλλιώς $uav_{i,j} = 0$
$Con_{i,j}$	Εάν το μάθημα c_i και c_j είναι σε σύγκρουση (conflict): $con_{ij} = \begin{cases} 0, & if(tCi \neq tCj) \wedge (\forall Crq, \quad ci\ Cr \wedge Cj \notin Crq), \\ 1, & διαφορετικά \end{cases}$
X_{ij}	Η τιμή του μαθήματος, που αντιστοιχεί στην περίοδο t_i και στην αίθουσα r_j
$nr_i(X)$	Ο αριθμός των αιθουσών που χρησιμοποιήθηκαν για τις διαλέξεις ενός μαθήματος ci σε μία υποψήφια λύση X: $\sigma_{ij}(\chi) = \begin{cases} 1, & if \forall Xkj \in X, \quad Xkj = Ci \\ 0, & διαφορετικά \end{cases}$
$nd_i(X)$	Ο αριθμός των διδακτικών ημερών των μαθημάτων Ci που λαμβάνει χώρα για κάθε υποψήφια λύση X. $nd_i(X) = \sum_{J=1}^{m} \beta ij(X)$,όπου

	$\beta_{ij}(\chi)=\begin{cases}1, & if\ \forall Xu,v\ \in X,\quad Xu,v = Ci \wedge \left[\frac{u}{h}\right] = j \\ 0, & διαφορετικά\end{cases}$
$App_{kj}(X)$	Εάν το curriculum crk εμφανίζεται σε μια περίοδο ti σε μια πιθανή λύση τότε : $app_{ij}(\chi)=\begin{cases}1, & if\ \forall Xi,j\ \in X,\quad Xi,j = Cu \wedge Cu\ \in Crk, \\ 0, & διαφορετικά\end{cases}$

Παρακάτω παρουσιάζεται το μαθηματικό μοντέλο για τους ανελαστικούς περιορισμούς H1, H2, H3 και H4

H1 Διαλέξεις (Lectures):

∀Ck ∈ C,

$$\sum_{i=1,\ldots\ldots pj=1,\ldots\ldots m} x\{X = C\,k\} = lk$$

όπου χ είναι αληθής δείκτης της συνάρτησης που λαμβάνει τιμές 1 εάν η δεδομένη πρόταση είναι αληθής και 0 εάν όχι.

H2 Χωρητικότητα Αιθουσών (Room occupancy) :

Ο συγκεκριμένος περιορισμός ικανοποιείται πάντα χρησιμοποιώντας την αναπαράσταση λύσεών.

H3 Συγκρούσεις (Conflicts):

$\forall x_{ij}, x_{ik} \in X, x_{i,j} = C_u, x_{i,k} = C_v$, $con_{uv} = 0$.

H4 Διαθεσιμότητα (Availability):

$\forall x_{ij} = C_k \in X$, $uav_{k,I} = 0$.

Παρακάτω παρουσιάζεται το μαθηματικό μοντέλο για τους ανελαστικούς περιορισμούς Η1, Η2, Η3 και Η4

S1 Χωρητικότητα Αιθουσών (RoomCapacity):

∀xij = Ck ∈ X,

$F_1(\chi_{i,j}) = \begin{cases} stdk - capj, & if\ stdk > capj \\ 0, & διαφορετικά \end{cases}$

S2 Ελάχιστος Αριθμός Ημερών (MinimumWorkingDays).

$F_3(C_{i,}) = \begin{cases} mdi - ndi(X), & if\ ndi(X) < mdi, \\ 0, & διαφορετικά \end{cases}$

S3 Πιεσμένο Πρόγραμμα Σπουδών

$F_3(C_{i,}) = \begin{cases} 1, & if\ (i\ mod\ h = 1 \vee appq, i-1(X) = 0) \\ \wedge (i\ mod\ h = 0 \vee appq, i+1(X) = 0) \\ 0, & διαφορετικά \end{cases}$

S4 Σταθερότητα αιθουσών (RoomStability).

F2(Ci) = nri(X) -1

Με την παραπάνω διατύπωση του προβλήματος, μπορεί στη συνέχεια να υπολογιστεί το συνολικό κόστος παραβιάσεων των ελαστικών περιορισμών για κάθε υποψήφια εφικτή λύση X σύμφωνα με τη συνάρτηση κόστους f όπως αυτή ορίζεται στον τύπο (1) . Ο στόχος είναι να βρεθεί μία εφικτή λύση X * έτσι ώστε f(X*)≤ f(X) για όλα τα X στον εφικτό χώρο αναζήτησης. Οπότε η Συνάρτηση f(x) θα είναι ίση με

$$\sum_{xij \in X} a1 * f1(xij) + \sum_{ci \in C} a2 * f2(ci) + \sum_{ci \in C} a3 * f3(ci) + \sum_{xij \in X} a4 * f4(xi,j),$$

Όπου α1,α2,α3 και α4 είναι τα κόστη των παραβιάσεων των ελαστικών περιορισμών. Για το πρόβλημα του CB-CTT έχουν οριστεί ως εξής: α1=1, α2=1, α3=5, α4 = 2.

3.7 Το ωρολόγιο πρόγραμμα στα ελληνικά πανεπιστήμια.

Το ωρολόγιο πρόγραμμα σε ένα ελληνικό Πανεπιστήμιο ή μια Πολυτεχνική Σχολή είναι πέντε ημέρες την εβδομάδα από Δευτέρα έως Παρασκευή και συνήθως τα μαθήματα πραγματοποιούνται από τις 09.00 έως τις 21.00. Επομένως αποτελείται από 12 διδακτικές ώρες ημερησίως. Σύμφωνα με τον Β. Σκουλλή στον παρακάτω πίνακα αναπαριστάται ένα ωρολόγιο πρόγραμμα με τις διδακτικές ώρες τις πέντε ημέρες της εβδομάδας. Η κάθε γραμμή του πίνακα αντιστοιχεί σε μια διδακτική ώρα του προγράμματος και η κάθε στήλη αντιστοιχεί στις ημέρες της εβδομάδας κατά τις οποίες πραγματοποιούνται διαλέξεις.

Διδακτικές ώρες	**Δευτέρα**	**Τρίτη**	**Τετάρτη**	**Πέμπτη**	**Παρασκευή**
08.00-09.00	1	13	25	37	49
09.00-10.00	2	14	26	38	50
10.00-11.00	3	15	27	39	51
11.00-12.00	4	16	28	40	52
13.00-14.00	5	17	29	41	53
14.00-15.00	6	18	30	42	54
15.00-16.00	7	19	31	43	55
16.00-17.00	8	20	32	44	56
17.00-18.00	9	21	33	45	57
18.00-19.00	10	22	34	46	58
19.00-20.00	11	23	35	47	59
20.00-21.00	12	24	36	48	60

Πίνακας αναπαράστασης ωρολόγιου προγράμματος

Σε κάθε κελί του πίνακα αντιστοιχεί ένα χρονικό διάστημα (timeslots). Σε κάθε ένα timeslot αντιστοιχεί ένας καθηγητής. Για παράδειγμα στη πρώτη γραμμή στη δεύτερη στήλη στον αριθμό 13 μπορούμε να αντιστοιχίσουμε έναν καθηγητή για παράδειγμα τον καθηγητή με τον αριθμό 2. Έτσι την πρώτη ώρα της Τρίτης δηλαδή 08.00-09.00 θα έχει μάθημα ο καθηγητής με τον κωδικό αριθμό 2.

Στη παρούσα εργασία οι χρονικές περίοδοι και ο αριθμός διδακτικών ωρών θα είναι οι ίδιοι που χρησιμοποιήθηκαν στο Διεθνή Διαγωνισμό Ωρολόγιου Προγράμματος (International Timetabling Competition)

4ο Κεφάλαιο

4.1 Ο αλγόριθμος Cat Swarm Optimization

Η διαδικασία του αλγόριθμου Cat Swarm Optimization απαιτεί το καθορισμό ορισμένων παραμέτρων οι οποίες εν γένει δύναται να επηρεάσουν τη σύγκλιση του δηλαδή τόσο τη λύση στην οποία θα καταλήξει ο αλγόριθμος όσο και το χρόνο που θα χρειαστεί για να καταλήξει στη λύση αυτή. Επιπρόσθετα ορισμένες από τις παραμέτρους αυτές επηρεάζουν και την απαίτηση μνήμης του αλγόριθμου (η οποία εξαρτάται επίσης και από την κωδικοποίηση της λύσης που θα ακολουθηθεί για κάθε πρόβλημα). Οι παράμετροι του αλγόριθμου παρουσιάζονται συγκεντρωτικά στο σημείο αυτό ωστόσο η ακριβής περιγραφή της επίδρασής τους στον αλγόριθμο γίνεται στις επόμενες παραγράφους όπου παρουσιάζεται αναλυτικά η διαδικασία του αλγόριθμου. Οι παράμετροι του αλγόριθμου ορίζονται στην αρχή της εκτέλεσης του και εν γένει παραμένουν σταθερές σε όλη τη διάρκεια της εκτέλεσής του. Οι παράμετροι αυτές είναι οι ακόλουθες:

- Το **μέγεθος του πληθυσμού** N_{cat} (δηλαδή το πλήθος των λύσεων ή στην ορολογία του αλγόριθμου το πλήθος των γατών – cat population). Το μέγεθος του πληθυσμού είναι φυσικός αριθμός (1,2,3,…). Θέτοντας μεγαλύτερη τιμή στο μέγεθος του πληθυσμού εν γένει αυξάνεται το εύρος του χώρου λύσεων που καλύπτουμε με τον αλγόριθμο αλλά ταυτόχρονα αυξάνεται το υπολογιστικό κόστος και ως προς την απαιτούμενη μνήμη και ως προς το χρόνο εκτέλεσης. Το μέγεθος του πληθυσμού εξαρτάται από το μέγεθος του χώρου του προβλήματος, δηλαδή τόσο από το πλήθος των διαστάσεων όσο και από το εύρος κάθε διάστασης.
- Ο **αριθμός των επαναλήψεων** $N_{maxiter}$ που θα εκτελεστούν. Το πλήθος των επαναλήψεων είναι φυσικός αριθμός (1,2,3,…). Η αύξηση του αριθμού των επαναλήψεων αυξάνει το χρόνο εκτέλεσης του αλγόριθμου αλλά συνήθως οδηγεί σε καλύτερες λύσεις.
- Ο **λόγος ανάμιξης** (**mixture ratio**) M_R ο οποίος καθορίζει το ποσοστό των λύσεων (γατών) που βρίσκονται σε καθεμία από τις δύο καταστάσεις (modes) που είναι η **αναζήτηση** (**seeking mode**) και η **ιχνηλάτηση** (**tracing mode**). Ο λόγος ανάμιξης μπορεί να εκφραστεί είτε ως ποσοστό επί τις εκατό είτε ως πραγματικός αριθμός στο διάστημα (0, 1). Οι δύο ακραίες τιμές (0% και 100% ή αντίστοιχα 0 και 1) δεν

συνίστανται διότι καταργούν τον έναν από τους δύο τρόπους διαδικασίας του αλγόριθμου.

- Το **μέγεθος της μνήμης αναζήτησης** (**seeking memory pool**) (SMP), το οποίο καθορίζει το πλήθος των αντιγράφων που χρησιμοποιούνται στη διαδικασία αναζήτησης. Το μέγεθος της μνήμης αναζήτησης είναι φυσικός αριθμός (1,2,3,…).
- Τη **συμπερίληψη της τρέχουσας θέσης** (**self position consideration**) (SPC) κατά την αναζήτηση. Η παράμετρος αυτή είναι λογική (δυαδική) μεταβλητή δηλαδή έχει τιμή αληθής ή ψευδής.
- Το **πλήθος των διαστάσεων που θα μεταβληθούν** (**counts of dimension to change**) (CDC), η οποία καθορίζει πόσες από τις διαστάσεις του προβλήματος θα μεταβληθούν κατά τη διάρκεια της αναζήτησης. Η παράμετρος είναι φυσικός αριθμός με μέγιστο των αριθμών των διαστάσεων του προβλήματος.
- Το **εύρος αναζήτησης** (**seeking range**) (SRD) που καθορίζει το διάστημα στο οποίο θα μεταβληθεί η κάθε επιλεγμένη διάσταση κατά την αναζήτηση. Το εύρος αναζήτησης μπορεί να εκφραστεί είτε ως ποσοστό επί τις εκατό είτε ως πραγματικός αριθμός στο διάστημα [0, 1].
- Το **εύρος ιχνηλάτησης** (**trace distance**) (TD) που καθορίζει το μέγιστο πλήθος των διαστάσεων που θα αλλάξουν κατά την ιχνηλάτηση. Το εύρος ιχνηλάτησης μπορεί να εκφραστεί είτε ως ποσοστό επί τις εκατό είτε ως πραγματικός αριθμός στο διάστημα [0, 1].

Ενίοτε ορίζονται βοηθητικές παράμετροι σχετικά με την καταγραφή της τρέχουσας καλύτερης λύσης και της συνολικά καλύτερης λύσης. Οι παράμετροι αυτές δεν επηρεάζουν τη σύγκλιση του αλγόριθμου, αλλά μόνο τον τρόπο παρουσίασης των αποτελεσμάτων του αλγόριθμου.

Πριν από την έναρξη του αλγόριθμου πρέπει να καθοριστεί ο τρόπος αναπαράστασης της λύσης. Ο τρόπος αναπαράστασης δεν είναι μοναδικός για κάθε πρόβλημα και μπορεί να επηρεάσει τη διαδικασία τους αλγόριθμου τόσο ως προς τη σύγκλιση όσο και ως προς το υπολογιστικό κόστος (απαίτηση μνήμης και χρόνο εκτέλεσης ανά επανάληψη). Ο σχεδιασμός μιας καλής αναπαράστασης της λύσης εξαρτάται κατά κύριο λόγο από την εμπειρία του προγραμματιστή του αλγόριθμου στη συγκεκριμένη κατηγορία προβλημάτων που προσπαθεί να επιλύσει. Πολλές φορές μία αναπαράσταση λύσης που δεν είναι καλά

σχεδιασμένη μπορεί να οδηγήσει σε αποτυχία του αλγόριθμου δηλαδή αδυναμίας του να βρει καλή λύση, ανεξάρτητα από τις τιμές των παραμέτρων του.

Ο τρόπος αναπαράστασης της λύσης επηρεάζει τον τρόπο υπολογισμού των αντικειμενικών συναρτήσεων (objective functions) και συνεπώς τον τρόπο υπολογισμού της συνάρτησης καταλληλότητας (fitness function) της κάθε λύσης. Παρότι ο τρόπος υπολογισμού των καταλληλότητας δεν επηρεάζει τη σύγκλιση του αλγόριθμου (η οποία επηρεάζεται μόνο από τις τιμές των συναρτήσεων αυτών και των αντίστοιχων τιμών της συνάρτησης καταλληλότητας), μπορεί να επηρεάσει την ταχύτητα εκτέλεσης του αλγόριθμου ή και τις απαιτήσεις μνήμης.
Η διαδικασία του αλγόριθμου Cat Swarm Optimization είναι η ακόλουθη:

1. Δημιουργείται ένα αριθμός πληθυσμός λύσεων μεγέθους N_{cat} ο οποίος παίρνει τυχαίες.
2. Υπολογίζεται η καταλληλότητα (fitness) κάθε λύσης μέσω του υπολογισμού των αντικειμενικών συναρτήσεων και αποθηκεύεται η βέλτιστη λύση καθώς και η τιμή της καταλληλότητας της.
3. Για κάθε λύση επιλέγεται τυχαία αν θα είναι στην κατάσταση αναζήτησης ή ιχνηλάτησης βάση της τιμής της παραμέτρου λόγου ανάμιξης (M_R).
4. Εκτελείται η διαδικασία της αντίστοιχης κατάστασης (αναζήτησης ή ιχνηλάτησης) όπως περιγράφεται στις ακόλουθες παραγράφους. Στο τέλος καθεμίας από τις καταστάσεις διαδικασίας υπολογίζεται η καταλληλότητα της προκύπτουσας λύσης,
5. Ενημερώνεται η βέλτιστη λύση και η τιμή της καταλληλότητάς της.
6. Ο αλγόριθμος επαναλαμβάνει τα βήματα 3 και 4 έως ότου εκτελεστεί ο προκαθορισμένος αριθμός επαναλήψεων ή ικανοποιηθεί κάποιο άλλο κριτήριο σύγκλισης.

Τα κριτήρια σύγκλισης που μπορεί να χρησιμοποιηθούν για τον τερματισμό του αλγόριθμου μπορεί να σχετίζονται με τον κορεσμό του πληθυσμού (saturation), ή την εύρεση λύσης με τη μέγιστη ή ελάχιστη τιμή της συνάρτησης καταλληλότητας.

Η ύπαρξη ακρότατου (μέγιστου ή ελάχιστου ανάλογα με τον ορισμό) της συνάρτησης καταλληλότητας εξαρτάται τόσο από το πρόβλημα όσο και από τον τρόπο ορισμού της συνάρτησης καταλληλότητας αυτή καθαυτής. Συνήθως ο αλγόριθμος Cat Swarm

Optimization χρησιμοποιείται για την επίλυση συνδυαστικών προβλημάτων με περιορισμούς (constrained combinatorial problems) στα οποία ως βέλτιστη λύση μπορεί να θεωρηθεί κάθε λύση που ικανοποιεί όλους τους περιορισμούς. Σε ένα τέτοιο πρόβλημα συνεπώς είναι γνωστή η τιμή της καταλληλότητας της βέλτιστης λύσης, συνεπώς, σε ένα τέτοιο πρόβλημα, μπορεί να χρησιμοποιηθεί ένα κριτήριο τερματισμού βασιζόμενη στην εύρεση του ακρότατου της συνάρτησης καταλληλότητας.

Ο κορεσμός του πληθυσμού (ή ισοδύναμα η ποικιλότητα του πληθυσμού) σχετίζεται με το πλήθος των διαφορετικών τιμών της συνάρτησης καταλληλότητας. Εν γένει στα συνδυαστικά προβλήματα δεν υπάρχει αμφιμονοσήμαντη αντιστοιχία ανάμεσα στις πιθανές λύσεις και τις τιμές της καταλληλότητας. Είναι πιθανό πολλές διαφορετικές λύσεις να έχουν την ίδια τιμή της συνάρτησης καταλληλότητας. Όταν κατά την εκτέλεση του αλγόριθμου πολλά μέλη του πληθυσμού έχουν την ίδια τιμή της συνάρτησης καταλληλότητας τότε πιθανότατα έχει βρεθεί ένα τοπικό ελάχιστο στο πρόβλημα και ο αλγόριθμος συχνά παγιδεύεται σε αυτό. Για το λόγο αυτό συχνά χρησιμοποιείται ένα κριτήριο τερματισμού το οποίο ελέγχει το ποσοστό των λύσεων που έχουν την ίδια τιμή της συνάρτησης καταλληλότητας.

Ενίοτε το κριτήριο κορεσμού δεν χρησιμοποιείται ως κριτήριο τερματισμού αλλά ως κριτήριο επανεκκίνησης δηλαδή δημιουργείται ένας καινούργιος πληθυσμός ένα ποσοστό του οποίου είναι οι λύσεις με τη συγκεκριμένη τιμή της καταλληλότητας και το υπόλοιπο μέρος του πληθυσμού είναι λύσεις μακριά από αυτές ώστε να αναζητηθεί λύση σε άλλες περιοχές του χώρου λύσεων και αν απεγκλωβιστεί ο αλγόριθμος από το τοπικό ακρότατο. Η εφαρμογή της επανεκκίνησης απαιτεί καλή γνώση του προβλήματος ώστε να γίνει με κατάλληλο τρόπο η επιλογή των λύσεων που είναι μακριά από τις κορεσμένες.

4.2 Η κατάσταση αναζήτησης

Ο αλγόριθμος της διαδικασίας σε κατάσταση αναζήτησης είναι ο ακόλουθος:

1. Δημιουργείται ένα πλήθος αντιγράφων της τρέχουσας λύσης. Το πλήθος των αντιγράφων είναι η τιμή της παραμέτρου μεγέθους μνήμης αναζήτησης (SMP).
2. Καθορίζεται εάν θα αλλάξουν οι τιμές σε όλα τα αντίγραφα ή εάν ένα αντίγραφο θα παραμείνει ως έχει, βάση της τιμής της παραμέτρου συμπερίληψης της τρέχουσας

θέσης (SPC). Εάν η τιμή της παραμέτρου είναι ψευδής αλλάζουν οι τιμές όλων των αντιγράφων ενώ αν είναι αληθής ένα αντίγραφο παραμένει ίδιο. Πρέπει να σημειωθεί ότι η βέλτιστη λύση διατηρείται ανεξάρτητα από τη διαδικασία του βήματος αυτού αφού η επιλογή της βέλτιστης λύσης γίνεται στο τέλος της διαδικασίας αναζήτησης.

3. Για καθένα από τα αντίγραφα που θα αλλάξουν (όπως καθορίστηκε από το προηγούμενο βήμα), υπολογίζεται το πλήθος των διαστάσεων που θα μεταβληθούν βάσει της αντίστοιχης παραμέτρου (CDC) και καθεμία από αυτές τις διαστάσεις αλλάζει σύμφωνα με την τιμή του εύρους αναζήτησης (SRD).
4. Υπολογίζονται οι τιμές των αντικειμενικών συναρτήσεων και της καταλληλότητας κάθε λύσης.
5. Βάσει των τιμών καταλληλότητας των λύσεων ορίζονται οι πιθανότητες επιλογής της κάθε λύσης για την εισαγωγή της στον πληθυσμό της επόμενης επανάληψης.
   5.1. Εάν η τιμή καταλληλότητας για όλες τις πιθανές λύσεις είναι ίδια τότε η πιθανότητα επιλογής είναι μονάδα (1).
   5.2. Εάν οι τιμές καταλληλότητας είναι διαφορετικές τότε η πιθανότητα επιλογής P_i της λύσης i ορίζεται ως:
      5.2.1. $P_i = \left|\frac{F_i - F_{max}}{F_{max} - F_{min}}\right|$ εάν ο αλγόριθμος αναζητά το ελάχιστο της συνάρτησης ή
      5.2.2. $P_i = \left|\frac{F_i - F_{min}}{F_{max} - F_{min}}\right|$ εάν ο αλγόριθμος αναζητά το μέγιστο της συνάρτησης

Στις παραπάνω σχέσεις F_i είναι η τιμή της καταλληλότητας της εξεταζόμενης λύσης, F_{max}, η μέγιστη τιμή καταλληλότητας των δημιουργηθέντων αντιγράφων, F_{min} η ελάχιστη τιμή καταλληλότητας των δημιουργηθέντων αντιγράφων.

Αξίζει να σημειωθεί ότι η τιμή της πιθανότητας επιλογής μίας λύσης που έχει τιμή καταλληλότητας ίση με του αναζητούμενου ακρότατου είναι μονάδα (1) και αν είναι αυτή του ανάποδου ακρότατου είναι μηδέν (0).

6. Βάσει των τιμών των πιθανοτήτων κάθε υποψήφια λύσης επιλέγεται τυχαία μία λύση για να εισαχθεί στον πληθυσμό των λύσεων της επόμενης επανάληψης.

4.3 Η κατάσταση ιχνηλάτησης

Ο αλγόριθμος της διαδικασίας σε κατάσταση ιχνηλάτησης είναι ο ακόλουθος:

1. Ενημερώνεται η ταχύτητα μεταβολής κάθε διάστασης (v_d) ανάλογα με την απόσταση της τρέχουσας λύσης από τη βέλτιστη και την τιμή του εύρους ιχνηλάτησης (TD). Οι ταχύτητες μεταβολής συνήθως έχουν μία μέγιστη τιμή οπότε εάν η υπολογισθείσα τιμή ξεπερνά τη μέγιστη τότε αντικαθίσταται από αυτήν.
2. Ενημερώνεται (τροποποιείται) η λύση ανάλογα με την ταχύτητα μεταβολής.

4.4 Ο αλγόριθμος Cat Swarm Optimization (ψευδοκώδικας)

Ο ψευδοκώδικας του αλγόριθμου Cat Swarm Optimization είναι ο ακόλουθος:

```
for each cat (solution)
                    initialize cat position, and velocity
end for
Do
                    for each cat (solution)
                        set cat into tracing or seeking mode randomly (according to
MR)
                        if cat is in seeking mode
                            apply the Seeking Mode process to the cat
                        else
                            apply the Tracing Mode process to the cat

                        calculate fitness value

                        if the fitness value is better than the best
                            set current value as the new best
                    end for
while maximum iterations not reached or other termination criteria are not met
```

4.5 Η κατάσταση αναζήτησης (ψευδοκώδικας)

Ο ψευδοκώδικας της κατάστασης αναζήτησης είναι ο ακόλουθος:

```
for each cat in seeking mode
                    if SPC is true
                        set N=(SMP−1) and keep the present position as a candidate
                    else
                        set N=SMP
                    make N copies of the present position of cat
                    for each copy of the cat
                        change CDC dimensions randomly by ± SRD
                        calculate the fitness values of all candidate solutions
                        if all FS are not exactly equal
                            calculate the selecting probability of each candidate solution
                        else
                            set selecting probability of each candidate solution = 1
                        randomly pick the solution to move to from the candidate solutions
                    end for
end for
```

4.6 Η κατάσταση ιχνηλάτησης (ψευδοκώδικας)

Ο ψευδοκώδικας της κατάστασης ιχνηλάτησης είναι ο ακόλουθος:

```
for each cat in tracing mode
```

```
                update the velocity for every dimension
                if any velocity is out of range
                    set it equal to the limit
                update the position of the cat
end for
```

4.7 Υβριδικός αλγόριθμος

Συχνά οι στοχαστικοί αλγόριθμοι εγκλωβίζονται σε τοπικά ακρότατα και χρησιμοποιούνται διάφορες τεχνικές για τον απεγκλωβισμό τους. Μια από τις πλέον συνηθισμένες τεχνικές είναι η χρήση υβριδικών αλγόριθμων.
Ένας υβριδικός αλγόριθμος χρησιμοποιεί έναν κύριο στοχαστικό αλγόριθμο για την αναζήτηση της λύσης και ανά διαστήματα χρησιμοποιεί έναν άλλο αλγόριθμο (είτε στοχαστικό είτε ντετερμινιστικό) για την εύρεση καλύτερης λύσης από αυτή που βρήκε ο κύριος αλγόριθμος.

Η ενεργοποίηση του δευτερεύοντος αλγόριθμου μπορεί να γίνεται είτε σε κάθε επανάληψη είναι ανά ορισμένο πλήθος επαναλήψεων είτε μέσω κάποιου άλλου κριτηρίου (πχ διασπορά των τιμών καταλληλότητας).
Στην παρούσα εργασία χρησιμοποιήθηκε ένας υβριδικός αλγόριθμος στον οποίο ο κύριος αλγόριθμος είναι ο Cat Swarm Optimization που παρουσιάστηκε παραπάνω και στο τέλος κάθε επανάληψης αυτού εκτελούταν έναν αλγόριθμος προσομοιωμένης ανόπτησης.

4.8 Αλγόριθμος προσομοιωμένης ανόπτησης

Ανόπτηση είναι η θέρμανση ενός μετάλλου σε μία θερμοκρασία τέτοια που να αρχίζει η ανακρυστάλλωσή του. Με την εφαρμογή ενός ελεγχόμενου ρυθμού ψύξης μπορούμε να έχουμε μέταλλο με την επιθυμητή κρυσταλλική δομή και το επιδιωκόμενο μέγεθος κόκκου. Η διεργασία της ανόπτησης παρότι καθορίζεται από ντετερμινιστικό κριτήριο

(ελαχιστοποίηση της ελεύθερης ενέργειας των κρυστάλλων) εμπεριέχει μεγάλο βαθμό στοχαστικότητας που σχετίζεται με τις σχετικές θέσεις των μορίων του υλικού.

Ο αλγόριθμος της προσομοιωμένης ανόπτησης (Simulated Annealing) προτάθηκε από τους Kirkpatrick, Gelatt και Vecchi, το 1983 και είναι εμπνευσμένος από τη φυσική διεργασία της ανόπτησης. Ο αλγόριθμος αυτός εν γένει εγκλωβίζεται δύσκολα σε τοπικά ακρότατα σε σχέση με άλλους αλγόριθμους. Αυτό επιτυγχάνεται μέσω μιας συνάρτησης αποδοχής των κακών λύσεων βάση μιας προκαθορισμένης πιθανότητας αποδοχής. Η πιθανότητα αποδοχής υπολογίζεται σε κάθε επανάληψη του αλγόριθμου με τέτοιο τρόπο ώστε στις αρχικές επαναλήψεις να είναι αρκετά μεγάλη, ενώ προς το τέλος του αλγορίθμου να είναι αρκετά μικρή.

Η χαρακτηριστική μεταβλητή του αλγόριθμου είναι η θερμοκρασία η οποία μεταβάλλεται κατά τη διάρκεια του αλγόριθμου σύμφωνα με ένα προκαθορισμένο σχήμα ψύξης. Σε κάθε επανάληψη του αλγόριθμου δημιουργούνται νέες πιθανές
λύσεις η απόσταση των οποίων από την τρέχουσα λύση είναι ανάλογη της τρέχουσας θερμοκρασίας. Μια λύση που είναι καλύτερη από την τρέχουσα επιλέγεται πάντα ως νέα λύση ενώ μια λύση που είναι χειρότερη από την τρέχουσα επιλέγεται βάσει της πιθανότητας αποδοχής που περιγράφηκε παραπάνω.
Πρέπει να σημειωθεί ότι στην παρούσα εργασία χρησιμοποιήθηκε η συνάρτηση του MATLAB ***simulannealbnd*** η οποία υλοποιεί τον συγκεκριμένο αλγόριθμο.

Ο αλγόριθμος προσομοιωμένης ανόπτησης εφαρμόζεται στο τέλος κάθε επανάληψης του αλγόριθμου Cat Swarm Optimization. Έτσι ο ψευδοκώδικας του συνολικού αλγόριθμου που χρησιμοποιείται στην εργασία είναι ο ακόλουθος:

```
for each cat (solution)
                    initialize cat position, and velocity
end for
Do
                    for each cat (solution)
                        set cat into tracing or seeking mode randomly (according to
MR)
```

```
                    if cat is in seeking mode
                        apply the Seeking Mode process to the cat
                    else
                        apply the Tracing Mode process to the cat

                    calculate fitness value

                    if the fitness value is better than the best
                        set current value as the new best
                end for

                apply simulated annealing procedure using best solution as starting point

while maximum iterations not reached or other termination criteria are not met
```

4.9 Υλοποίηση του αλγόριθμου

Στην παρούσα εργασία επιχειρήθηκε η επίλυση του προβλήματος του ωρολόγιου προγράμματος βασιζόμενου σε προγράμματα σπουδών (curriculum based course timetable) με τρεις διαφορετικές υλοποιήσεις του κώδικα.

- Στην πρώτη υλοποίηση χρησιμοποιήθηκε μόνο ο αλγόριθμος Cat Swarm Optimization.
- Στη δεύτερη υλοποίηση χρησιμοποιήθηκε ο υβριδικός αλγόριθμος Cat Swarm Optimization - Προσομοιωμένης Ανόπτησης με την υλοποίηση των αντικειμενικών συναρτήσεων που χρησιμοποιήθηκε στην πρώτη υλοποίηση και δημιουργία του αρχικού πληθυσμού με ίδιο τρόπο όπως στην πρώτη υλοποίηση.
- Στην τρίτη υλοποίηση χρησιμοποιήθηκε ο υβριδικός αλγόριθμος Cat Swarm Optimization - Προσομοιωμένης Ανόπτησης ωστόσο άλλαξε τόσο ο τρόπος υπολογισμού των αντικειμενικών συναρτήσεων όσο και ο τρόπος δημιουργίας του αρχικού πληθυσμού.

Οι αλγόριθμοι αρχικοποίησης του πληθυσμού των λύσεων και υπολογισμού των αντικειμενικών συναρτήσεων σε καθεμία από τις υλοποιήσεις παρουσιάζονται στις ακόλουθες υποενότητες.

4.10 Αναπαράσταση της λύσης

Η αναπαράσταση της λύσης είναι η ίδια και στις τρεις υλοποιήσεις της επίλυσης του προβλήματος. Κάθε λύση του προβλήματος αναπαρίσταται ως ένας τρισδιάστατος πίνακας διαστάσεων (DxTxR) όπου D ο αριθμός των ημερών, T ο αριθμός των ωρών διδασκαλίας (αριθμός περιόδων) ανά ημέρα, R ο αριθμός των διαθέσιμων αιθουσών. Οι παράμετροι αυτές καθορίζονται σε κάθε αρχείο δεδομένων του προβλήματος. Κάθε στοιχείο του πίνακα παίρνει τιμές φυσικών αριθμών στο διάστημα [0, C], όπου C ο μέγιστος αριθμός μαθημάτων (courses) που διδάσκονται. Η τιμή μηδέν (0) σε ένα στοιχείο του πίνακα δείχνει ότι στην αντίστοιχη μέρα και ώρα (περίοδο) στην αντίστοιχη αίθουσα δεν διδάσκεται κάποιο μάθημα

ενώ οποιαδήποτε τιμή από 1 έως C δείχνει ότι στην αντίστοιχη μέρα και ώρα (περίοδο) στην αντίστοιχη αίθουσα διδάσκεται το μάθημα με το συγκεκριμένο αύξοντα αριθμό.

Με το συγκεκριμένο τρόπο αναπαράστασης της λύσης ο δεύτερος ανελαστικός (σκληρός) περιορισμός του προβλήματος (Χρήση Αιθουσών – Room Occupancy), ο οποίος δηλώνει ότι σε κάθε ώρα της κάθε ημέρας και σε κάθε αίθουσα πρέπει να διδάσκεται το πολύ ένα μάθημα, ικανοποιείται αυτόματα αφού κάθε στοιχείο του πίνακα αναπαράστασης της λύσης (που αντιστοιχεί σε συγκεκριμένη ώρα συγκεκριμένης ημέρας και συγκεκριμένης αίθουσας) έχει μία μόνο τιμή που δηλώνει το μάθημα που διδάσκεται (ή είναι μηδέν αν δεν διδάσκεται κανένα μάθημα) και είναι αδύνατο να πάρει περισσότερες από μία τιμές.

Το γεγονός ότι ένας από τους περιορισμούς του προβλήματος ικανοποιείται αυτόματα από την επιλεγμένη αναπαράστασης της λύσης βοηθά τον οποιοδήποτε αλγόριθμο βελτιστοποίησης στην εύρεση μιας καλύτερης λύσης.

4.11 Αντικειμενικές συναρτήσεις στην πρώτη και δεύτερη υλοποίηση

Οι αντικειμενικές συναρτήσεις για την πρώτη και δεύτερη υλοποίηση υπολογίζονται βάσει ενός τύπου γραμμική μορφής:

$$O_i = w_i * v_i$$

όπου:

- O_i η τιμή της αντικειμενικής συνάρτησης για τον περιορισμό i.
- w_i το σταθερό (προκαθορισμένο) βάρος για τον περιορισμό i, το οποίο είναι ένας θετικός πραγματικός αριθμός.
- v_i το πλήθος των παραβιάσεων του περιορισμού i σε ολόκληρο τον πίνακα αναπαράστασης της λύσης.

Στην παραπάνω σχέση το i έχει τιμές 1, 2, 3, 4, 5, 6, 7 αφού όπως αναφέρθηκε παραπάνω ο δεύτερος ανελαστικός περιορισμός είναι αδύνατο να παραβιαστεί. Τα βάρη που χρησιμοποιούνται για τον υπολογισμό των αντικειμενικών συναρτήσεων καθορίζονται στην αρχή του αλγόριθμου και παραμένουν σταθερά σε όλη τη διάρκεια της εκτέλεσής του.

Το πλήθος των παραβιάσεων του κάθε περιορισμού υπολογίζεται εύκολα με τη συγκεκριμένη αναπαράσταση της λύσης. Πιο συγκεκριμένα:

- Για τον πρώτο ανελαστικό περιορισμό (πλήθος διαλέξεων – lectures) υπολογίζεται το πλήθος διαλέξεων που δίνεται για κάθε μάθημα, μετρώντας το πλήθος των στοιχείων του πίνακα που έχουν τιμή ίση με τον κωδικό) αύξοντα αριθμό του μαθήματος. Παραβίαση του περιορισμού υπάρχει αν το πλήθος αυτό είναι διαφορετικό από το καθορισμένο πλήθος διαλέξεων του μαθήματος.
- Για τον τρίτο ανελαστικό περιορισμό (συγκρούσεις – conflicts) για κάθε στοιχείο του πίνακα βρίσκεται ο κωδικός διδάσκοντα και ελέγχονται τα στοιχεία του πίνακα που αντιστοιχούν στην ίδια μέρα και ώρα αλλά σε διαφορετικές αίθουσες για το αν διδάσκει ο ίδιος διδάσκοντας εξετάζοντας το κωδικό διδάσκοντα του μαθήματος του αντίστοιχου στοιχείου. Παραβίαση του περιορισμού θεωρείται κάθε αλληλοεπικάλυψη παραδόσεων του ίδιου διδάσκοντα.
- Για τον τέταρτο ανελαστικό περιορισμό (διαθεσιμότητα – availability) ελέγχεται αν είναι διαθέσιμος ο διδάσκοντας του μαθήματος. Παραβίαση του περιορισμού θεωρείται εάν ο διδάσκοντας δεν είναι διαθέσιμος τη συγκεκριμένη ημέρα και ώρα.
- Για τον πρώτο ελαστικό περιορισμό (χωρητικότητα αιθουσών – room capacity) ελέγχεται αν η χωρητικότητα της συγκεκριμένης αίθουσας στην οποία έχει ανατεθεί ένα μάθημα είναι μεγαλύτερη ή ίση από τον αριθμό των φοιτητών που έχουν δηλώσει ότι θα παρακολουθούν το μάθημα. Παραβίαση του περιορισμού θεωρείται η διαφορά ανάμεσα στον αριθμό των φοιτητών που έχουν δηλώσει ότι θα παρακολουθούν το μάθημα και τη χωρητικότητα των αιθουσών.
- Για τον δεύτερο ελαστικό περιορισμό (ελάχιστος αριθμός ημερών – minimum working days) για κάθε μάθημα υπολογίζεται το πλήθος των διαφορετικών ημερών στις οποίες έχουν κατανεμηθεί οι διαλέξεις του συγκεκριμένου μαθήματος. Εφόσον το πλήθος αυτό είναι μικρότερο από τον ελάχιστο απαιτούμενο αριθμό ημερών, παραβίαση του περιορισμού θεωρείται η διαφορά του ελάχιστου απαιτούμενου αριθμού ημερών από το πλήθος των ημερών.
- Για τον τρίτο ελαστικό περιορισμό (πιεσμένο πρόγραμμα σπουδών – curriculum compactness) για κάθε μάθημα ενός προγράμματος ελέγχεται αν υπάρχουν κενές ώρες ή ώρες που αντιστοιχούν σε μάθημα άλλου προγράμματος σπουδών ανάμεσα στις ώρες που διδάσκεται το συγκεκριμένο μάθημα.
- Για τον τέταρτο ελαστικό περιορισμό (σταθερότητα αιθουσών – room stability) για κάθε μάθημα καταγράφεται το πλήθος των αιθουσών στις οποίες είναι κατανεμημένες οι διαλέξεις του μαθήματος. Εάν το πλήθος των αιθουσών είναι

μεγαλύτερο της μονάδας, παραβίαση θεωρείται το πλήθος των αιθουσών μείον τη μονάδα.

Η τιμή της συνάρτησης καταλληλότητας F ορίζεται ως

$$F = \sum_{i=1}^{7} O_i$$

όπου O_i η τιμή της αντικειμενικής συνάρτησης για τον περιορισμό i, όπως ορίστηκε παραπάνω.

Με τον τρόπο που ορίστηκαν οι αντικειμενικές συναρτήσεις και η συνάρτηση καταλληλότητας μπορούν να έχουν μόνο θετικές τιμές και η ελάχιστη τιμή της, η οποία είναι μηδέν (0) αντιστοιχεί σε ένα ωρολόγιο πρόγραμμα το οποίο δεν παραβιάζει κανέναν περιορισμό κατά συνέπεια μπορεί να θεωρηθεί ως βέλτιστη λύση για το πρόβλημα. Κατά συνέπεια το πρόβλημα είναι ένα πρόβλημα ελαχιστοποίησης.

4.12 Δημιουργία αρχικού πληθυσμού στην πρώτη και δεύτερη υλοποίηση

Η δημιουργία του αρχικού πληθυσμού γίνεται θέτοντας σε κάθε στοιχείο κάθε πιθανής λύσης τιμές φυσικών αριθμών στο διάστημα [1, C], όπου C ο μέγιστος αριθμός μαθημάτων (courses) που διδάσκονται.

Αυτός ο τρόπος δημιουργίας του αρχικού πληθυσμού επιλέχθηκε λόγω της απλότητας στην υλοποίησή του αφού στο MATLAB υλοποιείται με την κλήση μίας μόνο συνάρτησης. Ωστόσο παρουσιάζει ένα βασικό μειονέκτημα το οποίο έγινε εμφανές από τα αποτελέσματα που προέκυψαν από τις εκτελέσεις των δύο πρώτων υλοποιήσεων του κώδικα.

Το μειονέκτημα αυτό είναι ότι επιτρέπει την παραβίαση των περισσότερων περιορισμών με αποτέλεσμα ο αρχικός πληθυσμός των λύσεων να είναι πολύ κακός (να έχει μεγάλες τιμές της συνάρτησης καταλληλότητας) το οποίο με τη σειρά του επιδρά αρνητικά στη σύγκλιση του αλγόριθμου.

4.13 Τροποποίηση λύσεων κατά την αναζήτηση και ιχνηλάτηση στην πρώτη και δεύτερη υλοποίηση

Κατά τις διαδικασίες αναζήτησης και ιχνηλάτησης απαιτείται η τροποποίηση μέρους των λύσεων βάσει του αλγόριθμου της κάθε διαδικασίας. Ο αριθμός των αλλαγών που πρέπει να γίνει καθορίζεται σε κάθε διαδικασία βάσει των κριτηρίων που περιγράφηκαν σε προηγούμενες παραγράφους.

Η αλλαγή της λύσης γίνεται εναλλάσσοντας τις τιμές δύο στοιχείων του πίνακα της λύσης. Το πλήθος των εναλλαγών είναι διαφορετικό για κάθε διαδικασία και για κάθε πιθανή λύση, όπως προκύπτει από τους αλγόριθμους των διαδικασιών.

Η διαδικασία τροποποίησης όπως υλοποιήθηκε δεν ελέγχει αν αυξάνεται η τιμή της συνάρτησης καταλληλότητας με την επιβαλλόμενη αλλαγή, δηλαδή αν προκύπτει χειρότερη λύση. Επιτρέποντας την ύπαρξη χειρότερων λύσεων στον πληθυσμό αυξάνεται η ποικιλότητά του – ή αντίστοιχα μειώνεται ο κορεσμός – αποτρέποντας (θεωρητικά) τον εγκλωβισμό σε τοπικά ακρότατα αλλά ταυτόχρονα δυσχεραίνεται η σύγκλισή του.

4.14 Τροποποίηση λύσεων κατά την προσομοιωμένη ανόπτηση στη δεύτερη υλοποίηση

Στην προσομοιωμένη ανόπτηση απαιτείται η ύπαρξη μιας διαδικασίας για την εύρεση ενός νέου σημείου στο χώρο λύσεων – δηλαδή μιας νέας λύσης – η απόσταση της οποίας από την τρέχουσα λύση εξαρτάται από τη θερμοκρασία. Στη συγκεκριμένη υλοποίηση ως απόσταση θεωρείται το πλήθος των διαφορετικών στοιχείων μεταξύ δύο λύσεων. Η τροποποίηση των λύσεων για την εύρεση της νέας λύσης γίνεται αλλάζοντας τυχαία τις τιμές του – καθοριζόμενου από τη θερμοκρασία – πλήθους στοιχείων.

4.15 Αντικειμενικές συναρτήσεις στην τρίτη υλοποίηση

Οι αντικειμενικές συναρτήσεις για την τρίτη υλοποίηση υπολογίζονται ως ο αριθμός των παραβιάσεων των περιορισμών:

$$O_i = v_i$$

όπου:

- O_i η τιμή της αντικειμενικής συνάρτησης για τον περιορισμό i.

- v_i το πλήθος των παραβιάσεων του περιορισμού i σε ολόκληρο τον πίνακα αναπαράστασης της λύσης.

Στην παραπάνω σχέση το i έχει τιμές 1, 2, 3, 4, 5, 6, 7 αφού όπως αναφέρθηκε παραπάνω ο δεύτερος ανελαστικός περιορισμός είναι αδύνατο να παραβιαστεί.
Το πλήθος των παραβιάσεων του κάθε περιορισμού υπολογίζεται εύκολα με τη συγκεκριμένη αναπαράσταση της λύσης. Πιο συγκεκριμένα:

- Για τον πρώτο ανελαστικό περιορισμό (πλήθος διαλέξεων – lectures) υπολογίζεται το πλήθος διαλέξεων που δίνεται για κάθε μάθημα, μετρώντας το πλήθος των στοιχείων του πίνακα που έχουν τιμή ίση με τον κωδικό) αύξοντα αριθμό του μαθήματος. Παραβίαση του περιορισμού υπάρχει αν το πλήθος αυτό είναι διαφορετικό από το καθορισμένο πλήθος διαλέξεων του μαθήματος.
- Για τον τρίτο ανελαστικό περιορισμό (συγκρούσεις – conflicts) για κάθε στοιχείο του πίνακα βρίσκεται ο κωδικός διδάσκοντα και ελέγχονται τα στοιχεία του πίνακα που αντιστοιχούν στην ίδια μέρα και ώρα αλλά σε διαφορετικές αίθουσες για το αν διδάσκει ο ίδιος διδάσκοντας εξετάζοντας το κωδικό διδάσκοντα του μαθήματος του αντίστοιχου στοιχείου. Παραβίαση του περιορισμού θεωρείται κάθε αλληλεπικάλυψη παραδόσεων του ίδιου διδάσκοντα.
- Για τον τέταρτο ανελαστικό περιορισμό (διαθεσιμότητα – availability) ελέγχεται αν είναι διαθέσιμος ο διδάσκοντας του μαθήματος. Παραβίαση του περιορισμού θεωρείται εάν ο διδάσκοντας δεν είναι διαθέσιμος τη συγκεκριμένη ημέρα και ώρα.
- Για τον πρώτο ελαστικό περιορισμό (χωρητικότητα αιθουσών – room capacity) ελέγχεται αν η χωρητικότητα της συγκεκριμένης αίθουσας στην οποία έχει ανατεθεί ένα μάθημα είναι μεγαλύτερη ή ίση από τον αριθμό των φοιτητών που έχουν δηλώσει ότι θα παρακολουθούν το μάθημα. Παραβίαση του περιορισμού θεωρείται η διαφορά ανάμεσα στον αριθμό των φοιτητών που έχουν δηλώσει ότι θα παρακολουθούν το μάθημα και τη χωρητικότητα των αιθουσών.
- Για τον δεύτερο ελαστικό περιορισμό (ελάχιστος αριθμός ημερών – minimum working days) για κάθε μάθημα υπολογίζεται το πλήθος των διαφορετικών ημερών στις οποίες έχουν κατανεμηθεί οι διαλέξεις του συγκεκριμένου μαθήματος. Εφόσον το πλήθος αυτό είναι μικρότερο από τον ελάχιστο απαιτούμενο αριθμό ημερών,

παραβίαση του περιορισμού θεωρείται η διαφορά του ελάχιστου απαιτούμενου αριθμού ημερών από το πλήθος των ημερών.

- Για τον τρίτο ελαστικό περιορισμό (πιεσμένο πρόγραμμα σπουδών – curriculum compactness) για κάθε μάθημα ενός προγράμματος ελέγχεται αν υπάρχουν κενές ώρες ή ώρες που αντιστοιχούν σε μάθημα άλλου προγράμματος σπουδών ανάμεσα στις ώρες που διδάσκεται το συγκεκριμένο μάθημα.
- Για τον τέταρτο ελαστικό περιορισμό (σταθερότητα αιθουσών – room stability) για κάθε μάθημα καταγράφεται το πλήθος των αιθουσών στις οποίες είναι κατανεμημένες οι διαλέξεις του μαθήματος. Εάν το πλήθος των αιθουσών είναι μεγαλύτερο της μονάδας, παραβίαση θεωρείται το πλήθος των αιθουσών μείον τη μονάδα.

Η τιμή της συνάρτησης καταλληλότητας F ορίζεται ως

$$F = \sum_{i=1}^{7} O_i$$

όπου O_i η τιμή της αντικειμενικής συνάρτησης για τον περιορισμό i, όπως ορίστηκε παραπάνω.

Με τον τρόπο που ορίστηκαν οι αντικειμενικές συναρτήσεις και η συνάρτηση καταλληλότητας μπορούν να έχουν μόνο θετικές τιμές και η ελάχιστη τιμή της, η οποία είναι μηδέν (0) αντιστοιχεί σε ένα ωρολόγιο πρόγραμμα το οποίο δεν παραβιάζει κανέναν περιορισμό κατά συνέπεια μπορεί να θεωρηθεί ως βέλτιστη λύση για το πρόβλημα. Κατά συνέπεια το πρόβλημα είναι ένα πρόβλημα ελαχιστοποίησης.

4.16 Δημιουργία αρχικού πληθυσμού στην τρίτη υλοποίηση

Για τη δημιουργία του αρχικού πληθυσμού των λύσεων στην τρίτη υλοποίηση αναπτύχθηκε μια πιο διαδικασία η οποία εξασφαλίζει ότι κάθε αρχική λύση ικανοποιεί πλήρως περισσότερους από έναν περιορισμούς δηλαδή οι τιμές κάποιων αντικειμενικών συναρτήσεων είναι μηδέν (0) για κάθε αρχική λύση.

Η διαδικασία με την οποία δημιουργείται κάθε αρχική λύση είναι η ακόλουθη. Ο πίνακας αναπαράστασης της λύσης αρχικοποιείται με την τιμή μηδέν (0). Για κάθε διαθέσιμο μάθημα [1, C], όπου C ο μέγιστος αριθμός μαθημάτων (courses) που διδάσκονται εκτελούνται τα ακόλουθα βήματα:

1. Από το πλήθος των διαλέξεων που πρέπει να διδαχθούν και τον ελάχιστο αριθμό ημερών στις οποίες πρέπει αυτές να κατανεμηθούν υπολογίζεται ο αριθμός διαλέξεων ανά ημέρα.
2. Λαμβάνοντας υπόψη τη χωρητικότητα των αιθουσών και το πλήθος των φοιτητών που παρακολουθούν το μάθημα βρίσκονται οι αίθουσες στις οποίες μπορεί να διδαχθεί το μάθημα.
3. Λαμβάνοντας υπόψη τη διαθεσιμότητα του διδάσκοντα δημιουργείται ένας πίνακας με τις διαθέσιμες ημέρες και ώρες στις οποίες μπορεί να διδαχθεί το μάθημα.
4. Επιλέγουμε τυχαία μία από τις διαθέσιμες για το μάθημα αίθουσες (οι οποίες βρέθηκαν στο βήμα 2).
5. Επιλέγουμε τυχαία μία από τις διαθέσιμες ημέρες στις οποίες μπορεί να διδαχθεί το μάθημα.
6. Από τον πίνακα που δημιουργήθηκε στο βήμα 3 αφαιρούνται οι ημέρες και ώρες στις οποίες διδάσκεται ήδη ένα μάθημα στην επιλεγμένη αίθουσα ή διδάσκεται μάθημα του ίδιου διδάσκοντα σε οποιαδήποτε αίθουσα.
7. Επιλέγονται τυχαία συνεχόμενες ώρες στις οποίες μπορεί να διδαχθεί ο προκαθορισμένος αριθμός διαλέξεων ανά ημέρα.
8. Η διαδικασία επαναλαμβάνεται από το βήμα 3 μέχρι να κατανεμηθεί το απαραίτητο πλήθος διαλέξεων για το συγκεκριμένο μάθημα.

Η διαδικασία αυτή κάθε παραγόμενη αρχική λύση πρέπει να ικανοποιεί τους ακόλουθους περιορισμούς:

- Τον πρώτο ανελαστικό περιορισμό (πλήθος διαλέξεων – lectures).
- Τον τρίτο ανελαστικό περιορισμό (συγκρούσεις – conflicts).
- Τον τέταρτο ανελαστικό περιορισμό (διαθεσιμότητα – availability).
- Τον πρώτο ελαστικό περιορισμό (χωρητικότητα αιθουσών – room capacity).
- Τον δεύτερο ελαστικό περιορισμό (ελάχιστος αριθμός ημερών – minimum working days).

Ωστόσο σε αρχικές δοκιμές που έγιναν για την ορθή λειτουργία της διαδικασίας παρατηρήθηκε ότι σε ορισμένα από τα διαθέσιμα δεδομένα δεν ήταν δυνατή η δημιουργία αρχικής λύσης. Για το λόγο αυτό ενσωματώθηκαν στη διαδικασία κριτήρια τα οποία παρακάμπτουν κάποιους από τους περιορισμούς προκειμένου να δημιουργηθεί μία αρχική λύση. Όπως φάνηκε από τα αποτελέσματα η διαδικασία αυτή επιτυγχάνει την ικανοποίηση τουλάχιστον δύο από τους παραπάνω περιορισμούς σε όλα τα διαθέσιμα δεδομένα, και στα περισσότερα από αυτά επιτυγχάνει την ικανοποίηση τριών από τους παραπάνω περιορισμούς. Λαμβάνοντας υπόψη ότι ο δεύτερος ανελαστικός περιορισμός ικανοποιείται πάντα – λόγω της αναπαράστασης της λύσης – η διαδικασία δημιουργεί αρχικές λύσεις που ικανοποιούν τουλάχιστον τρεις περιορισμούς σε όλα τα διαθέσιμα δεδομένα, ενώ στην πλειονότητα των διαθέσιμων δεδομένων οι αρχικές λύσεις που παράγονται ικανοποιούν τέσσερις περιορισμούς.

4.17 Τροποποίηση λύσεων κατά την αναζήτηση και ιχνηλάτηση στην τρίτη υλοποίηση

Κατά τις διαδικασίες αναζήτησης και ιχνηλάτησης απαιτείται η τροποποίηση μέρους των λύσεων βάσει του αλγόριθμου της κάθε διαδικασίας. Ο αριθμός των αλλαγών που πρέπει να γίνει καθορίζεται σε κάθε διαδικασία βάσει των κριτηρίων που περιγράφηκαν σε προηγούμενες παραγράφους.

Η αλλαγή της λύσης γίνεται εναλλάσσοντας τις τιμές δύο στοιχείων του πίνακα της λύσης. Το πλήθος των εναλλαγών είναι διαφορετικό για κάθε διαδικασία και για κάθε πιθανή λύση, όπως προκύπτει από τους αλγόριθμους των διαδικασιών.

Η διαδικασία τροποποίησης όπως υλοποιήθηκε δεν ελέγχει αν αυξάνεται η τιμή της συνάρτησης καταλληλότητας με την επιβαλλόμενη αλλαγή, δηλαδή αν προκύπτει χειρότερη λύση. Επιτρέποντας την ύπαρξη χειρότερων λύσεων στον πληθυσμό αυξάνεται η ποικιλότητά του – ή αντίστοιχα μειώνεται ο κορεσμός – αποτρέποντας (θεωρητικά) τον εγκλωβισμό σε τοπικά ακρότατα αλλά ταυτόχρονα δυσχεραίνεται η σύγκλισή του.

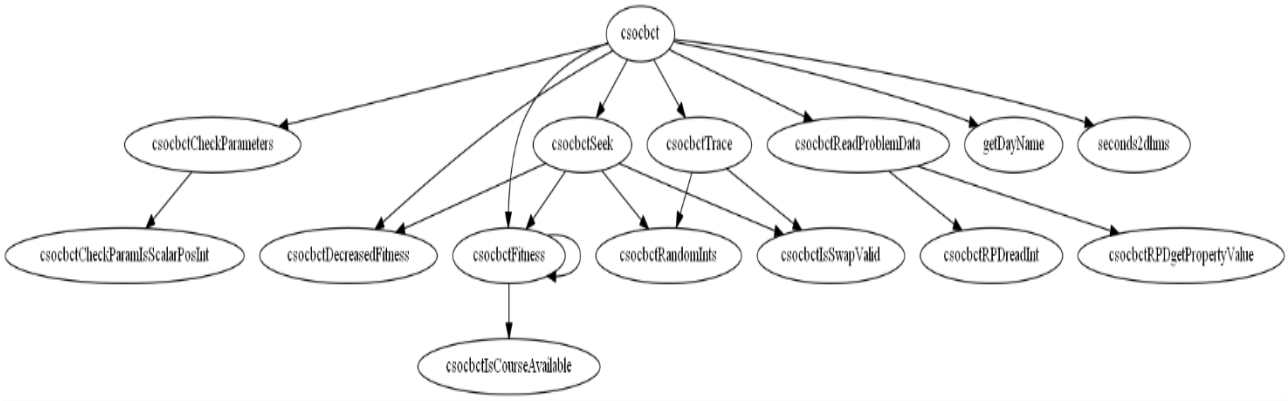
4.18 Τροποποίηση λύσεων κατά την προσομοιωμένη ανόπτηση στην τρίτη υλοποίηση

Όπως αναφέρθηκε παραπάνω, στην προσομοιωμένη ανόπτηση απαιτείται η ύπαρξη μιας διαδικασίας για την εύρεση ενός νέου σημείου στο χώρο λύσεων – δηλαδή μιας νέας λύσης – η απόσταση της οποίας από την τρέχουσα λύση εξαρτάται από τη θερμοκρασία. Στη συγκεκριμένη υλοποίηση ως απόσταση θεωρείται το πλήθος των διαφορετικών στοιχείων μεταξύ δύο λύσεων. Η τροποποίηση των λύσεων για την εύρεση της νέας λύσης γίνεται αλλάζοντας τυχαία τις τιμές του – καθοριζόμενου από τη θερμοκρασία – πλήθους στοιχείων.

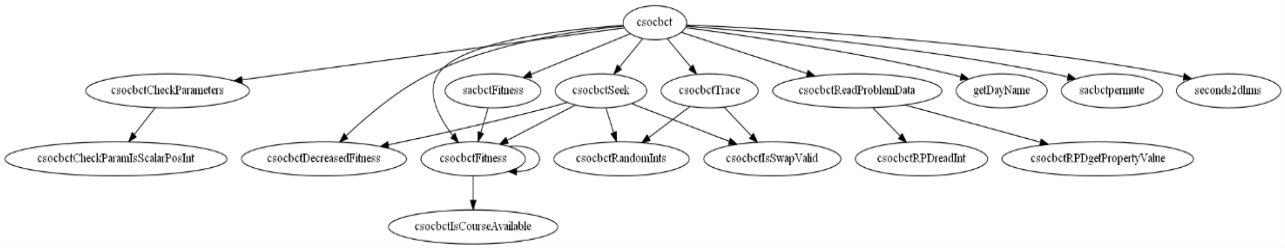
4.19 Διαγράμματα κλήσεων συναρτήσεων

Ακολούθως παρουσιάζονται τα διαγράμματα κλήσεων των συναρτήσεων για τις τρεις υλοποιήσεις του κώδικα.

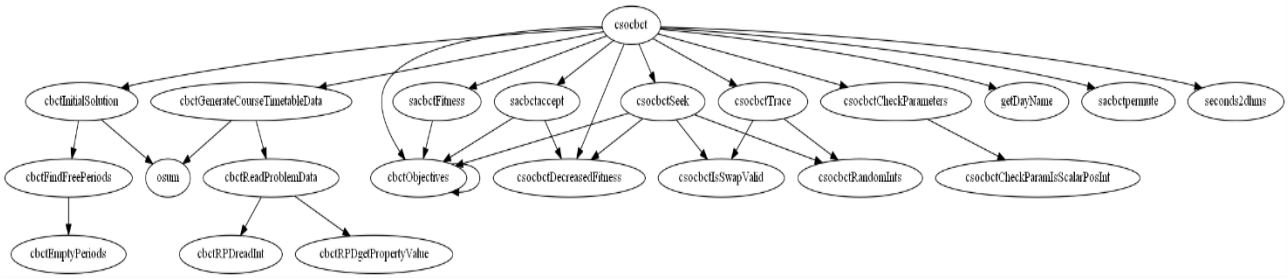
Διάγραμμα κλήσεων συναρτήσεων για την πρώτη υλοποίηση



Διάγραμμα κλήσεων συναρτήσεων για τη δεύτερη υλοποίηση



Διάγραμμα κλήσεων συναρτήσεων για την τρίτη υλοποίηση



5ο Κεφάλαιο

5.1 Αποτελέσματα

Για την εκτέλεση του κώδικα χρησιμοποιήθηκαν τα δεδομένα που είναι διαθέσιμα από τον Διεθνή Διαγωνισμό Ωρολόγιου Προγράμματος (International Timetabling Competition – http://www.cs.qub.ac.uk/eventmap/), τα οποία είναι διαθέσιμα στην ιστοσελίδα του διαγωνισμού http://www.cs.qub.ac.uk/eventmap/Login/SecretPage.php. Στην ιστοσελίδα αυτή είναι διαθέσιμα συνολικά 21 αρχεία δεδομένων τα οποία αντιστοιχούν σε ισάριθμα προβλήματα. Η δομή των αρχείων αυτών περιγράφεται στην ιστοσελίδα http://www.cs.qub.ac.uk/eventmap/curriculmcourse/course_curriculm_index_files/Inputformat.htm. Βάσει αυτής της περιγραφής αναπτύχθηκε κατάλληλη συνάρτηση η οποία διαβάζει το κάθε αρχείο και εξάγει τα δεδομένα που είναι απαραίτητα για την επίλυση του προβλήματος.

Οι κώδικες τις πρώτης και δεύτερης υλοποίησης εκτελέστηκαν για 3 από τα 21 δεδομένα, συγκεκριμένα για τα αρχεία comp01.ctt, comp02.ctt, comp03.ctt. Η εκτέλεσή τους για τα υπόλοιπα αρχεία δεδομένων παραλήφθηκε διότι από τα αποτελέσματα που παρατίθενται στις επόμενες υποενότητες έγινε φανερό ότι οι υλοποιήσεις αυτές δεν κατόρθωναν να συγκλίνουν σε καλές λύσεις. Οι τελικές λύσεις που προέκυψαν από τις εκτελέσεις των δύο πρώτων υλοποιήσεων παραβιάζουν πολλούς περιορισμούς.

Αντίθετα οι τελικές λύσεις που προέκυψαν από τις εκτελέσεις της τρίτης υλοποίησης ικανοποιούν αρκετούς περιορισμούς και για το λόγο αυτό η Τρίτη υλοποίηση εκτελέστηκε και με τα 21 διαθέσιμα αρχεία (προβλήματα).

Στις ακόλουθες υποενότητες παρουσιάζονται συνοπτικά τα αποτελέσματα που λήφθηκαν για κάθε αρχείο (πρόβλημα) από κάθε υλοποίηση. Στις υποενότητες αυτές παρουσιάζονται μόνο οι τιμές της καταλληλότητας και των αντικειμενικών συναρτήσεων από την πρώτη και την τελευταία επανάληψη..

Για όλες τις εκτελέσεις χρησιμοποιήθηκαν οι ακόλουθες παράμετροι:

Αριθμός γατών (μέγεθος πληθυσμού)	100
Επαναλήψεις	50
λόγος ανάμιξης (Mixture Ratio) MR	0.2
μέγεθος της μνήμης αναζήτησης (seeking memory pool) SMP	5
συμπερίληψη της τρέχουσας θέσης (self position consideration) SPC	Αληθής
πλήθος των διαστάσεων που θα μεταβληθούν (counts of dimension to change) (CDC)	0.8
εύρος αναζήτησης (seeking range) (SRD)	0.2
εύρος ιχνηλάτησης (trace distance) (TD)	1.0

Επιπρόσθετα τα βάρη που χρησιμοποιήθηκαν για τον υπολογισμό των αντικειμενικών συναρτήσεων στις δύο πρώτες υλοποιήσεις ήταν τα ακόλουθα:

Πρώτη αντικειμενική συνάρτηση (πρώτος ανελαστικός περιορισμός)	10
Δεύτερη αντικειμενική συνάρτηση (τρίτος ανελαστικός περιορισμός)	5
Τρίτη αντικειμενική συνάρτηση (τεταρτος ανελαστικός περιορισμός)	10
Τέταρτη αντικειμενική συνάρτηση (πρώτος ελαστικός περιορισμός)	1
Πέμπτη αντικειμενική συνάρτηση (δεύτερος ελαστικός περιορισμός)	2
Έκτη αντικειμενική συνάρτηση (τρίτος ελαστικός περιορισμός)	5
Έβδομη αντικειμενική συνάρτηση (τέταρτος ελαστικός περιορισμός)	2

5.2 Αποτελέσματα πρώτης υλοποίησης

Πρώτο πρόβλημα (αρχείο comp01.ctt)

Επανάληψη	Καταλληλότητα	Αντικειμενικές						
		1	2	3	4	5	6	7
Αρχική	3893	280	150	1700	1091	152	160	360
Τελική	2162	150	50	1440	10	152	0	360

Δεύτερο πρόβλημα (αρχείο comp02.ctt)

Επανάληψη	Καταλληλότητα	Αντικειμενικές						
		1	2	3	4	5	6	7
Αρχική	13997	660	500	2920	8776	326	15	800
Τελική	5118	570	200	2690	532	326	0	800

Τρίτο πρόβλημα (αρχείο comp03.ctt)

Επανάληψη	Καταλληλότητα	Αντικειμενικές						
		1	2	3	4	5	6	7
Αρχική	11128	600	470	3210	5575	288	185	800
Τελική	5208	530	180	2730	680	288	0	800

5.3 Αποτελέσματα δεύτερης υλοποίησης

Πρώτο πρόβλημα (αρχείο comp01.ctt)

Επανάληψη	Καταλληλότητα	Αντικειμενικές						
		1	2	3	4	5	6	7
Αρχική	5205	280	160	1640	2563	152	50	360
Τελική	2463	300	80	1570	1	152	0	360

Δεύτερο πρόβλημα (αρχείο comp02.ctt)

Επανάληψη	Καταλληλότητα	Αντικειμενικές						
		1	2	3	4	5	6	7
Αρχική	13440	680	520	3040	7204	326	870	800
Τελική	4744	800	480	2320	18	326	0	800

Τρίτο πρόβλημα (αρχείο comp03.ctt)

Επανάληψη	Καταλληλότητα	Αντικειμενικές						
		1	2	3	4	5	6	7
Αρχική	15273	620	610	3180	3235	288	540	800
Τελική	4911	720	380	2670	53	288	0	800

5.4 Αποτελέσματα τρίτης υλοποίησης

Πρώτο πρόβλημα (αρχείο comp01.ctt)

Επανάληψη	Καταλληλότητα	Αντικειμενικές						
		1	2	3	4	5	6	7
Αρχική	240	2	0	6	0	76	0	156
Τελική	106	30	0	0	0	76	0	0

Δεύτερο πρόβλημα (αρχείο comp02.ctt)

Επανάληψη	Καταλληλότητα	Αντικειμενικές						
		1	2	3	4	5	6	7
Αρχική	739	0	22	218	0	172	0	327
Τελική	245	82	0	0	0	163	0	0

Τρίτο πρόβλημα (αρχείο comp03.ctt)

Επανάληψη	Καταλληλότητα	Αντικειμενικές						
		1	2	3	4	5	6	7
Αρχική	593	0	4	194	0	144	0	251
Τελική	216	72	0	0	0	144	0	0

Τέταρτο πρόβλημα (αρχείο comp04.ctt)

Επανάληψη	Καταλληλότητα	Αντικειμενικές						
		1	2	3	4	5	6	7
Αρχική	610	0	2	186	0	136	0	286
Τελική	215	79	0	0	0	136	0	0

Πέμπτο πρόβλημα (αρχείο comp05.ctt)

Επανάληψη	Καταλληλότητα	Αντικειμενικές						
		1	2	3	4	5	6	7
Αρχική	303	1	2	54	0	95	0	151
Τελική	149	54	0	0	0	95	0	0

Έκτο πρόβλημα (αρχείο comp06.ctt)

Επανάληψη	Καταλληλότητα	Αντικειμενικές						
		1	2	3	4	5	6	7
Αρχική	835	0	18	251	0	250	0	361
Τελική	313	108	0	0	0	205	0	0

Έβδομο πρόβλημα (αρχείο comp07.ctt)

Επανάληψη	Καταλληλότητα	Αντικειμενικές						
		1	2	3	4	5	6	7
Αρχική	951	0	34	244	0	239	0	434
Τελική	370	131	0	0	0	239	0	0

Όγδοο πρόβλημα (αρχείο comp08.ctt)

Επανάληψη	Καταλληλότητα	Αντικειμενικές						
		1	2	3	4	5	6	7
Αρχική	665	0	10	175	0	156	0	324
Τελική	242	86	0	0	0	156	0	0

Ένατο πρόβλημα (αρχείο comp09.ctt)

Επανάληψη	Καταλληλότητα	Αντικειμενικές						
		1	2	3	4	5	6	7
Αρχική	612	0	6	183	0	144	0	279
Τελική	220	76	0	0	0	144	0	0

Δέκατο πρόβλημα (αρχείο comp10.ctt)

Επανάληψη	Καταλληλότητα	Αντικειμενικές						
		1	2	3	4	5	6	7
Αρχική	797	0	18	205	0	240	0	370
Τελική	319	115	0	0	0	204	0	0

Ενδέκατο πρόβλημα (αρχείο comp11.ctt)

Επανάληψη	Καταλληλότητα	Αντικειμενικές						
		1	2	3	4	5	6	7
Αρχική	231	1	2	2	0	67	0	159
Τελική	97	30	0	0	0	67	0	0

Δωδέκατο πρόβλημα (αρχείο comp12.ctt)

Επανάληψη	Καταλληλότητα	Αντικειμενικές						
		1	2	3	4	5	6	7
Αρχική	406	0	6	52	0	130	0	218
Τελική	218	88	0	0	0	130	0	0

Δέκατο Τρίτο πρόβλημα (αρχείο comp13.ctt)

Επανάληψη	Καταλληλότητα	Αντικειμενικές						
		1	2	3	4	5	6	7
Αρχική	659	0	6	197	0	148	0	308
Τελική	230	82	0	0	0	148	0	0

Δέκατο Τέταρτο πρόβλημα (αρχείο comp14.ctt)

Επανάληψη	Καταλληλότητα	Αντικειμενικές						
		1	2	3	4	5	6	7
Αρχική	624	0	4	173	0	172	0	275
Τελική	257	85	0	0	0	172	0	0

Δέκατο Πέμπτο πρόβλημα (αρχείο comp15.ctt)

Επανάληψη	Καταλληλότητα	Αντικειμενικές						
		1	2	3	4	5	6	7
Αρχική	589	0	2	192	0	144	0	251
Τελική	216	72	0	0	0	144	0	0

Δέκατο Έκτο πρόβλημα (αρχείο comp16.ctt)

Επανάληψη	Καταλληλότητα	Αντικειμενικές						
		1	2	3	4	5	6	7
Αρχική	793	0	14	209	0	204	0	366
Τελική	312	108	0	0	0	204	0	0

Δέκατο Έβδομο πρόβλημα (αρχείο comp17.ctt)

Επανάληψη	Καταλληλότητα	Αντικειμενικές						
		1	2	3	4	5	6	7
Αρχική	758	0	18	215	0	186	0	339
Τελική	285	99	0	0	0	186	0	0

Δέκατο Όγδοο πρόβλημα (αρχείο comp18.ctt)

Επανάληψη	Καταλληλότητα	Αντικειμενικές						
		1	2	3	4	5	6	7
Αρχική	239	0	0	10	0	91	0	138
Τελική	138	47	0	0	0	91	0	0

Δέκατο Ένατο πρόβλημα (αρχείο comp19.ctt)

Επανάληψη	Καταλληλότητα	Αντικειμενικές						
		1	2	3	4	5	6	7
Αρχική	627	0	8	189	0	153	0	277
Τελική	227	74	0	0	0	153	0	0

Εικοστό πρόβλημα (αρχείο comp20.ctt)

Επανάληψη	Καταλληλότητα	Αντικειμενικές						
		1	2	3	4	5	6	7
Αρχική	890	1	18	263	0	220	0	388
Τελική	341	121	0	0	0	220	0	0

Εικοστό Πρώτο πρόβλημα (αρχείο comp21.ctt)

Επανάληψη	Καταλληλότητα	Αντικειμενικές						
		1	2	3	4	5	6	7
Αρχική	739	0	22	218	0	172	0	327
Τελική	266	94	0	0	0	172	0	0

5.5 Συμπεράσματα αποτελεσμάτων

Όπως αναφέρθηκε και στην αρχή της παρούσας ενότητας το πρώτο βασικό συμπέρασμα που προκύπτει από τα αποτελέσματα των τριών υλοποιήσεων είναι ότι η τρίτη υλοποίηση επιτυγχάνει σημαντικά καλύτερα αποτελέσματα από τις δύο πρώτες. Δεδομένου ότι η τρίτη υλοποίηση χρησιμοποιεί διαφορετική τεχνική για την αρχικοποίηση του πληθυσμού των λύσεων γίνεται φανερό ότι η επίδραση της αρχικοποίησης στη σύγκλιση του αλγόριθμου είναι σημαντική.

Πρέπει να σημειωθεί στο σημείο αυτό ότι δεν συγκρίνονται οι τιμές των αντικειμενικών συναρτήσεων ανάμεσα στις τρεις υλοποιήσεις – αφού οι δυο πρώτες χρησιμοποιούν άλλο τρόπο υπολογισμού από την τρίτη – αλλά το πόσο χαμηλότερη είναι η τιμή της κάθε αντικειμενικής συνάρτησης σε σχέση με την αρχική της.

Ένα δεύτερο σημαντικό συμπέρασμα είναι ότι η τρίτη υλοποίηση επιτυγχάνει την ικανοποίηση αρκετών περιορισμών σε όλα τα προβλήματα.

Ένα τρίτο συμπέρασμα είναι ότι η τρίτη υλοποίηση σε πολλές περιπτώσεις αποτυγχάνει να ικανοποιήσει τον πρώτο ανελαστικό περιορισμό ακόμα και αν η αρχική λύση τον ικανοποιούσε.

Ένα τέταρτο συμπέρασμα είναι ότι η μείωση των τιμών των αντικειμενικών συναρτήσεων στις υλοποιήσεις 2 και 3 γίνεται κατά κύριο λόγω από την προσομοιωμένη ανόπτηση και όχι από τον αλγόριθμο cat swarm optimization.

Το γεγονός αυτό μας οδηγεί στο συμπέρασμα ότι ο τρόπος που χρησιμοποιήθηκε για τη μεταβολή της λύσης στις διαδικασίες αναζήτησης και ιχνηλάτησης του αλγόριθμου cat swarm optimization απαιτεί σημαντική βελτίωση ώστε να οδηγεί σε καλύτερη αναζήτηση λύσεων.

ΒΙΒΛΙΟΦΡΑΦΙΑ

S. Kirkpatrick C. D. Gelatt, Jr., M. P. Vecchi, "Optimization by Simulated Annealing", SCIENCE, 13 May 1983, Vol 220, Issue 4598, pp. 671-680, DOI: 10.1126/science.220.4598.671

Β. Σκουλλής «Εφαρμογή σύγχρονων αλγόριθμων Cat Swarm Optimization για την αποδοτική επίλυση του προβλήματος κατασκευής βέλτιστου ωρολογίου προγράμματος για σχολεία Δευτεροβάθμιας εκπαίδευσης» Ε.Α.Π.,Πάτρα,2015.

J. McCarthy, «What is Artificial Intelligence?,» 12 Νοέμβριος 2007.http://www.formal.stanford.edu/jmc/whatisai/whatisai.html.

Ν. Ζήκος «Εφαρμογή σύγχρονων τεχνικών υπολογιστικής νοημοσύνης Artificial Bee Colony (ABC) για την αποδοτική επίλυση του προβλήματος Curriculum based Course Timetabling» Ε.Α.Π.,Πάτρα, 2019.

Eberhart R. C., Kennedy J., A new optimizer using particle swarm theory, 1995, pp. 39-43.

Chu Shu-Chuan, Tsai Pei-Wei, Computational intelligence based on the behavior of cats, τόμ. III, 2007, pp. 163-173.

Kirkpatrick S., Gelatt Jr. C.D., Vecchi M.P., Optimization by Simulated Annealing,τόμ.220, 1983, pp. 671-680.

Wikipedia, «Τεχνητή νοημοσύνη,» http://el.wikipedia.org/wiki/Τεχνητή_νοημοσύνη.

N. Pillay, «A survey of school timetabling research,» σε Annals of Operations Research, τόμ. 218, Springer US, 2014, pp. 261-293.

I. Tassopoulos και G. Beligiannis, «A hybrid particle swarm optimization based algorithm for high school timetabling problems,» Applied Soft Computing, τόμ. 12, 2012, p. 3472–3489

I. Γκάτσης, «Αποδοτική επίλυση του προβλήματος School Timetabling (εύρεση βέλτιστου ωρολογίου προγράμματος για σχολεία δευτεροβάθμιας εκπαίδευσης) με χρήση αλγόριθμων Variable Neighborhood Search (VNS),» Ε.Α.Π., Πάτρα, 2014.

I. Κατσαραγάκης, «Εφαρμογή Σύγχρονων Τεχνικών Υπολογιστικής Νοημοσύνης για την Αποδοτική Επίλυση του Προβλήματος School Timetabling,» Ε.Α.Π., Πάτρα, 2014.

J. Ullman, «NP-Complete Scheduling Problems,» Journal of Computer and System Sciences, τόμ. 10, αρ. 3, pp. 384-393, 1975.

Wikipedia, «NP-complete,» http://en.wikipedia.org/wiki/NP-complete.

C. Mandl, Applied Network Optimization, Academic Press, 1979.

D. Abramson. Constructing School Timetables using Simulated Annealing: Sequential and Parallel Algorithms. Management Science, Vol 37, No 1, Jan 1991, pp 98 - 113.

A. Aggoun and N. Beldiceanu. Extending CHIP in order to solve complex scheduling and placement problems. Mathematical and Computer Modelling, 1993, pp. 57–73, 1993.

A. Atamturk, Martin W. P. Savelsbergh. Integer-Programming Software Systems. Annals of Operations Research 2005, pp. 67–124.

Tsai, Pei-Wei; Pan, Jeng-Shyang; Chen, Shyi-Ming; Liao, Bin-Yih; Hao, Szu-Ping;, «Parallel Cat Swarm Optimization,» Proceedings of the Seventh International Conference on Machine Learning and Cybernetics, τόμ. 6, pp. 3328-3333, 2008.

Chu, Shu-Chuan; Chen, Yi-Tin; Ho, Jiun-Huei;, «Timetabling scheduling using particle swarm optimization,» σε 1st International Conference on Innovative Computing, Information and Control, 2006.

Β. Κουρεπίνης «Εφαρμογή Σύγχρονων Αλγορίθμων Υπολογιστικής Νοημοσύνης για την Αποδοτική Επίλυση του Προβλήματος Δρομολόγησης Αστικών Μεταφορών» Ε.Α.Π., Πάτρα, 2019.

Orouskhani, Maysam; Mansouri, Mohammad; Teshnehlab, Mohammad;, «AverageInertia Weighted Cat Swarm Optimization,» σε Advances in Swarm Intelligence, 2011, pp. 321-328

A. Slowik «Swarm Intelligence Algorithm» pp. 55-68, 2020

Chu, Shu-Chuan; Tsai, Pei-Wei; Pan, Jeng-Shyang;, «Cat Swarm Optimization,» σε Proceedings of the 9th Pacific Rim International Conference on Artificial Intelligence, LNAI 4099, 2006, pp. 854-858.

Υπεύθυνη Δήλωση Συγγραφέα:

Δηλώνω ρητά ότι, σύμφωνα με το άρθρο 8 του Ν.1599/1986, η παρούσα εργασία αποτελεί αποκλειστικά προϊόν προσωπικής μου εργασίας, δεν προσβάλλει κάθε μορφής δικαιώματα διανοητικής ιδιοκτησίας, προσωπικότητας και προσωπικών δεδομένων τρίτων, δεν περιέχει έργα/εισφορές τρίτων για τα οποία απαιτείται άδεια των δημιουργών/δικαιούχων και δεν είναι προϊόν μερικής ή ολικής αντιγραφής, οι πηγές δε που χρησιμοποιήθηκαν περιορίζονται στις βιβλιογραφικές αναφορές και μόνον και πληρούν τους κανόνες της επιστημονικής παράθεσης.