Αλεξάνδρειο Τεχνολογικό Εκπαιδευτικό Ίδρυμα Θεσσαλονίκης
Σχολή Τεχνολογικών Εφαρμογών
Τμήμα Πληροφορικής

ΠΕΡΙΒΑΛΛΟΝ ΑΝΑΠΤΥΞΗΣ ΚΑΙ ΓΡΑΦΙΚΗΣ ΑΝΑΠΑΡΑΣΤΑΣΗΣ ΕΞΕΛΙΚΤΙΚΩΝ ΑΛΓΟΡΙΘΜΩΝ

Σπουδαστής: Αναστάσιος Σκαρλατίδης
Εισηγητής: Δρ. Παναγιώτης Αδαμίδης

Θεσσαλονίκη, Ιούνιος 2006

Αφιερώνεται στους γονείς μου,
για την αγάπη, τη συμπαράστασή τους
και την ευκαιρία που μου έδωσαν
να πραγματοποιήσω το όνειρό μου,
να σπουδάσω Πληροφορική.

Ευχαριστίες

Ευχαριστώ θερμά τον καθηγητή μου κ. Π. Αδαμίδη, για την πολύτιμη βοήθειά του, την καθοδήγησή του και την άριστη συνεργασία μας καθ' όλη την πορεία της παρούσας πτυχιακής εργασίας.

Περιεχόμενα

Κατάλογος Σχημάτων **ix**

1 Εισαγωγή **xv**

2 Εξελικτικοί Αλγόριθμοι **17**
2.1 Γιατί οι Εξελικτικοί Αλγόριθμοι 17
2.2 Τι είναι οι Εξελικτικοί Αλγόριθμοι 19
2.2.1 Δομή ενός Εξελικτικού Αλγόριθμου 21
2.3 Γενετικοι Αλγόριθμοι 23
2.4 Αναπαράσταση 24
2.4.1 Δυαδική Αναπαράσταση 25
2.4.2 Αναπαράσταση Ακεραίων 26
2.4.3 Αναπαράσταση Κινητής Υποδιαστολής 26
2.4.4 Αναπαράσταση Μετάθεσης 27
2.5 Συνάρτηση Ποιότητας 27
2.6 Αρχικοποίηση 27
2.7 Πληθυσμός 28
2.8 Ανασυνδυασμός 28
2.8.1 Τελεστές Ανασυνδυασμού για Δυαδική Αναπαράσταση . 28
2.8.2 Τελεστές Ανασυνδυασμού για Αναπαράσταση Κινητής Υποδιαστολής 29
2.8.3 Τελεστές Ανασυνδυασμού για Αναπαράσταση Ακεραίων 31
2.8.4 Τελεστές Ανασυνδυασμού για Αναπαράσταση Μετάθεσης 31

2.9 Μετάλλαξη 33
2.9.1 Τελεστές Μετάλλαξης για Δυαδική Αναπαράσταση . . . 34
2.9.2 Τελεστές Μετάλλαξης για Αναπαράσταση Κινητής Υποδιαστολής 35
2.9.3 Τελεστές Μετάλλαξης για Αναπαράσταση Ακεραίων . . 35
2.9.4 Τελεστές Μετάλλαξης για Αναπαράσταση Μετάθεσης . 35
2.10 Επιλογή 36
2.10.1 Επιλογή με αναλογία της ποιότητας του ατόμου 36
2.10.2 Rank Roulette Wheel 39
2.10.3 Tournament Selection 39
2.11 Φυσική Επιλογή 40
2.11.1 Επιλογή ως προς την ηλικία 40
2.11.2 Επιλογή ως προς την τιμή ποιότητας 41
2.12 Συνθήκη Τερματισμού 41

3 Γραφική Αναπαράσταση Γενετικών Αλγόριθμων 43
3.1 Γραφική απεικόνιση της συνολικής πορείας του ΓΑ 44
3.1.1 Διάγραμμα καλύτερης και μέσης τιμής του ΓΑ 44
3.1.2 Απεικόνιση τιμών του καλύτερου ατόμου 44
3.2 Γραφική απεικόνιση της κατάστασης του ΓΑ 46
3.2.1 Τιμές ποιότητας όλων των ατόμων 46
3.2.2 Τιμές των ατόμων 47

4 Περιγραφή Απαιτήσεων Περιβάλλοντος 51

5 Υλοποίηση 55
5.1 Σχεδίαση 56
5.1.1 Περιβάλλον ανάπτυξης και βοηθητικές βιβλιοθήκες . . . 56
5.1.2 Δυνατότητες 60
5.2 Η βιβλιοθήκη GAJLib 64
5.2.1 Χρωμόσωμα 64
5.2.2 Αναπαράσταση 64
5.2.3 Σύστημα Ρυθμίσεων 65
5.2.4 Αρχείο Evolution History 67
5.3 Η εφαρμογή GAJVis 70
5.3.1 Μηχανισμός διαχείρισης των Επεκτάσεων 71
5.3.2 Ρύθμιση και Εκτέλεση Πειραμάτων 71
5.3.3 Γραφική Αναπαράσταση Δεδομένων 73

6 Συμπεράσματα και Περαιτέρω Ανάπτυξη 75

Α΄ Οδηγίες Εγκατάστασης **77**
Α΄.1 Μεταγλώττιση πηγαίου κώδικα και εγκατάσταση 78
Α΄.2 Εγκατάσταση μεταγλωττισμένου κώδικα 79
Α΄.3 Εγκατάσταση Επεκτάσεων 79

Β΄ Οδηγίες χρήσης **81**
Β΄.1 Εκτέλεση Γενετικών Αλγορίθμων 81
Β΄.2 Διαγράμματα στο GAJVis 82

Γ΄ Παραδείγματα Χρήσης **85**
Γ΄.1 Παραδείγματα χρήσης της βιβλιοθήκης GAJLib 85
Γ΄.1.1 Συναρτήσεις 85
Γ΄.1.2 Παράδειγμα ελαχιστοποίησης συνάρτησης 86
Γ΄.1.3 Το πρόβλημα με τις σφηκοφωλιές 91

Βιβλιογραφία **97**

Κατάλογος Σχημάτων

2.1 Πρόβλημα βελτιστοποίησης. 18
2.2 Πρόβλημα μοντελοποίησης ή αναγνώρισης συστήματος 19
2.3 Πρόβλημα προσομοίωσης 19
2.4 Εξελικτικός κύκλος 21
2.5 Χρωμόσωμα 25
2.6 Χρωμόσωμα δυαδικής αναπαράστασης 26
2.7 Χρωμόσωμα με αναπαράσταση ακεραίων 26
2.8 Χρωμόσωμα με αναπαράσταση κινητής υποδιαστολής 27
2.9 Χρωμόσωμα με αναπαράσταση μετάθεσης 27
2.10 Τελεστής single point ανασυνδυασμού 29
2.11 Τελεστής N Point ανασυνδυασμού, με $n = 2$. 29
2.12 Τελεστής uniform ανασυνδυασμού 30
2.13 Τελεστές ανασυνδυασμού αναπαράστασης κινητής υποδιαστολής 32
2.14 PMX πρώτο στάδιο 33
2.15 PMX δεύτερο στάδιο 33
2.16 PMX τελευταίο στάδιο 34
2.17 Τελεστής Cycle Recombination 34
2.18 Τελεστής μετάλλαξης δυαδικής αναπαράστασης 34
2.19 Inversion Mutation 36
2.20 Swap Mutation 36
2.21 Cost Roulette Wheel και Stochastic Universal Sampling 38

3.1 Δισδιάστατο διάγραμμα μέσης και καλύτερης τιμής ποιότητας του πληθυσμού για όλες τις γενιές του ΓΑ. 45
3.2 Δισδιάστατο διάγραμμα εικόνας 46
3.3 τρισδιάστατο διάγραμμα σημείων/επιφάνειας 47
3.4 Δισδιάστατο διάγραμμα με τις τιμές ποιότητας όλων των ατόμων ενός πληθυσμού 47
3.5 Δισδιάστατο διάγραμμα εικόνας για όλες τις τιμές των γονιδίων 48
3.6 Δισδιάστατο διάγραμμα καλύτερων, μέσων γονιδίων 48
3.7 τρισδιάστατο διάγραμμα σημείων/επιφάνειας από έναν ολόκληρο πληθυσμό 49

5.1 Διάγραμμα Τάξεων, χρωμόσωμα και γονίδια. 65
5.2 Διάγραμμα Τάξεων, αναπαραστάσεις. 66
5.3 Ρύθμιση και εκτέλεση. 67
5.4 Αρχείο Evolution History. 68
5.5 Οι δύο τύποι Evolution History αρχείων. 69
5.6 Genetic Algorithm Java Visualization application 71
5.7 Wizard ρυθμίσεων 72
5.8 Διάγραμμα πραγματικού χρόνου 72
5.9 Γραφική αναπαράσταση δεδομένων 74

Β′.1 Παράθυρο επιλογής τύπου αρχείου ΕΗ 82
Β′.2 Παράθυρο wizard 83
Β′.3 Η εφαρμογή GAJVis κατά την εκτέλεση ενός ΓΑ 84
Β′.4 Επιλογή υποσυνόλου από το σύνολο των γενεών. 84
Β′.5 Διαχείριση χρωμάτων 84

Γ′.1 Διαγραμμα μέσης και καλήτερης τιμής, συνάρτηση Rastrigin . . 89
Γ′.2 Διάγραμμα μέσης και καλύτερης τιμής, πρόβλημα με τις σφηκοφωλιές 96

Περίληψη

Η Εξελικτική Υπολογιστική (ΕΥ) είναι ένας χώρος της Επιστήμης των Υπολογιστών, ο οποίος μιμείται βασικές αρχές των διαδικασιών της βιολογικής Εξέλιξης. Ο χώρος αυτός, αποτελείται από ένα σύνολο εξελικτικών υπολογιστικών μοντέλων, τα οποία αναφέρονται με τον όρο Εξελικτικοί Αλγόριθμοι (ΕΑ). Οι αλγόριθμοι αυτοί εφαρμόζονται ως μία γενετική μέθοδο επίλυσης προβλημάτων, που εξερευνεί το χώρο λύσεων του προβλήματος, έχοντας ως στόχο τον εντοπισμό καλών λύσεων. Το μοντέλο στο οποίο επικεντρωνόμαστε είναι οι Γενετικοί Αλγόριθμοι, οι οποίοι αποτελούν την περισσότερο διαδεδομένη μορφή ΕΑ. Στη σημερινή τους μορφή, διαθέτουν ένα πλήθος από διαφορετικές δομές δεδομένων που αναπαριστούν τις πιθανές λύσεις του προβλήματος, καθώς και ένα μεγάλο αριθμό από διαφορετικούς γενετικούς τελεστές. Η πολύπλοκη λειτουργία τους, σε συνδυασμό με την πληθώρα των παραμέτρων που απαιτούν, καθιστά δύσκολη την κατανόηση του τρόπου αναζήτησης. Στόχος της πτυχιακής εργασίας, είναι η κατασκευή ενός εύχρηστου περιβάλλοντος για την επίλυση προβλημάτων με ΓΑ, καθώς και η γραφική μελέτη των δεδομένων που παράγονται από την επαναλαμβανόμενη διαδικασία της εξέλιξης.

Abstract

The Evolutionary Computation (EC) is a field of Computer Science that imitates the basic principles of the biological evolutionary process. The field consists of a variety of evolutionary computational models, referred as Evolutionary Algorithms (EA). EA are typically applied as a genetic problem solving method, searching a problem space in order to locate good solutions. We focus on the Genetic Algorithms (GA) model which constitutes the most common variation of EA. Nowadays, GA employs both a number of different data structures which represent the problem's potential solutions and a great number of different genetic operators. Their complexity in combination with the numerous required parameters, makes them difficult to understand. The aim of this dissertation is the development of an easy to use framework for solving problems using GA, and also for the visualization of the produced data from the iterative process of evolution.

1

Εισαγωγή

Η Εξελικτική Υπολογιστική, είναι ένας χώρος της Επιστήμης των Υπολογιστών που ξεκίνησε από τα μέσα του $20^{ου}$ αιώνα και παρουσιάζει έντονο ερευνητικό ενδιαφέρον. Γενικότερα ο χώρος αυτός αποτελείται από ένα σύνολο εξελικτικών υπολογιστικών μοντέλων, τα οποία αναφέρονται με τον όρο Εξελικτικοί Αλγόριθμοι (ΕΑ). Το αντικείμενο με το οποίο ασχολούνται οι ΕΑ, είναι η επίλυση δύσκολων προβλημάτων αναζήτησης και βελτιστοποίησης. Αυτό όμως που τους κάνει να ξεχωρίζουν από άλλους αλγόριθμους, είναι ο τρόπος με τον οποίο λειτουργούν, ο οποίος είναι εμπνευσμένος από τις αρχές της βιολογικής εξέλιξης. Πιο συγκεκριμένα αποτελούν στοχαστικές μεθόδους αναζήτησης οι οποίες διερευνούν πολύπλοκους χώρους, προσομοιώνοντας τεχνικές φυσικής εξέλιξης (Holland 1975).

Αντίθετα με άλλες μεθόδους αναζήτησης, οι ΕΑ διατηρούν ένα πλήθος από δομές δεδομένων, με τις οποίες αναπαριστούν τις πιθανές λύσεις του προβλήματος. Η διερεύνηση στο χώρο πραγματοποιείται με την εκμετάλλευση των καλών λύσεων, με στόχο τον εντοπισμό άλλων καλύτερων. Με τον τρόπο αυτό, οι ΕΑ εστιάζουν την έρευνα προς τις καταλληλότερες περιοχές του χώρου.

Παρόλα αυτά, η πολυπλοκότητα των ΕΑ και η πληθώρα των διαφορετικών συνδυασμών εξελικτικών μεθόδων, καθιστά δύσκολη την κατανόηση της συμπεριφοράς του μηχανισμού αναζήτησης. Ο χρήστης βρίσκεται αντιμέτωπος μπροστά σε ένα μεγάλο αριθμό παραμέτρων και διαφορετικών επιλογών, που έχουν να κάνουν με την εστίαση των πιθανών λύσεων, με τον τύπο των δομών δεδομένων και με τον τρόπο με τον οποίο διαταράσσονται οι τιμές τους.

Αντικείμενο αυτής της πτυχιακής εργασίας, είναι η κατασκευή ενός περιβάλλοντος για την ανάπτυξη και την γραφική αναπαράσταση των Γενετικών Αλγόριθμων (ΓΑ), που αποτελεί την περισσότερο διαδεδομένη μορφή των ΕΑ

και αναπτύχθηκε το 1975 από τον John Holland. Το περιβάλλον της εργασίας χωρίζεται σε δύο τμήματα, μία βιβλιοθήκη ανάπτυξης ΓΑ και μία εφαρμογή για την γραφική μελέτη τους. Ο στόχος της βιβλιοθήκης είναι η παροχή ενός ολοκληρωμένου πλαισίου επίλυσης προβλημάτων με ΓΑ. Αντίθετα με άλλες υλοποιήσεις βιβλιοθηκών, η συγκεκριμένη προσπαθεί να προσφέρει πολλές δυνατότητες και ευκολίες στον χρήστη, ενώ ταυτόχρονα παραμένει επεκτάσιμη σε όλα τα επίπεδά της. Επιπλέον σημαντικό ρόλο στην σχεδίαση και την υλοποίηση της βιβλιοθήκης κατέχουν οι παράγοντες της δυνατότητας εκτέλεσης σε διαφορετικές αρχιτεκτονικές Η/Υ και της ταχύτητας. Για τον λόγο αυτό η βιβλιοθήκη αναπτύχθηκε με την γλώσσα προγραμματισμού Java ™ και ο πυρήνας της βιβλιοθήκης χρησιμοποιεί την βιβλιοθήκη Javolution.

Όσον αφορά το δεύτερο τμήμα, αναπτύχθηκε μία εφαρμογή γραφικής αναπαράστασης, η οποία προσφέρει στον χρήστη ένα εύχρηστο γραφικό περιβάλλον για την μελέτη και την εφαρμογή των ΓΑ. Έτσι ώστε να διευκολύνεται η έρευνα και η κατανόηση της συμπεριφοράς του αλγορίθμου. Η εφαρμογή εκμεταλλεύεται το γεγονός ότι κατά την επαναληπτική διαδικασία της εξέλιξης, είναι δυνατό να εξαχθεί μεγάλος όγκος πληροφοριών, μέσω των οποίων μπορούμε να μελετήσουμε την συνολική πορεία της εξέλιξης καθώς και την μεμονωμένη κατάσταση του ΓΑ. Για την επίτευξη όλων αυτών, η εφαρμογή βασίζεται στη χρήση των βιβλιοθηκών VisAD και JChart2D για την δισδιάστατη και τρισδιάστατη γραφική αναπαράσταση των δεδομένων.

Στο δεύτερο κεφάλαιο παρουσιάζεται η θεωρία των ΕΑ όπου επικεντρωνόμαστε στους ΓΑ. Αναφέρονται οι πιο συνηθισμένες αναπαραστάσεις τους, οι μέθοδοι επιλογής, τα χαρακτηριστικά και οι δυνατότητες των τελεστών τους. Ουσιαστικά το κεφάλαιο αυτό αποτελεί το θεωρητικό υπόβαθρο της βιβλιοθήκης.

Στο τρίτο κεφάλαιο παρουσιάζεται ένα σύνολο από τεχνικές γραφικής αναπαράστασης των ΓΑ. Στόχος τους είναι η μελέτη και κατανόηση της συμπεριφοράς του αλγόριθμου, με κύριο χαρακτηριστικό τη δυνατότητα εφαρμογής τους στο σύνολο των αναπαραστάσεων που αναφέρονται στο δεύτερο κεφάλαιο.

Στο τέταρτο κεφάλαιο περιγράφονται οι απαιτήσεις του περιβάλλοντος, που θα πρέπει να πληρούν η βιβλιοθήκη και η εφαρμογή. Οι απαιτήσεις αυτές αφορούν την δομή και την υποστήριξη των διαφορετικών στοιχείων που συνθέτουν έναν ΓΑ, την επεκτασιμότητα και τις γραφικές δυνατότητες του περιβάλλοντος.

Στο πέμπτο κεφάλαιο παρουσιάζεται η σχεδίαση και η υλοποίηση του περιβάλλοντος. Αναφέρονται τα χαρακτηριστικά και οι δυνατότητες της βιβλιοθήκης και της εφαρμογής, καθώς και ο τρόπος λειτουργίας τους.

Στο έκτο κεφάλαιο αναφέρονται τα συμπεράσματα από την υλοποίηση του περιβάλλοντος καθώς και τα σημεία στα οποία μπορεί να γίνει περαιτέρω έρευνα. Διατυπώνονται διάφορες προτάσεις βελτίωσης και προσθήκης νέων δυνατοτήτων της παρούσας σχεδίασης.

Ολοκληρώνοντας την εργασία, στα παρατήματα Α, Β και Γ που ακολουθούν, δίνονται αντίστοιχα οι οδηγίες εγκατάστασης και χρήσης, καθώς και η παρουσίαση μερικών παραδειγμάτων χρήσης. Διατυπώνεται η μεθοδολογία επίλυσης των συναντήσεων Ackley, Rastrigin και Sphere, καθώς και ένα παράδειγμα προβλήματος αναζήτησης.

2

Εξελικτικοί Αλγόριθμοι

2.1 Γιατί οι Εξελικτικοί Αλγόριθμοι

Από τα μέσα του $20^{ου}$ αιώνα ένα από τα θέματα που παρουσιάζουν έντονο ενδιαφέρον στις επιστήμες των μαθηματικών και των υπολογιστών, είναι η ανάπτυξη αλγόριθμων για την αυτόματη επίλυση προβλημάτων. Οι επιστήμονες παρατηρούν τις λύσεις που δίνει η φύση και τις χρησιμοποιούν ως πηγή έμπνευσης για τις έρευνές τους. Το ανθρώπινο μυαλό και η διαδικασία της βιολογικής εξέλιξης, αποτελούν τα βασικότερα σημεία στα οποία επικεντρώνονται οι μελέτες τους. Από την μελέτη του ανθρώπινου μυαλού οδηγούμαστε στον τομέα των τεχνητών νευρωνικών δικτύων, ενώ από την μελέτη της βιολογικής εξέλιξης των ειδών στον τομέα των Εξελικτικών Αλγόριθμων (ΕΑ).

Η ραγδαία εφαρμογή των ηλεκτρονικών υπολογιστών σε όλους τους επιστημονικούς και εφαρμοσμένους τομείς, έχει οδηγήσει στην ανάγκη για την αυτόματη επίλυση των προβλημάτων, με τα οποία ασχολούνται. Ο ρυθμός έρευνας και ανάπτυξης πολλές φορές δεν μπορεί να αντεπεξέλθει στις απαιτούμενες ανάγκες. Αυτό έχει ως αποτέλεσμα τη μείωση του διαθέσιμου χρόνου για την ανάλυση και την σχεδίαση του απαιτούμενου αλγόριθμου. Παράλληλα, η πολυπλοκότητα των προβλημάτων προς επίλυση ολοένα και αυξάνεται. Με πολύ συχνό φαινόμενο να χρειάζονται αναζητήσεις λύσεων μέσα σ' ένα πολύ μεγάλο χώρο λύσεων, ο οποίος μπορεί να μην είναι σταθερός και να μεταβάλλεται. Σ' αυτές τις περιπτώσεις απαιτείται δυνατότητα προσαρμογής του αλγόριθμου.

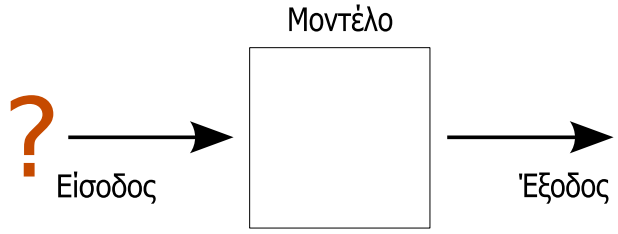
Με βάση τα προηγούμενα οδηγούμαστε στην ανάγκη ανάπτυξης αλγόριθμων οι οποίοι δεν θα πρέπει να είναι εξειδικευμένοι σε συγκεκριμένα προβλήματα, και συγχρόνως να μην υστερούν σε απόδοση. Δηλαδή να εφαρμόζονται σε διαφορετικού τύπου προβλήματα χωρίς επίπονες τροποποιήσεις και ταυτόχρονα

να δίνουν καλές λύσεις (όχι απαραίτητα τη βέλτιστη) μέσα σε ένα λογικό χρονικό περιθώριο. Οι ΕΑ είναι ένα σύνολο από αλγόριθμους, οι οποίοι βασίζονται στις ίδιες σχεδιαστικές αρχές και διαθέτουν αυτές τις δυνατότητες.

Επιπλέον υπάρχει και ένα τρίτο κίνητρο το οποίο πηγάζει πίσω από κάθε επιστήμη. Οι φυσικές διαδικασίες της εξέλιξης αποτελούν αντικείμενο επιστημονικής έρευνας και μελέτης με στόχο την πλήρη κατανόησή τους. Από αυτήν την οπτική γωνία, οι ΕΑ μπορούν να αποτελέσουν ένα διαφορετικό τρόπο εφαρμογής πειραμάτων σε σχέση με τη παραδοσιακή βιολογία. Η διαδικασία της εξέλιξης μπορεί να προσομοιωθεί και να επαναληφθεί υπό διαφορετικές συνθήκες σ΄ έναν ή περισσότερους υπολογιστές όπου εκατομμύρια γενεές μπορούν να αναπτυχθούν μέσα σε σχετικά μικρό χρόνο.

Παρακάτω θα υποδείξουμε την αποτελεσματικότητα της εξέλιξης στην προσέγγιση της αυτόματης επίλυσης προβλημάτων που ανήκουν σε διαφορετικούς τομείς. Πρώτα απ΄ όλα σε ένα σύστημα παρατηρούμε τρία βασικά στοιχεία, τις εισόδους, τις εξόδους και ένα εσωτερικό μοντέλο που ενώνει τα δύο προηγούμενα στοιχεία. Όταν λέμε ότι γνωρίζουμε το μοντέλο, εννοούμε ότι κατέχουμε την γνώση του τρόπου λειτουργίας του συστήματος. Στην περίπτωση αυτή είναι δυνατό να υπολογίσουμε την έξοδο για οποιαδήποτε είσοδο. Έτσι μπορούμε να διαχωρίσουμε τρεις τύπους προβλημάτων με βάση τα στοιχεία που γνωρίζουμε.

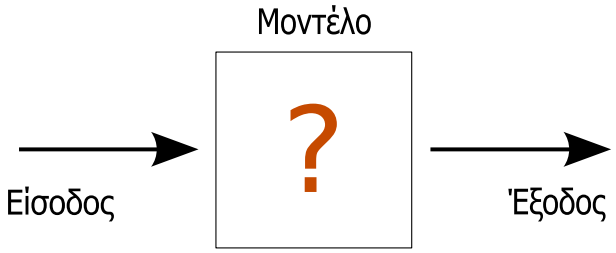
1. Σ΄ ένα πρόβλημα βελτιστοποίησης (σχήμα 2.1) γνωρίζουμε το μοντέλο και την έξοδο. Αυτό το οποίο αναζητούμε είναι η είσοδος που οδηγεί στην επιθυμητή έξοδο. Για παράδειγμα μπορούμε να αναφέρουμε το γνωστό πρόβλημα του περιοδεύοντος πωλητή, όπου αναζητούμε την συντομότερη διαδρομή μεταξύ ενός πλήθους πόλεων. Γνωρίζουμε το μοντέλο, δηλαδή τις αποστάσεις για κάθε διαδρομή. Κάθε διαδρομή αποτελεί την είσοδο και το συνολικό μήκος της περιοδείας αποτελεί την έξοδο.



Σχήμα 2.1: *Πρόβλημα βελτιστοποίησης.*

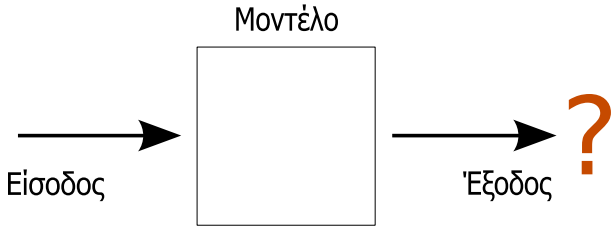
2. Σ΄ ένα πρόβλημα μοντελοποίησης ή αναγνώρισης συστήματος, οι είσοδοι και οι έξοδοι είναι γνωστοί (σχήμα 2.2). Για παράδειγμα στον κεντρικό πίνακα του χρηματιστηρίου, παρατηρούμε τον γενικό δείκτη ως έξοδο και ως είσοδο διάφορους άλλους δείκτες όπως η τιμή του χρυσού, η διαφορά

δολαρίου ευρό κ.τ.λ. Στόχος μας είναι να βρούμε ένα μοντέλο το οποίο να συνδέει τις εισόδους με την έξοδο. Συνεπώς, με βάση τα γνωστά δεδομένα του παρελθόντος μπορούμε να προβλέψουμε την πορεία του δείκτη για τα νέα δεδομένα.



Σχήμα 2.2: *Πρόβλημα μοντελοποίησης ή αναγνώρισης συστήματος*

3. Τέλος, σ΄ ένα πρόβλημα προσομοίωσης γνωρίζουμε το μοντέλο και τις εισόδους, αλλά χρειαζόμαστε να υπολογίσουμε τις εξόδους (σχήμα 2.3). Ως αντιπροσωπευτικό παράδειγμα μπορούμε να αναφέρουμε ένα ηλεκτρονικό κύκλωμα που λειτουργεί ως φίλτρο αποκοπής χαμηλών συχνοτήτων από ένα σήμα. Το μοντέλο αποτελείται από τις εξισώσεις που περιγράφουν τη λειτουργία του συγκεκριμένου κυκλώματος. Για οποιαδήποτε είσοδο, το μοντέλο μπορεί να υπολογίσει την έξοδο. Κάνοντας χρήση αυτού του μοντέλου μπορούμε να συγκρίνουμε διάφορα σχέδια κυκλωμάτων χωρίς να τα υλοποιήσουμε και να τα μετρήσουμε, δηλαδή με το τρόπο αυτό έχουμε μία ταχύτερη και φθηνότερη διαδικασία.



Σχήμα 2.3: *Πρόβλημα προσομοίωσης*

2.2 Τι είναι οι Εξελικτικοί Αλγόριθμοι

Οι ΕΑ είναι αλγόριθμοι εμπνευσμένοι από τις αρχές της Δαρβίνειας θεωρίας της Εξέλιξης. Είναι άμεσοι, αλγόριθμοι αναζήτησης και βελτιστοποίησης με ευρύ φάσμα εφαρμογών. Αποτελούν μία στοχαστική και επαναληπτική διαδικασία,

κατά την οποία διατηρούν ένα πλήθος από προφανείς λύσεις του προβλήματος. Επιπλέον έχουν μία διαδικασία επιλογής βασισμένη στην ποιότητα των λύσεων και σε «κάποιουσ»γενετικούς τελεστές. Υπάρχουν αρκετές υλοποιήσεις ΕΑ, οι οποίες αν και διαφέρουν μεταξύ τους, βασίζονται στις ίδιες αρχές. Οι κυριότερες κατηγορίες είναι οι εξής:

1. **Εξελικτικές Στρατηγικές** (Evolution Strategies), προτάθηκαν από τους Rechenberg και Schwefel. Αποτελούν αλγόριθμους οι οποίοι μιμούνται τις αρχές τις φυσικής επιλογής για την επίλυση προβλημάτων βελτιστοποίησης παραμέτρων. Λειτουργούν με διανύσματα πραγματικών αριθμών που εκφράζουν τις προφανείς λύσεις.

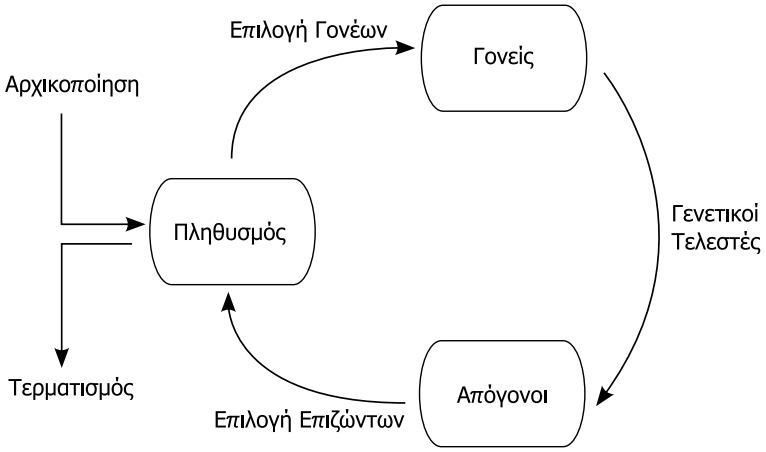
2. **Γενετικός Προγραμματισμός** (Genetic Programming), προτάθηκε από τον Koza. Χρησιμοποιείται για την αναζήτηση του καλύτερου προγράμματος υπολογιστή το οποίο επιλύει ένα συγκεκριμένο πρόβλημα.

3. **Εξελικτικός Προγραμματισμός** (Evolutionary Programming), προτάθηκε από τον Fogel, είναι μία τεχνική αναζήτησης στο χώρο μικρών μηχανών πεπερασμένων καταστάσεων (finite-state machines).

4. **Γενετικοί Αλγόριθμοι** (Genetic Algorithms), προτάθηκε από τον Holland. Αποτελούν τον δημοφιλέστερο τύπο ΕΑ και χρησιμοποιούνται κυρίως για την επίλυση προβλημάτων βελτιστοποίησης παραμέτρων. Οι προφανείς λύσεις που διατηρούν έχουν την δομή μίας σειράς αριθμών (παραδοσιακά δυαδικών) και επάνω σ΄ αυτές εφαρμόζονται οι γενετικοί τελεστές του ανασυνδυασμού και της μετάλλαξης.

Οι Εξελικτικοί Αλγόριθμοι ενεργούν σε ένα σύνολο από άτομα (individuals), το οποίο ονομάζεται πληθυσμός (population) $P(t) = \{x_1^t, x_2^t ... x_n^t\}$. Κάθε άτομο x είναι μία δομή η οποία αναπαριστά μία πιθανή λύση του προβλήματος, δηλαδή ένα σημείο στο χώρο των πιθανών λύσεων. Παράλληλα το άτομο περιέχει και κάποια γνώση για τους κανόνες του περιβάλλοντος του προβλήματος. Επίσης σε κάθε ένα άτομο του πληθυσμού αντιστοιχίζεται μία τιμή που εκφράζει το μέτρο της ποιότητας (fitness value) που διαθέτει το άτομο στο συγκεκριμένο περιβάλλον. Το περιβάλλον αυτό τυποποιείται και κωδικοποιείται μέσω της συνάρτησης ποιότητας (fitness function). Η δημιουργία μίας νέας γενιάς (generation), προέρχεται από την εφαρμογή ενός κανόνα επιλογής (selection) και κάποιων τελεστών (operators), όπως ο ανασυνδυασμός (recombination) και η μετάλλαξη (mutation). Κατά την επιλογή τα άτομα του πληθυσμού με υψηλή τιμή ποιότητας επιλέγονται για να εφαρμοστούν επάνω στη δομή τους οι τελεστές ανασυνδυασμού και μετάλλαξης. Η διερεύνηση στον χώρο πραγματοποιείται από την εφαρμογή των τελεστών, η οποία διαταράσσει τη δομή των ατόμων ώστε να προκύψουν νέα άτομα, συνεπώς νέα σημεία στον χώρο.

```
procedure EA
begin
    t=0
    initialize P(t)
    evaluate P(t)
        while(termination condition is not satisfied) do
        begin
            t=t+1
            select P(t) from P(t-1)
            alter P(t)
            evaluate P(t)
        end
end
```



Σχήμα 2.4: *Εξελικτικός κύκλος*

2.2.1 Δομή ενός Εξελικτικού Αλγόριθμου

Ανάλογα με τις απαιτήσεις του προβλήματος, οι ΕΑ υλοποιούνται με διαφορετικό τρόπο. Πιο συγκεκριμένα, μπορούν να διαφέρουν στη δομή δεδομένων του ατόμου, στους τελεστές, στην αρχικοποίηση του πληθυσμού (να είναι δηλαδή είτε τυχαία, είτε να προέρχεται από δεδομένα), στις παραμέτρους του αλγόριθμου (όπως μέγεθος πληθυσμού, πιθανότητες εφαρμογής τελεστών κ.τ.λ) κ.α. Παρόλα αυτά, όλοι οι ΕΑ μοιράζονται μία κοινή αρχή: ο πληθυσμός από τα άτομα υφίσταται κάποιους μετασχηματισμούς κατά την διάρκεια της διαδικασίας της εξέλιξης. Η πίεση που δημιουργείται από το περιβάλλον προκαλεί τη φυσική επιλογή με αποτέλεσμα την αύξηση της ποιότητας του πληθυσμού.

Κατά την διαδικασία της εξέλιξης υπάρχουν δύο δυνάμεις οι οποίες διαμορφώνουν το σύστημα της εξέλιξης:

1. Οι γενετικοί τελεστές (ανασυνδυασμός και μετάλλαξη), με τους οποίους δημιουργείται η απαραίτητη ανομοιομορφία στον πληθυσμό.

2. Και η επιλογή, η οποία πιέζει προς την κατεύθυνση ομοιογένειας του πληθυσμού.

Όλοι οι ΕΑ διαθέτουν ένα πλήθος στοιχείων, διαδικασίες και τελεστές τα οποία πρέπει να οριστούν, και είναι τα εξής:

1. Αναπαράσταση (Representation), ορίζει την δομή δεδομένων και την κωδικοποίηση της πληροφορίας του ατόμου.

2. Συνάρτηση ποιότητας (Fitness, Objective function), με την βοήθεια αυτής εκτιμάται η ποιότητα του κάθε ατόμου στον πληθυσμό.

3. Πληθυσμός (Population), είναι η δομή δεδομένων που περιέχει ένα σύνολο από άτομα της ίδιας γενίας.

4. Μηχανισμός επιλογής γονέων (Parent Selection Mechanism), για την επιλογή τον γονέων που θα συμμετέχουν στη διαδικασία της δημιουργίας των απογόνων.

5. Γενετικοί τελεστές (Genetic Operators), οι οποίοι διαφοροποιούν τα άτομα και πραγματοποιείται η αναζήτηση.

6. Φυσική επιλογή (Natural Selection), για την επιλογή τον ατόμων που θα συμμετέχουν στην επόμενη γενιά.

7. Αρχικοποίηση (Initialization) του πληθυσμού.

8. Συνθήκη τερματισμού (Termination Condition) του ΕΑ.

2.3 Γενετικοι Αλγόριθμοι

Οι Γενετικοί Αλγόριθμοι (ΓΑ) αναπτύχθηκαν από τον John Holland το 1975 και έγιναν γνωστοί από τον μαθητή του τον David Goldberg [1] το 1989. Αποτελούν την γνωστότερη υλοποίηση των Εξελικτικών Αλγόριθμων. Παραδοσιακά χρησιμοποιούσαν ως δομή δεδομένων για την αναπαράσταση των μεταβλητών συμβολοσειρές από δυαδικά ψηφία. Παρόλα αυτά πολλές εφαρμογές των ΓΑ έχουν επικεντρωθεί σε διαφορετικές αναπαραστάσεις, όπως διανύσματα αριθμών κινητής υποδιαστολής, γραφήματα, εκφράσεις Lisp κ.α.

Οι ΓΑ διαθέτουν όλα τα χαρακτηριστικά των ΕΑ, εφόσον ανήκουν στην ίδια οικογένεια. Μετά την αρχικοποίηση του πληθυσμού, επιλέγονται οι γονείς πάνω στους οποίους εφαρμόζονται οι γενετικοί τελεστές, έτσι ώστε να παραχθούν νέα άτομα, οι απόγονοι. Τα άτομα με την καλύτερη τιμή ποιότητας (fitness, objective value) επιλέγονται για την επόμενη γενιά. Στους ΓΑ οι γενετικοί τελεστές είναι δύο, ο ανασυνδυασμός ή διασταύρωση (recombination ή crossover) και η μετάλλαξη (mutation).

Το θεωρητικό υπόβαθρο των ΓΑ είναι διαφορετικό απ΄ αυτό των Εξελικτικών Στρατηγικών (ΕΣ) και του Εξελικτικού Προγραμματισμού (ΕΠ). Ο Holland επικεντρώνεται στη Θεωρία των Σχημάτων (Schema Theorem), όπου ένα σχήμα θεωρείται ως ένα σύνολο από άτομα στα οποία η δομή τους ταιριάζει με ένα πρότυπο.

Η αναζήτηση πραγματοποιείται από τους γενετικούς τελεστές, με κύριο τελεστή τον ανασυνδυασμό. Όσον αφορά τον τελεστή της μετάλλαξης, συνηθίζεται να δίνεται ελάχιστη πιθανότητα εφαρμογής, το αντίθετο δηλαδή τις ΕΣ και τον ΕΠ. Αυτή η άποψη έχει αναθεωρηθεί και το ενδιαφέρον προς τον τελεστή της μετάλλαξης έχει αυξηθεί. Έχει αποδειχθεί πειραματικά ότι η μετάλλαξη είναι ένας αρκετά αποτελεσματικός τελεστής (Schaffer and Eshelman 1991).

Παρακάτω αναφέρονται τα πλεονεκτήματα και τα μειονεκτήματα που παρουσιάζουν οι ΓΑ:

- Πλεονεκτήματα
    1. Βελτιστοποιεί με μεταβλητές που είναι συνεχείς ή διακριτές.
    2. Εφαρμόζει αναζήτηση σε πολλά σημεία του χώρου ταυτόχρονα.
    3. Είναι αποτελεσματικός σε προβλήματα με μεγάλο πλήθος μεταβλητών.
    4. Μπορεί να εκμεταλλευτεί παράλληλα συστήματα.
    5. Παρέχει όχι μόνο μία, αλλά ένα πλήθος από προφανείς λύσεις.
    6. Λειτουργεί με αριθμητικά και πειραματικά δεδομένα ή με αναλυτικές συναρτήσεις.

7. Μπορούν να αλληλεπιδρούν με τον χρήστη κατά την εκτέλεση.
8. Εύκολη συνεργασία με άλλες μεθόδους.

- Μειονεκτήματα

1. Δεν επιστρέφουν πάντα τη βέλτιστη λύση μέσα σε πεπερασμένο χρονικό διάστημα.
2. Χρειάζονται ρυθμίσεις σε πολλές παραμέτρους ώστε να έχουν ικανοποιητική απόδοση.
3. Απαιτούν μεγάλη υπολογιστική ισχύ.
4. Δεν είναι καλύτερη μέθοδος για επίλυση προβλημάτων στα οποία έχουν αναπτυχθεί εξειδικευμένοι υπολογιστικοί αλγόριθμοι. Οι εξειδικευμένοι αλγόριθμοι είναι ταχύτεροι και βρίσκουν την βέλτιστη λύση.

2.4 Αναπαράσταση

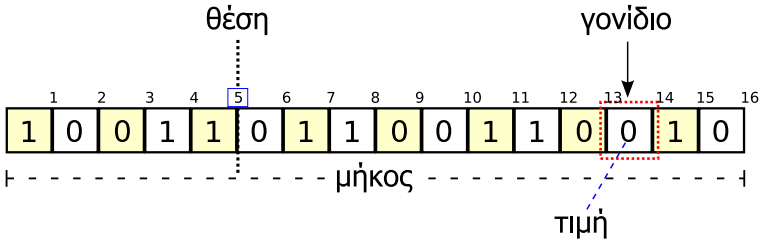
Πρωταρχικό στάδιο υλοποίησης ενός ΓΑ είναι ο ορισμός της αναπαράστασης (representation), έτσι ώστε να διασυνδέσουμε τον «πραγματικό κόσμο»του προβλήματος, με τον «κόσμο του ΓΑ». Τα αντικείμενα που αναπαριστούν τις πιθανές λύσεις στον «πραγματικό κόσμο»αποτελούν τον φαινότυπο (phenotype), ενώ η κωδικοποίησή τους, δηλαδή τα άτομα του πληθυσμού αποτελούν τον γονότυπο (genotype). Όμως πολύ συχνά η αντιστοίχιση αυτή μπορεί να μην είναι απαραίτητη. Είναι πολύ πιθανό η δόμή των ατόμων του πληθυσμού να αποτελεί ταυτόχρονα τον φαινότυπο του προβλήματος. Συνεπώς η αναζήτηση να πραγματοποιείται στο χώρο του φαινοτύπου.

Στους ΓΑ τα άτομα του πληθυσμού ονομάζονται επίσης χρωμοσώματα (chromosome). Κάθε χρωμόσωμα (σχήμα 2.5) είναι μία δομή δεδομένων που περιέχει ένα σύνολο μεταβλητών. Οι μεταβλητές αυτές ονομάζονται γονίδια (genes) και περιέχουν τις τιμές (alleles).

	Πραγματικός κόσμος	**ΓΑ**
Χώρος πιθανών λύσεων	Χώρος φαινοτύπου	Χώρος γονότυπου
Τιμή ενός στοιχείου	Τιμή	Allele
Θέση στοιχείου	Θέση	Locus, gene

Είναι σημαντικό να τονιστεί πως ο όρος αναπαράσταση χρησιμοποιείται με δύο ερμηνείες. Άλλες φορές εκφράζει την αντιστοίχιση μεταξύ φαινότυπου και γονότυπου, δηλαδή την κωδικοποίηση. Και άλλες φορές εκφράζει τη δομή δεδομένων του χρωμοσώματος.

Η σωστή επιλογή της αναπαράστασης για ένα συγκεκριμένο πρόβλημα, θεωρείται κρίσιμη και αποτελεί ένα από τα δυσκολότερα τμήματα σχεδίασης του ΓΑ. Σύμφωνα με την δομή και τους κανόνες που ορίζει η αναπαράσταση ορίζονται οι γενετικοί τελεστές που θα εφαρμοστούν. Οι τελεστές πρέπει να υποστηρίζουν την χρησιμοποιούμενη αναπαράσταση, δηλαδή να μπορούν να λειτουργούν και να παράγουν συμβατές λύσεις. Οι αναπαραστάσεις που παρουσιάζονται στις επόμενες παραγράφους αποτελούν τις περισσότερο γνωστές και συχνότερα χρησιμοποιούμενες. Παρόλα αυτά, σε πραγματικές εφαρμογές είναι πολύ πιθανό να χρειάζεται η κατασκευή άλλων ειδικών αναπαραστάσεων και τελεστών.



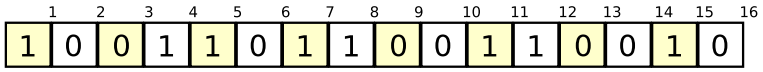
Σχήμα 2.5: *Χρωμόσωμα*

2.4.1 Δυαδική Αναπαράσταση

Η πρώτη αναπαράσταση που έχει χρησιμοποιηθεί στους ΓΑ, είναι η δυαδική αναπαράσταση. Ο γονότυπος έχει τη μορφή της συμβολοσειράς από δυαδικά ψηφία (bit-strings). Κατά την χρήση της δυαδικής αναπαράστασης, καλούμαστε να ορίσουμε το μήκος που θα έχει η συμβολοσειρά, καθώς και τον τρόπο κωδικοποίησης του χώρου του φαινότυπου. Στη περίπτωση που το πρόβλημα διαθέτει φαινότυπο που μπορεί να εκφραστεί άμεσα από μία δυαδική συμβολοσειρά δεν τίθεται θέμα κωδικοποίησης και αποκωδικοποίησης. Είναι όμως συνηθισμένο να χρησιμοποιείται η δυαδική αναπαράσταση σε προβλήματα με μη δυαδική πληροφορία. Για παράδειγμα σε μία δυαδική συμβολοσειρά μήκους 160 bits μπορούν να περιγραφούν πέντε αριθμοί κινητής υποδιαστολής (floating point) των 32 bits.

Η δυαδική αναπαράσταση εμφανίζει προβλήματα όταν εφαρμόζεται σε πολυδιάστατα προβλήματα με μεγάλη αριθμητική ακρίβεια. Για παράδειγμα, έστω 100 μεταβλητές κινητής υποδιαστολής, με απαιτούμενη ακρίβεια έξη δεκαδικών ψηφίων και με πεδίο ορισμού $[-500, 500]$. Στην περίπτωση αυτή το μήκος της δυαδικής συμβολοσειράς ανέρχεται στα 3000 ψηφία και ο χώρος αναζήτησης του γονότυπου αποτελείται από 10^{1000} τιμές. Η απόδοση του ΓΑ είναι πολύ χαμηλή, εξαιτίας του μεγάλου χώρου αναζήτησης. Επίσης είναι πολύ πιθανό (όπως συμβαίνει και με το παράδειγμα) ο χώρος αναζήτησης να περιέχει και

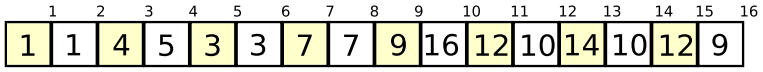
τιμές οι οποίες δεν είναι σωστές, δηλαδή να μην ανήκουν στο πεδίο ορισμού των μεταβλητών (σε επίπεδο φαινότυπου). Αυτό συμβαίνει όταν το μήκος του χώρου του φαινοτύπου δεν είναι δύναμη του δύο. Επιπλέον η χρήση αλγόριθμων αποκωδικοποίησης του γονότυπου επιβαρύνουν σε αρκετό βαθμό την ταχύτητα του ΓΑ. Για τους παραπάνω λόγους είναι προτιμότερο, όταν είναι δυνατό, η χρήση άλλων αναπαραστάσεων με δομή που είναι παρόμοια με τη φύση του προβλήματος (πχ διάνυσμα ακέραιων αριθμών).



Σχήμα 2.6: *Χρωμόσωμα δυαδικής αναπαράστασης*

2.4.2 Αναπαράσταση Ακεραίων

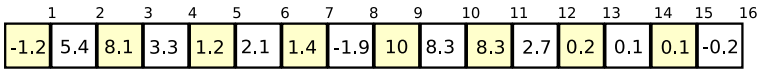
Στην αναπαράσταση αυτή κάθε άτομο περιέχει ένα διάνυσμα από ακέραιους αριθμούς. Είναι χρήσιμη για προβλήματα στα οποία ο φαινότυπος αποτελείται από ακέραιους αριθμούς, έτσι ώστε να μην χρειάζεται η εφαρμογή αλγόριθμων αποκωδικοποίησης όπως συμβαίνει με την περίπτωση της δυαδικής αναπαράστασης. Κάθε στοιχείο του διανύσματος βρίσκεται μέσα σε ένα ορισμένο διάστημα τιμών και κάθε τελεστής που εφαρμόζεται, ενεργεί και παράγει τιμές που ανήκουν μόνο σ΄ αυτό. Με το τρόπο αυτό αποφεύγεται η εμφάνιση λύσεων που δεν ανήκουν στο χώρο αναζήτησης.



Σχήμα 2.7: *Χρωμόσωμα με αναπαράσταση ακεραίων*

2.4.3 Αναπαράσταση Κινητής Υποδιαστολής

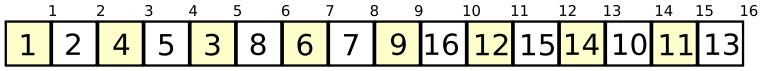
Στην αναπαράσταση κινητής υποδιαστολής κάθε χρωμόσωμα αποτελείται από ένα διάνυσμα αριθμών κινητής υποδιαστολής. Παρόλο πού η ακρίβεια των αριθμών εξαρτάται από την αρχιτεκτονική του συστήματος, η ταχύτητα είναι πολύ μεγαλύτερη και η υλοποίηση καλύτερη. Επιπλέον η αναπαράσταση κινητής υποδιαστολής είναι ικανή να διαχειριστεί αποτελεσματικά, χωρίς να ελαττωθεί η ταχύτητα εκτέλεσης, προβλήματα με μεγάλο χώρο αναζήτησης. Αντίθετα, η δυαδική αναπαράσταση θα πρέπει να μειώσει την ακρίβεια των τιμών της ώστε να διατηρήσει ικανοποιητική απόδοση. Ομοίως με την αναπαράσταση ακεραίων, κάθε στοιχείο του διανύσματος βρίσκεται μέσα σε ένα ορισμένο διάστημα τιμών και οι τελεστές ενεργούν μόνο μέσα σ΄ αυτό.



Σχήμα 2.8: *Χρωμόσωμα με αναπαράσταση κινητής υποδιαστολής*

2.4.4 Αναπαράσταση Μετάθεσης

Στις αναπαραστάσεις που έχουν αναφερθεί το διάνυσμα του χρωμοσώματος περιέχει αριθμούς οι οποίοι μπορούν να εμφανιστούν περισσότερο από μία φορά πάνω σ΄ αυτό. Υπάρχουν όμως προβλήματα (π.χ το πρόβλημα του περιοδεύοντος πωλητή) που έχουν ως περιορισμό, στο κάθε χρωμόσωμα να εμφανίζονται μία μόνο φορά αριθμοί που ανήκουν σε μία συγκεκριμένη συλλογή αριθμών. Στις περιπτώσεις αυτές μας ενδιαφέρει η σειρά με την οποία τοποθετούνται τα γονίδια στο χρωμόσωμα και όχι η τιμή που περιέχουν. Οι γενετικοί τελεστές για την αναπαράσταση μετάθεσης (permutation) είναι κατασκευασμένοι με βάση τον περιορισμό αυτό, ώστε να ενεργούν στη σειρά των γονιδίων και όχι στην αλλαγή των τιμών τους.



Σχήμα 2.9: *Χρωμόσωμα με αναπαράσταση μετάθεσης*

2.5 Συνάρτηση Ποιότητας

Η συνάρτηση ποιότητας παίζει τον ρόλο του περιβάλλοντος. Εφαρμόζεται σε όλα τα άτομα του πληθυσμού και τα αποτελέσματά του διαμορφώνουν τις βάσεις της επιλογής. Κατά την εφαρμογή της, αποδίδει το μέτρο της ποιότητας του ατόμου σύμφωνα με τις τιμές του φαινότυπου.

2.6 Αρχικοποίηση

Με την αρχικοποίηση εισάγονται οι αρχικές τιμές του πληθυσμού, οι οποίες είναι συμβατές με την αναπαράσταση που έχει οριστεί. Συνήθως οι τιμές αυτές είναι τυχαίες και παράγονται από κάποια γεννήτρια τυχαίων αριθμών (Random Number Generator). Είναι όμως δυνατό, οι αρχικές τιμές να μην είναι τυχαίες, αλλά να προέρχονται από υπάρχοντα πειραματικά δεδομένα ή από αποτελέσματα κάποιου άλλου αλγόριθμου.

2.7 Πληθυσμός

Ο ρόλος του πληθυσμού είναι να περιέχει τις πιθανές λύσεις (άτομα) του προς επίλυση προβλήματος. Είναι συνήθως μία απλή δομή δεδομένων (πίνακας, συνδεδεμένη λίστα) με μοναδική παράμετρο το μέγεθος του, δηλαδή το πλήθος των ατόμων που θα περιέχει.

2.8 Ανασυνδυασμός

Ο ανασυνδυασμός εκμεταλλεύεται την πληροφορία από δύο (ή περισσότερα) άτομα που έχουν επιλεχθεί (γονείς) από τον πληθυσμό. Δημιουργεί έναν ή περισσότερους απογόνους συνδυάζοντας τα χαρακτηριστικά των γονέων. Η λειτουργία του στηρίζεται στη πιθανότητα δημιουργίας καλύτερων λύσεων από τον συνδυασμό άλλων σχετικά καλών. Υπάρχουν ποικίλες υλοποιήσεις τελεστών ανασυνδυασμού οι οποίες διαφέρουν με βάση την αναπαράσταση και τη φύση του προβλήματος.

Παρακάτω συμβολίζουμε με $C_1 = (c_1^1, c_2^1, ..., c_n^1)$ και $C_2 = (c_1^2, c_2^2, ..., c_n^2)$ τα δύο χρωμοσώματα που έχουν επιλεχθεί από τον πληθυσμό, στα οποία θα εφαρμοστεί ο τελεστής του ανασυνδυασμού. Όπου c_i^1 και c_i^2 το i-οστό γονίδιο του C_1 και C_2 χρωμοσώματος αντίστοιχα, με $i \epsilon [1, n]$ και n ίσο με το πλήθος των γονιδίων του χρωμοσώματος.

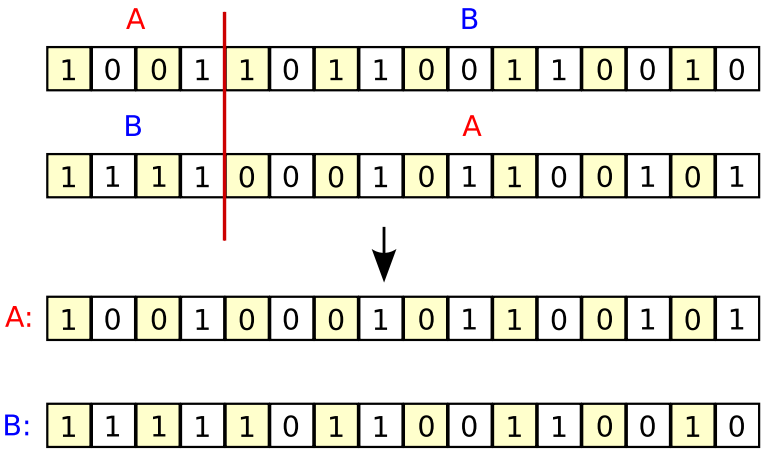
2.8.1 Τελεστές Ανασυνδυασμού για Δυαδική Αναπαράσταση

Single Point

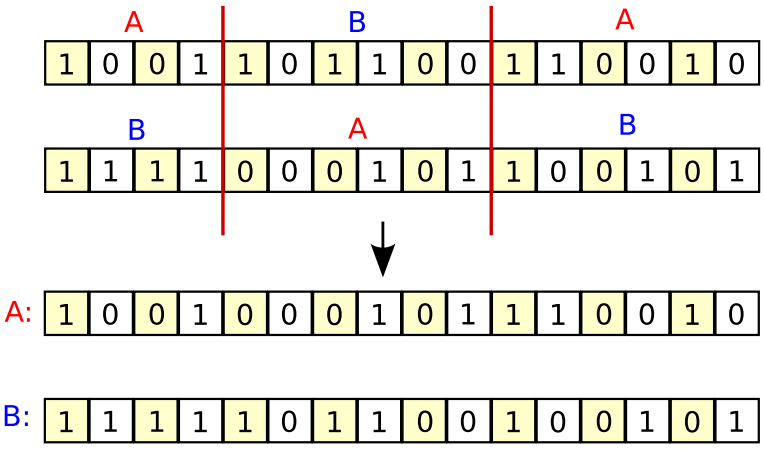
Αποτελεί τον πρώτο τελεστή ανασυνδυασμού που αναπτύχθηκε, όπου από τα χρωμοσώματα των δύο γονέων (C_1 και C_2) δημιουργούνται δύο απόγονοι με χρωμοσώματα $H_1 = (c_1^1, ..., c_i^1, c_{i+1}^2, ..., c_n^2)$ και $H_2 = (c_1^2, ..., c_i^2, c_{i+1}^1, ..., c_n^1)$. Όπου i ένας τυχαίος αριθμός στο διάστημα $[1, n]$ και ορίζει το σημείο κοπής της πληροφορίας.

N Point

Στον τελεστή αυτό, ο ανασυνδιασμός πραγματοποιείται με n σημεία κοπής. Για παράδειγμα θέτουμε $n = 2$, τότε επιλέγονται σημεία κοπής. Έστω i, j δύο τυχαίοι αριθμοί, με $i < j$ και $i, j \epsilon [1, n]$, τότε $H_1 = (c_1^1, ...c_i^1, c_{i+1}^2, ..., c_j^2, c_{j+1}^1, ..., c_n^1)$ και $H_2 = (c_1^2, ...c_i^2, c_{i+1}^1, ..., c_j^1, c_{j+1}^2, ..., c_n^2)$ είναι οι δύο απόγονοι.



Σχήμα 2.10: *Τελεστής single point ανασυνδυασμού*



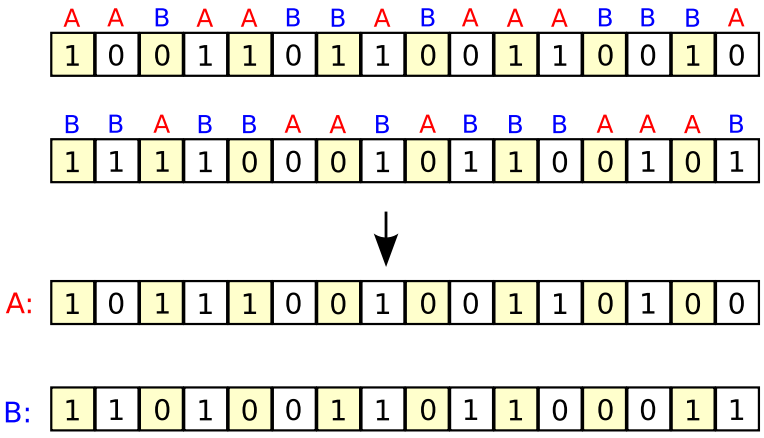
Σχήμα 2.11: *Τελεστής N Point ανασυνδυασμού, με* $n = 2$.

Uniform

Οι τιμές στο κάθε γονίδιο των δύο απογόνων επιλέγονται τυχαία από τις τιμές των αντίστοιχων γονιδίων των γονέων. Δηλαδή, $H_k = (c_1^\lambda, ..., c_n^\lambda)$, με $k = 1, 2$ και $\lambda = 1$ ή 2.

2.8.2 Τελεστές Ανασυνδυασμού για Αναπαράσταση Κινητής Υποδιαστολής

Στην αναπαράσταση κινητής υποδιαστολής ισχύουν οι τελεστές τις δυαδικής αναπαράστασης που έχουν αναφερθεί προηγουμένως. Όμως στην περίπτωση αυτή, οι τελεστές ενεργούν στο φαινότυπο και όχι στο γονότυπο όπως συμβαί-



Σχήμα 2.12: *Τελεστής uniform ανασυνδυασμού*

νει με την δυαδική αναπαράσταση. Το πρόβλημα που εμφανίζουν, είναι ότι κατά την εφαρμογή τους δεν εισάγεται κάποια νέα πληροφορία. Πιο συγκεκριμένα, οι τιμές που δίνονται στο πληθυσμό κατά την αρχικοποίηση διατηρούνται και στις επόμενες γενεές, το μόνο που αλλάζει είναι οι συνδυασμοί τους, δηλαδή η θέση τους. Για την εισαγωγή νέων τιμών στον πληθυσμό, οι προσεγγίσεις αυτές στηρίζονται ολοκληρωτικά στον τελεστή της μετάλλαξης.

Για να ξεπεραστεί το πρόβλημα αυτό αναπτύχθηκαν νέοι τελεστές ανασυνδυασμού, οι οποίοι έχουν την δυνατότητα να εισάγουν καινούριες τιμές ενεργώντας επάνω στο φαινότυπο, με βάση την πληροφορία των γονέων, όπως συμβαίνει με τους τελεστές συνδυασμού στη δυαδική αναπαράσταση. Παρακάτω περιγράφονται οι γνωστότεροι τελεστές ανασυνδυασμού αναπαράστασης κινητής υποδιαστολής.

Flat

Δημιουργείται ένας απόγονος $H = (h_1, h_2, ...h_i, h_{i+1}, ...h_n)$, με h_i μία τυχαία τιμή από το διάστημα $[c_i^{min}, c_i^{max}]$, όπου $c_i^{min} = min(c_i^1, c_i^2)$ και $c_i^{max} = max(c_i^1, c_i^2)$.

Arithmetical

Δημιουργούνται δύο απόγονοι, $H_k = (h_1^k, h_2^k, ...h_i^k, h_{i+1}^k, ...h_n^k)$ με $h_i^1 = \lambda c_i^1 + (1-\lambda)c_i^2$, όπου $k = 1, 2$, $h_i^2 = \lambda c_i^2 + (1-\lambda)c_i^1$ και λ ένας τυχαίος αριθμός στο $[0, 1]$.

Extended Line

Δημιουργείται ένας μόνο απόγονος $H = (h_1, h_2, ...h_i, h_{i+1}, ...h_n)$, με $h_i = c_i^{min} + \alpha(c_i^{max} - c_i^{min})$ όπου α ένας τυχαίος αριθμός στο $[-0.25, 1.25]$, $c_i^{min} = min(c_i^1, c_i^2)$ και $c_i^{max} = max(c_i^1, c_i^2)$.

Average

Δημιουργείται ένας απόγονος $H = (h_1, h_2, ...h_i, h_{i+1}, ...h_n)$ με $h_i = \frac{c_i^1 + c_i^2}{2}$.

BLX-α

Δημιουργείται ένας απόγονος $H = (h_1, h_2, ...h_i, h_{i+1}, ...h_n)$ με h_i ένας τυχαίος αριθμός στο διάστημα $[c_i^{min} - I\alpha, c_i^{max} + I\alpha]$, όπου $c_i^{min} = min(c_i^1, c_i^2)$, $c_i^{max} = max(c_i^1, c_i^2)$, $I = c_i^{max} - c_i^{min}$ και α μία σταθερά. Παρατηρούμε ότι ο BLX-α με $\alpha = 0.0$ είναι ισοδύναμος με τον Flat.

Heuristic

Έστω ότι ο γονέας C_1 έχει καλύτερη τιμή ποιότητας από τον C_2. Τότε δημιουργείται ένας απόγονος $H = (h_1, h_2, ...h_i, h_{i+1}, ...h_n)$ με $h_i = r(c_i^1 - c_i^2) + c_i^1$ όπου r ένας τυχαίος αριθμός στο διάστημα $[0, 1]$.

2.8.3 Τελεστές Ανασυνδυασμού για Αναπαράσταση Ακεραίων

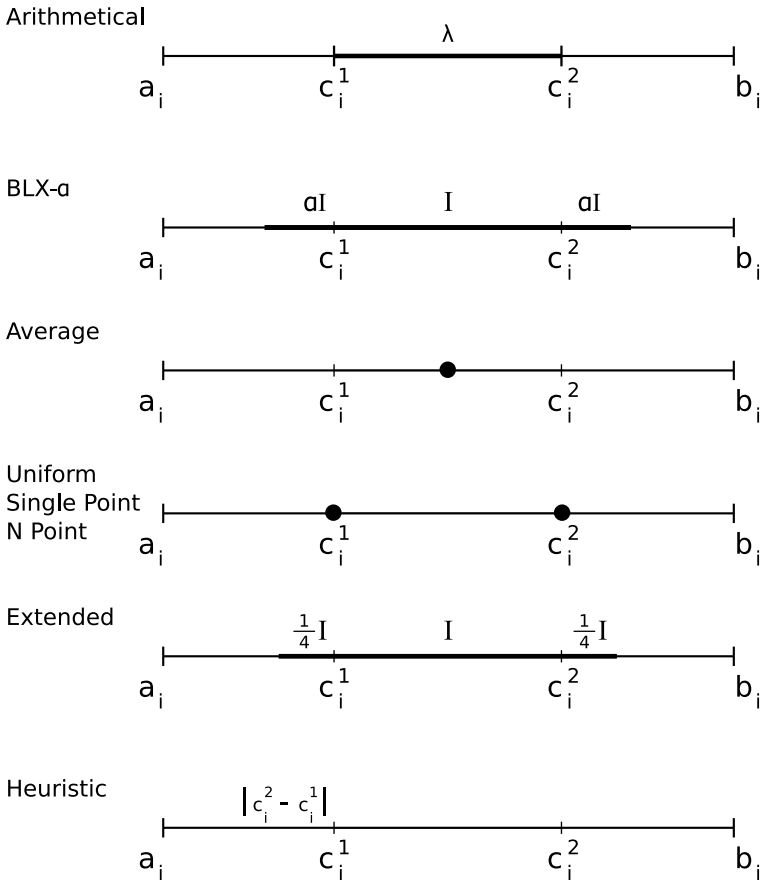
Στις αναπαραστάσεις όπου κάθε γονίδιο έχει ως τιμή ακέραιο αριθμό, είναι φανερό πως μπορούμε να χρησιμοποιήσουμε παρόμοιους τελεστές ανασυνδυασμού με αυτούς της δυαδικής αναπαράστασης. Αλλά δεν υπάρχει κάποιο νόημα χρήσης των τελεστών που κατασκευάστηκαν για την αναπαράσταση κινητής υποδιαστολής, διότι δεν μπορούν να δώσουν αποτέλεσμα με ακέραιες τιμές.

2.8.4 Τελεστές Ανασυνδυασμού για Αναπαράσταση Μετάθεσης

PMX

Ο τελεστής PMX (Partially Mapped Crossover), δημιουργεί δύο απόγονους από τους δύο γονείς που έχουν επιλεχθεί με τα παρακάτω βήματα:

- Επιλέγεται ένα υποσύνολο του χρωμοσώματος, διαλέγοντας τυχαία δύο σημεία. Γίνεται ανταλλαγή των υποσυνόλων, έτσι ώστε οι απόγονοι να



Σχήμα 2.13: *Τελεστές ανασυνδυασμού αναπαράστασης κινητής υποδιαστολής*

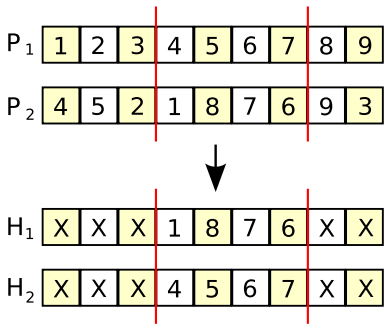
έχουν την μορφή που εμφανίζεται στο σχήμα, όπου με x συμβολίζουμε τα άγνωστα γονίδια.

- Μετά σε κάθε απόγονο εισάγονται τα υπόλοιπα γονίδια, που είναι διαφορετικά από τα γονίδια του υποσυνόλου.
- Τέλος πραγματοποιείται αναταλλαγή των γονίδιων που δεν εισήχθησαν στο προηγούμενο στάδιο.

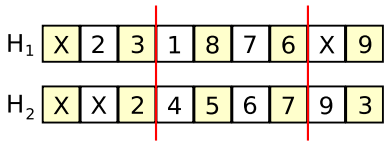
Ο PMX εκμεταλεύεται τις ομοιότητες που έχουν να κάνουν με την τιμή και την θέση των γονίδιων.

Cycle

Ο τελεστής αυτός δημιουργεί δύο απόγονους από δύο γονείς. Κατά την εφαρμογή του διαχωρίζει τα γονίδια σχηματίζοντας κύκλους. Ένας κύκλος



Σχήμα 2.14: *PMX πρώτο στάδιο*



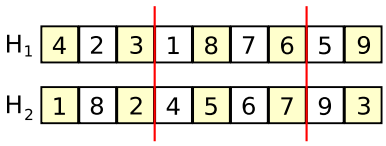
Σχήμα 2.15: *PMX δεύτερο στάδιο*

είναι ένα υποσύνολο από γονίδια, όπου κάθε γονίδιο εμφανίζεται να αντιστοιχίζεται με άλλο γονίδιο του ίδιου κύκλου μεταξύ των δύο γονέων. Οι απόγονοι δημιουργούνται με την εναλλάξ επιλογή των κύκλων. Η διαδικασία σχηματισμού κύκλου λειτουργεί ως εξής:

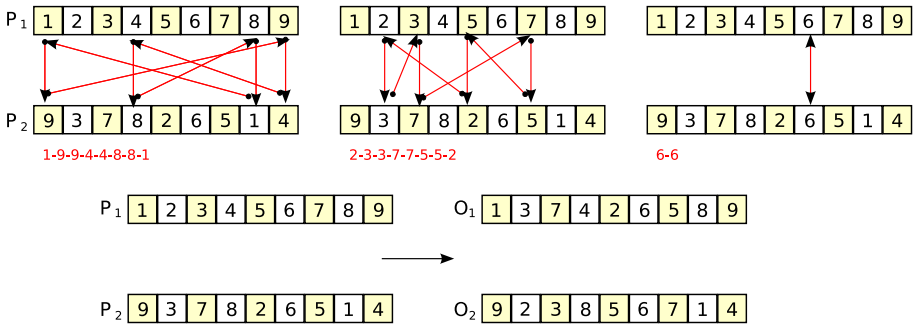
1. Ξεκίνησε με την πρώτη αχρησιμοποίητη θέση γονιδίου του πρώτου γονέα.
2. Αντιστοίχισε με το γονίδιο αντίστοιχης θέσης του δεύτερου γονέα.
3. Αντιστοίχισε με το ίδιο γονίδιο του πρώτου γονέα.
4. Εισήγαγε αυτό το γονίδιο στον κύκλο.
5. Επανέλαβε τα βήματα 2 έως 4 μέχρι να φτάσεις στο γονίδιο του βήματος 1 του πρώτου γονέα.

2.9 Μετάλλαξη

Ο τελεστής της μετάλλαξης τροποποιεί αυθαίρετα ένα ή περισσότερα γονίδια του επιλεγμένου χρωμοσώματος, με σκοπό να αυξήσει την ανομοιομορφία του πληθυσμού. Με τον τρόπο αυτό αποτρέπει την πρόωρη σύγκληση του ΓΑ, διασφαλίζοντας παράλληλα την πιθανότητα εξέτασης οποιουδήποτε σημείου στο χώρο αναζήτησης.



Σχήμα 2.16: *PMX τελευταίο στάδιο*

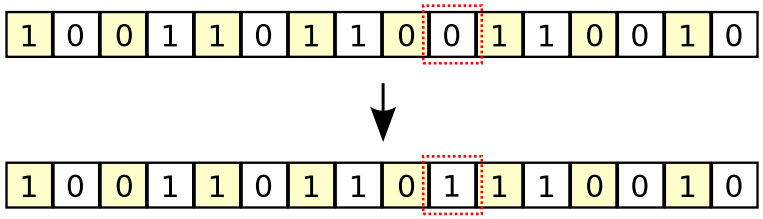


Σχήμα 2.17: *Τελεστής Cycle Recombination*

Παρακάτω, συμβολίζουμε με $C = (c_1, ..., c_i, ..., c_n)$ το χρωμόσωμα που επιλέχθηκε για να εφαρμοστεί επάνω στο c_i γονίδιό του ο τελεστής της μετάλλαξης. Το αποτέλεσμα της μετάλλαξης θα είναι ένα γονίδιο με νέα τιμή $c_i^{'}$, δηλαδή ένα νέο χρωμόσωμα $C^{'} = (c_1, ..., c_i^{'}, ..., c_n)$.

2.9.1 Τελεστές Μετάλλαξης για Δυαδική Αναπαράσταση

Στους ΓΑ με δυαδική αναπαράσταση ο τελεστής της μετάλλαξης που εφαρμόζεται, είναι ο bit flip. Στον τελεστή αυτό, στο χρωμόσωμα επιλέγεται τυχαία ένα γονίδιο και η τιμή του αντιστρέφεται, δηλαδή το 1 γίνεται 0 και το αντίστροφο.



Σχήμα 2.18: *Τελεστής μετάλλαξης δυαδικής αναπαράστασης*

2.9.2 Τελεστές Μετάλλαξης για Αναπαράσταση Κινητής Υποδιαστολής

Uniform Random

Στην περίπτωση αυτή, το $c_i^{'}$ είναι ένας τυχαίος αριθμός από το διάστημα $[a_i, b_i]$, όπου a_i, b_i είναι αντίστοιχα το έλαχιστο και το μέγιστο όριο της τιμής του c_i.

Non-Uniform Random

Έστω t η τρέχουσα γενιά και g_{max} ο μέγιστος αριθμός επαναλήψεων. Τότε ο τελεστής αυτός έχει αποτέλεσμα:

$$c_i^{'} = \begin{cases} c_i + \triangle(t, b_i - c_i) & \text{αν r=0} \\ c_i - \triangle(t, c_i - a_i) & \text{αν r=1} \end{cases}$$

όπου $\triangle(t, y) = y(1 - r^{(1-\frac{t}{g_{max}})^b})$, με r ένας τυχαίος αριθμός ίσος με 0 ή 1 και b μία σταθερά που ορίζεται από τον χρήστη και δηλώνει το βαθμό εξάρτησης στο πλήθος των επαναλήψεων. Η συνάρτηση $\triangle(t, y)$ δίνει αποτέλεσμα στο διάστημα $[0, y]$.

Creep

Ο τελεστής αυτός λειτουργεί προσθέτοντας μία μικρή θετική ή αρνητική τιμή σε ένα τυχαίο γονίδιο, δηλαδή $c_i^{'} = c_i + (b_i, a_i)r$, όπου r ένας τυχαίος αριθμός από το διάστημα $[-1, 1]$ και a_i, b_i το ελάχιστο και μέγιστο επιτρεπτό όριο του c_i γονίδιου.

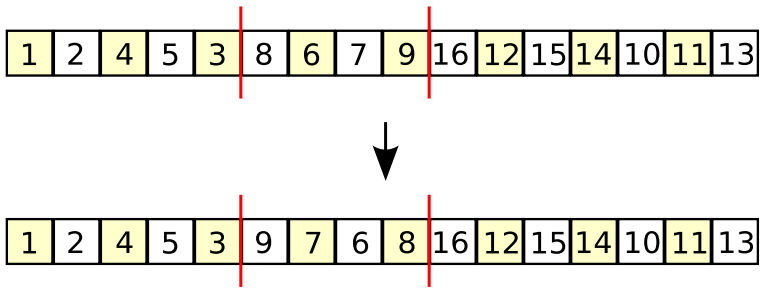
2.9.3 Τελεστές Μετάλλαξης για Αναπαράσταση Ακεραίων

Στην περίπτωση της αναπαράστασης ακέραιων αριθμών, οι τελεστές μετάλλαξης είναι οι Uniform Random και non-Uniform Random που αναφέρθηκαν στις προηγούμενες παραγράφους.

2.9.4 Τελεστές Μετάλλαξης για Αναπαράσταση Μετάθεσης

Inversion

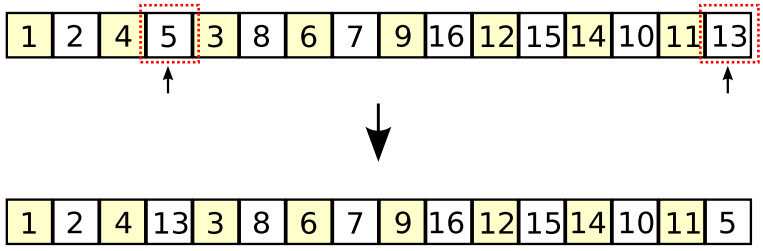
Ο τελεστής αυτός επιλέγει με τυχαίο τρόπο δύο γονίδια του χρωμοσώματος και ανταλλάσσει τις θέσεις τους.



Σχήμα 2.19: *Inversion Mutation*

Swap

Εδώ επιλέγονται δύο σημεία με τυχαίο τρόπο και αντιστρέφεται η σειρά με την οποία εμφανίζονται τα γονίδια του χρωμοσώματος μέσα σ΄ αυτά.



Σχήμα 2.20: *Swap Mutation*

2.10 Επιλογή

Ο τελεστής της επιλογής διαφέρει από τους προηγούμενους καθώς δεν δημιουργεί καινούριες λύσεις. Σκοπός του είναι η επιλογή των σχετικά καλών λύσεων, με βάση την τιμή της ποιότητας των ατόμων του πληθυσμού της ίδιας γενιάς.

2.10.1 Επιλογή με αναλογία της ποιότητας του ατόμου

Μία από τις στρατηγικές που ακολουθούνται για την επιλογή των γονέων, είναι η μέθοδος των αναλογιών με βάση την ποιότητα των ατόμων. Στηρίζεται στην ιδέα ότι τα άτομα με την καλύτερη τιμή ποιότητας θα πρέπει να έχουν μεγαλύτερη πιθανότητα επιλογής. Εδώ η επιλογή χωρίζεται σε δύο στάδια,

το πρώτο έχει να κάνει με την κατασκευή των ποσοστών επιλογής για όλα τα άτομα που συμμετέχουν και το δεύτερο με τον αλγόριθμο επιλογής.

Όσον αφορά το πρώτο στάδιο, δημιουργείται ένας μεροληπτικός τροχός με θέσεις που αντιστοιχίζονται στα άτομα του πληθυσμού, οι οποίες έχουν μέγεθος ανάλογο της ποιότητας του ατόμου. Έστω ότι θέλουμε να επιλέξουμε λ άτομα από ένα πληθυσμό $P = (C_1, ..., C_N)$ ατόμων, όπου τα άτομά του είναι ταξινομημένα από την καλύτερη προς την χειρότερη τιμή ποιότητας , οι πιθανότητες επιλογής δημιουργούνται ως εξής:

- Υπολογίζεται η συνολική τιμή ποιότητας του πληθυσμού.

  $F = \sum_{i=1}^{\lambda} f(C_i)$, με $i \in [1, \lambda]$, όπου $f(C_i)$ η ποιότητα του i-οστού ατόμου.

- Υπολογίζεται η πιθανότητα επιλογής P_i για κάθε άτομο C_i.

  $$P_i = \frac{f(C_i)}{F}$$

- Τέλος υπολογίζεται η αθροιστική πιθανότητα q_i για κάθε άτομο C_i.

  $$q_i = \sum_{j=1}^{\lambda} P_j$$

Στην μέθοδο αυτή παρατηρείται ότι, τα άτομα με τις καλύτερες τιμές ποιότητας καταλαμβάνουν πολύ σύντομα τον πληθυσμό εξαιτίας της μεγάλης πιθανότητας επιλογής. Για τον λόγο αυτό, είναι πολύ πιθανό ο ΓΑ να συγκλίνει πρόωρα σε κάποιο τοπικό βέλτιστο του προβλήματος.

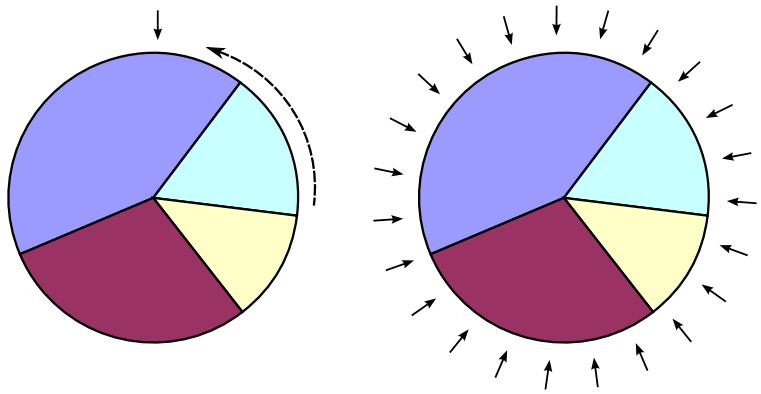
Για το δεύτερο στάδιο υπάρχουν δύο αλγόριθμοι επιλογής γονέων που βασίζονται στις πιθανότητες που υπολογίστηκαν προηγουμένως.

Cost Roulette Wheel

Η διαδικασία επιλογής πραγματοποιείται με το γύρισμα του τροχού λ φορές. Κάθε φορά επιλέγεται ένα άτομο ως εξής:

- Έστω r ένας τυχαίος αριθμός από το διάστημα $[0, 1]$.

- Αν $r \leq q_1$, τότε επιλέγεται το πρώτο άτομο. Διαφορετικά επιλέγεται το i-οστό άτομο όπου ισχύει $q_{i-1} < r \leq q_i, i \in [2, \lambda]$.



Σχήμα 2.21: *Αριστερά Cost Roulette Wheel, δεξιά Stochastic Universal Sampling*

Stochastic Universal Sampling

Ο αλγόριθμος αυτός προσομοιώνει την Cost Roulette Wheel με τη διαφορά ότι υπάρχουν λ ίσα κατανεμημένοι δείκτες γύρο από τον τροχό, έτσι ώστε με μία μόνο περιστροφή να πραγματοποιείται η επιλογή. Από αυτή τη διαδικασία το πλήθος των επιλογών κάθε χρωμοσώματος C_i, ορίζεται μεταξύ $\lfloor p_i N \rfloor$ και $\lceil p_i N \rceil$. Υλοποιείται ως εξής:

- Έστω r ένας τυχαίος αριθμός από το διάστημα τιμών $[0, \frac{1}{N}]$
- Για όσο ισχύει $r \leq q_i$, τότε επιλέγεται το i-οστό άτομο και ο αριθμός r αυξάνεται κατά $1/N$.
- Η διαδικασία επαναλαμβάνεται έως ότου επιλεχθούν λ άτομα συνολικά.

Κλιμάκωση Ποιότητας

Πέραν του προβλήματος της πρόωρης σύγκλισης του ΓΑ που αναφέρθηκε προηγουμένως εμφανίζονται και άλλα δύο μειονεκτήματα. Σε περιπτώσεις όπου οι τιμές ποιότητας των ατόμων διαφέρουν ελάχιστα μεταξύ τους, οι πιθανότητες επιλογής που υπολογίζονται είναι σχεδόν ίσες. Συνεπώς η επιλογή γίνεται περισσότερο τυχαία. Επιπλέον, ο μηχανισμός επιλογής συμπεριφέρεται διαφορετικά στην ίδια συνάρτηση ποιότητας όταν αλλάζει η κλίμακα των τιμών. Για να ξεπεραστούν τα προβλήματα αυτά, εφαρμόζονται διαδικασίες που μεταβάλλουν την τιμή της ποιότητας όλων των ατόμων του πληθυσμού, είναι οι εξής:

1. Windowing. Από την τιμή ποιότητας κάθε ατόμου αφαιρείται η τιμή ποιότητας του χειρότερου.

$$f^{'}(C_i) = f(C_i) - f_{min}$$

2. Linear Scaling. Η τιμή ποιότητας μεταβάλλεται γραμμικά, όπου οι τιμές των παραμέτρων α, β επιλέγονται από τον χρήστη.

$$f^{'}(C_i) = \alpha f(C_i) + \beta$$

3. Sigma Scaling. Η τιμή ποιότητας μεταβάλλεται σύμφωνα με την μέση τιμή και την τυπική απόκλιση του πληθυσμού.

$$f^{'}(C_i) = max(f(C_i) + (\overline{f} - c * \sigma), 0.0)$$

Όπου c ένας ακέραιος αριθμός μικρού μεγέθους (συνήθως από το διάστημα 1 έως 5), $\overline{f}$ η μέση τιμή και σ η τυπική απόκλιση.

4. Power Law. Με την μέθοδο αυτή, η τιμή ποιότητας υψώνεται σε δύναμη κοντά στο ένα (πχ. 1.005).

$$f^{'}(C_i) = (f(C_i))^k$$

2.10.2 Rank Roulette Wheel

Ο τελεστής αυτός είναι παρόμοιος με τον τελεστή επιλογής Cost Roulette Wheel, με μόνη διαφορά τον τρόπο υπολογισμού των πιθανοτήτων επιλογής, δηλαδή διαφέρει στο πρώτο στάδιο. Εδώ η πιθανότητα υπολογίζεται με βάση την θέση (rank) που κατέχει το χρωμόσωμα στον ταξινομημένο (ως προς την τιμή ποιότητας) πληθυσμό.

$$P_i = \frac{\lambda - i + 1}{\sum_{i=1}^{\lambda} i}$$

2.10.3 Tournament Selection

Οι τελεστές επιλογής που έχουν παρουσιαστεί μέχρι στιγμής χρειάζονται πληροφορίες από ολόκληρο τον πληθυσμό, ο οποίος θα πρέπει να είναι ταξινομημένος ως προς την τιμή ποιότητας. Επιπλέον, όταν ο πληθυσμός είναι μεγάλου μεγέθους ή είναι κατανεμημένος (σε κάποιο παράλληλο σύστημα), η απόκτηση αυτών των πληροφοριών σε συνδυασμό με την ταξινόμηση του πληθυσμού καταναλώνει αρκετή επεξεργαστική ισχύ. Παράλληλα εμφανίζονται μειονεκτήματα που έχουν αναφερθεί στις προηγούμενες παραγράφους, όπως η πρόωρη σύγκλιση του ΓΑ.

Σε αντίθεση λοιπόν με τις παραπάνω δυσκολίες ο τελεστής Tournament Selection δεν χρειάζεται πληροφορίες από όλο τον πληθυσμό, αλλά ούτε προϋποθέτει να είναι ταξινομημένος ο πληθυσμός. Επιπλέον είναι απλός στην υλοποίηση και γρήγορος στη λειτουργία του. Λειτουργεί ως εξής:

- Επιλέγονται τυχαία κ άτομα από τον πληθυσμό, με $\kappa \geq 2$.
- Από τα κ άτομα, επιλέγεται το άτομο με την καλύτερη τιμή ποιότητας.
- Η διαδικασία επαναλαμβάνεται έως ότου επιλεχθούν και τα λ άτομα που απαιτούνται.

2.11 Φυσική Επιλογή

Ο τελεστής φυσικής επιλογής (Natural Selection) ή τελεστής αντικατάστασης ασχολείται με την κατασκευή της επόμενης γενιάς από τα δεδομένα της τρέχουσας και των απογόνων της. Είναι το τελευταίο βήμα που πραγματοποιεί ο ΓΑ σε κάθε επανάληψη. Γενικότερα, έχουν αναπτυχθεί αρκετές μέθοδοι για τη δημιουργία την επόμενης γενιάς, οι οποίες χρησιμοποιούν ως βασικό μέτρο επιλογής την ηλικία ή την τιμή ποιότητας των ατόμων.

Όταν έρχεται η στιγμή της φυσικής επιλογής, στον ΓΑ έχει σχηματιστεί ένα σύνολο από άτομα που προήλθε από τα προηγούμενα βήματα. Το σύνολο αυτό, αποτελείται από μ άτομα της τρέχουσας γενιάς t και από τους λ απογόνους τους. Επίσης ανεξάρτητα της μεθόδου επιλογής, μπορεί να εφαρμοστεί ο ελιτισμός (elitism), σύμφωνα με τον οποίο το άτομο με την καλύτερη τιμή ποιότητας κρατείτε πάντοτε στον πληθυσμό. Συνεπώς διατηρείται η καλύτερη λύση, έως ότου βρεθεί ακόμα καλύτερη.

2.11.1 Επιλογή ως προς την ηλικία

Στην συγκεκριμένη περίπτωση πάντοτε κρατούνται οι απόγονοι στον πληθυσμό της νέας γενιάς και αντικαθιστούν το αντίστοιχο πλήθος της τρέχουσας γενιάς. Αυτό επιτυγχάνεται με δύο τρόπους:

1. Αντικατάσταση του χειρότερου. Τα λ χειρότερα άτομα του πληθυσμού της τρέχουσας γενιάς, επιλέγονται για να καταλάβουν τη θέση τους οι λ απόγονοι. Στη περίπτωση όπου οι απόγονοι είναι όσο όλος ο πληθυσμός, δηλαδή ισχύει $\lambda = \mu$, τότε αντικαθιστώνται όλα τα άτομα από τους απογόνους.

2. Τυχαία αντικατάσταση. Η αντικατάσταση των ατόμων του πληθυσμού από τους απογόνους τους πραγματοποιείται με τυχαίο τρόπο. Δηλαδή επιλέγονται τυχαία τα λ άτομα που θα εγκαταλείψουν τον πληθυσμό και την θέση τους καταλαμβάνουν οι απόγονοι.

2.11.2 Επιλογή ως προς την τιμή ποιότητας

Αντίθετα με την επιλογή ως προς την ηλικία, η επόμενη γενιά επιλέγεται από τα άτομα των απογόνων και του τρέχοντος πληθυσμού. Κριτήριο επιλογής είναι η τιμή ποιότητας. Επάνω στο μοντέλο αυτό μπορούν να εφαρμοστούν τεχνικές του τελεστή επιλογής, που έχουν παρουσιαστεί στη παράγραφο 2.10.

1. Tournament. Από το σύνολο των ατόμων (απόγονοι και τρέχουσα γενιά) επιλέγεται ένα μικρό υποσύνολο k ατόμων και το άτομο με την χειρότερη τιμή ποιότητας απομακρύνεται. Η διαδικασία αυτή επαναλαμβάνεται έως ότου παραμείνουν μ άτομα.

2. GENITOR. Από το σύνολο όλων των ατόμων $(\lambda+\mu)$ απομακρύνονται τα λ χειρότερα άτομα. Παρόλα αυτά η μέθοδος αυτή μπορεί να οδηγήσει τον ΓΑ σε πρόωρη σύγκλιση, καθώς τον κατευθύνει μόνο προς τα υπάρχοντα ποιοτικότερα άτομα.

2.12 Συνθήκη Τερματισμού

Ο ΓΑ εκτελείται επαναληπτικά έως ότου ικανοποιηθεί η συνθήκη ή ικανοποιηθούν οι συνθήκες τερματισμού που έχουν οριστεί. Συνήθως η συνθήκη τερματισμού ορίζεται από το πρόβλημα, ωστόσο οι περισσότερο συνηθισμένες είναι η εξής:

1. Μέγιστο πλήθος επαναλήψεων.

2. Μέγιστος χρόνος εκτέλεσης του ΓΑ.

3. Η μη βελτίωση της ποιότητας του καλύτερου ατόμου για ένα προκαθορισμένο πλήθος γενεών.

4. Η εύρεση της βέλτιστης ή μίας αποδεκτής λύσης.

3

Γραφική Αναπαράσταση Γενετικών Αλγόριθμων

Στο προηγούμενο κεφάλαιο ασχοληθήκαμε με τη θεωρεία των ΓΑ. Ωστόσο, το θεωρητικό μοντέλο από μόνο του μπορεί να αποδειχθεί δύσκολο ή να μην επαρκεί για τη μελέτη και τη κατανόησή τους. Όπως έχει προαναφερθεί, ένας ΓΑ διατηρεί ένα πλήθος προφανών λύσεων (άτομα) καθ' όλη τη διάρκεια των επαναλήψεων (γενιές). Αυτό το χαρακτηριστικό μας δίνει δυνατότητες καταγραφής μεγάλου όγκου δεδομένων, που έχουν να κάνουν με όλη τη πορεία του αλγόριθμου καθ' όλη την εφαρμογή του σε ένα πρόβλημα. Επίσης είναι γνωστό ότι ένα διάγραμμα μπορεί να συνδυάζει πολλές πληροφορίες μαζί και να τις εμφανίζει με κατανοητό τρόπο. Για τους παραπάνω λόγους, παρουσιάζεται έντονο ενδιαφέρον στην εύρεση μεθόδων γραφικής αναπαράστασης που έχουν στόχο την εξαγωγή και την μελέτη χρήσιμων πληροφοριών από την εφαρμογή των ΓΑ.

Έχοντας ως γνώμονα το σύνολο των δεδομένων που παράγονται από έναν ΓΑ, παρακάτω θα ασχοληθούμε με μεθόδους που αφορούν την εξαγωγή και την αναπαράσταση χρήσιμων πληροφοριών, που είναι ανεξάρτητες από το πρόβλημα. Από τις μεθόδους αυτές, έχουμε τη δυνατότητα να σχηματίσουμε εικόνες όχι μόνο για την συνολική πορεία του ΓΑ, αλλά και για την κατάσταση των ατόμων του πληθυσμού οποιασδήποτε γενιάς. Όλες οι μέθοδοι που θα παρουσιαστούν, έχουν επιλεχθεί με βάση τη χρησιμότητα αλλά και την δυνατότητα να εφαρμόζονται σε όλες τις αναπαραστάσεις που έχουν αναφερθεί στο προηγούμενο κεφάλαιο.

Τα στοιχεία που συγκρατούν τα δεδομένα της εξέλιξης, είναι το άτομο και ο πληθυσμός. Επιπλέον μπορούμε να χωρίσουμε την πληροφορία σε δύο ομάδες με βάση το χρονικό εύρος στο οποίο αναφέρονται (μία γενιά ή ένα πλήθος

γενεών). Ως πρώτη ομάδα, διακρίνουμε την «Πορεία του ΓΑ», η οποία συγκεντρώνει τις πληροφορίες που εκφράζουν την συνολική εικόνα του ΓΑ. Τα δεδομένα που απαιτούνται συγκεντρώνονται από όλες ή από ένα υποσύνολο τις γενεών. Όσον αφορά τη δεύτερη ομάδα, την αναφέρουμε ως «Κατάσταση του ΓΑ»και τα δεδομένα που απαιτούνται προέρχονται από μία μόνο γενιά. Έτσι λοιπόν, με τις δύο αυτές ομάδες μπορούμε να μελετήσουμε όλη την διαδικασία της εξέλιξης που συνέβη από ένα ΓΑ ώστε να επιλυθεί ένα συγκεκριμένο πρόβλημα.

3.1 Γραφική απεικόνιση της συνολικής πορείας του ΓΑ

Στη κατηγορία αυτή μελετάμε τη συνολική εικόνα του αλγόριθμου. Τα δεδομένα τα οποία χρειάζονται είναι τα εξής:

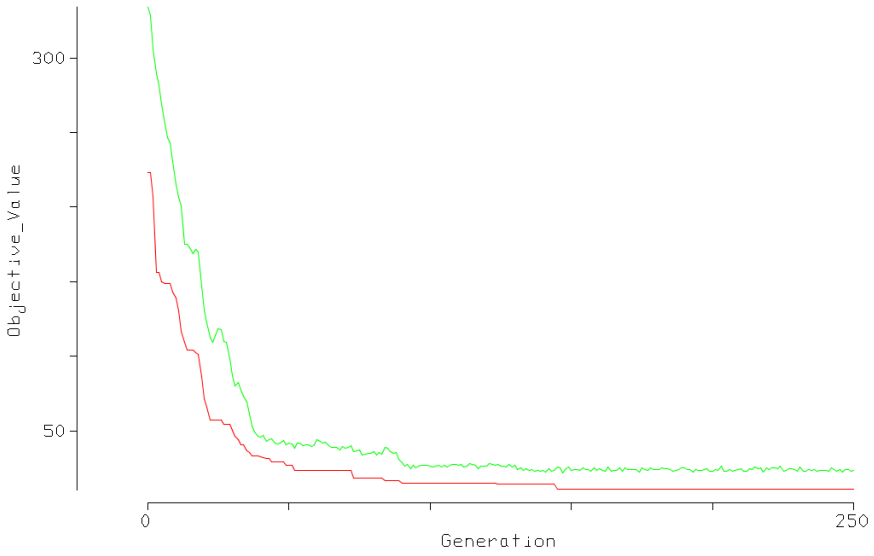
1. Η τιμή της ποιότητας του καλύτερου ατόμου για κάθε γενιά.
2. Όλες τις τιμές των γονιδίων του καλύτερου ατόμου.
3. Όλες τις τιμές ποιότητας από όλα τα υπόλοιπα άτομα του πληθυσμού για όλες τις γενιές.

3.1.1 Διάγραμμα καλύτερης και μέσης τιμής του ΓΑ

Η περισσότερο χρησιμοποιούμενη μέθοδος γραφικής μελέτης του ΓΑ, είναι το δισδιάστατο διάγραμμα μέσης και καλύτερης τιμής ποιότητας του πληθυσμού για όλες τις γενιές. Όπως φαίνεται στην εικόνα 3.1, από το διάγραμμα αυτό δίνεται η συνολική εικόνα της σύγκλισης του ΓΑ. Επίσης, είναι συνηθισμένο να απεικονίζεται ταυτόχρονα και η πληροφορία της χειρότερης τιμής του πληθυσμού, ωστόσο η μεγάλη διαφορά τιμών μεταξύ της χειρότερης και της καλύτερης τιμής ποιότητας, έχει ως αποτέλεσμα την καταστροφή της ανάλυσης του διαγράμματος.

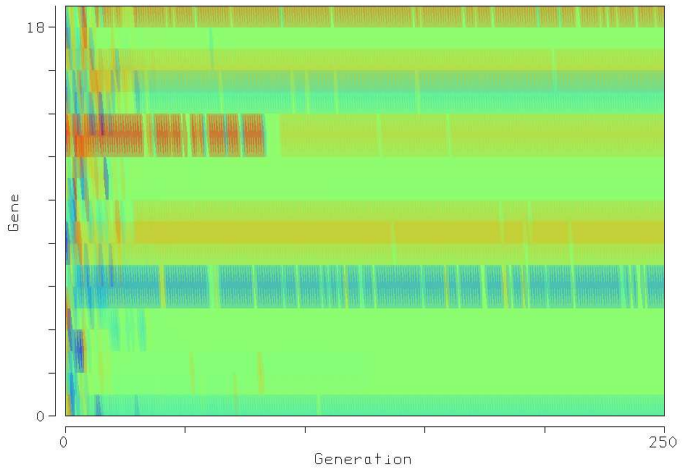
3.1.2 Απεικόνιση τιμών του καλύτερου ατόμου

Έχοντας τις τιμές των γονιδίων του καλύτερου ατόμου του πληθυσμού από όλες τις γενιές, μας δίνεται η δυνατότητα να σχηματίσουμε μία αντιπροσωπευτική εικόνα για την πρόοδο της αναζήτησης σε επίπεδο γονιδίου. Δηλαδή, ασχολούμαστε με τις μεταβλητές του προβλήματος και όχι με τη τιμή ποιότητας του ατόμου. Με τη μέθοδο αυτή, παρατηρούμε τις αλλαγές των τιμών και



Σχήμα 3.1: *Δισδιάστατο διάγραμμα μέσης και καλύτερης τιμής ποιότητας του πληθυσμού για όλες τις γενιές του ΓΑ. Με κόκκινο είναι η καλύτερη τιμή και με πράσινο η χειρότερη. Στον οριζόντιο άξονα είναι οι γενιές και στον κάθετο η τιμή ποιότητας*

τους συνδυασμούς των γονιδίων, από τα οποία αποκτούνται καλές λύσεις. Ο πιο απλός τρόπος για την απεικόνιση αυτών των δεδομένων είναι ένα διάγραμμα με τις γραφικές παραστάσεις των τιμών των γονιδίων σε κάθε γενιά. Ωστόσο, είναι εμφανές ότι το συγκεκριμένο διάγραμμα δεν είναι εύχρηστο καθώς είναι δύσκολο να ξεχωρίσουν όλες οι γραφικές παραστάσεις, με αποτέλεσμα να μην γίνεται κατανοητό. Για τον λόγο αυτό είναι προτιμότερο να αναπαραστήσουμε τις πληροφορίες αυτές με περισσότερο κατανοητό τρόπο, σε δύο ή τρεις διαστάσεις. Σε δύο διαστάσεις, όπως φαίνεται στο σχήμα 3.2, χρησιμοποιούμε το διάγραμμα εικόνας (image plot), ορίζουμε στον άξονα $x'x$ τις επαναλήψεις (γενιές) και στον $y'y$ την θέση στην οποία ανήκει το γονίδιο. Η τιμή που περιέχει ένα γονίδιο σε μία συγκεκριμένη γενιά, αντιστοιχίζεται και απεικονίζεται με ένα συγκεκριμένο χρώμα. Το διάγραμμα αυτό εύκολα μεταφέρεται σε τρεις διαστάσεις (σχήμα 3.3), αρκεί η τιμή του γονιδίου να αντιστοιχηθεί με τον άξονα $z'z$ που απεικονίζει τις τιμές. Και στις δύο μεθόδους, οι τιμές των γονιδίων που αντιστοιχίζονται στα χρώματα και στον άξονα $z'z$, κυμαίνονται μεταξύ της ελάχιστης και μέγιστης τιμής που έχει εμφανιστεί συνολικά σε όλα τα γονίδια. Τέλος, με τα διαγράμματα αυτά παρατηρούμε την κατανομή και τις εναλλαγές των τιμών που συμβαίνουν στα γονίδια του καλύτερου χρωμοσώματος.



Σχήμα 3.2: *Δισδιάστατο διάγραμμα εικόνας, στον οριζόντιο άξονα απεικονίζονται οι γενιές και στον κάθετο τα γονίδια. Με τα χρώματα απεικονίζονται οι τιμές των γονιδίων*

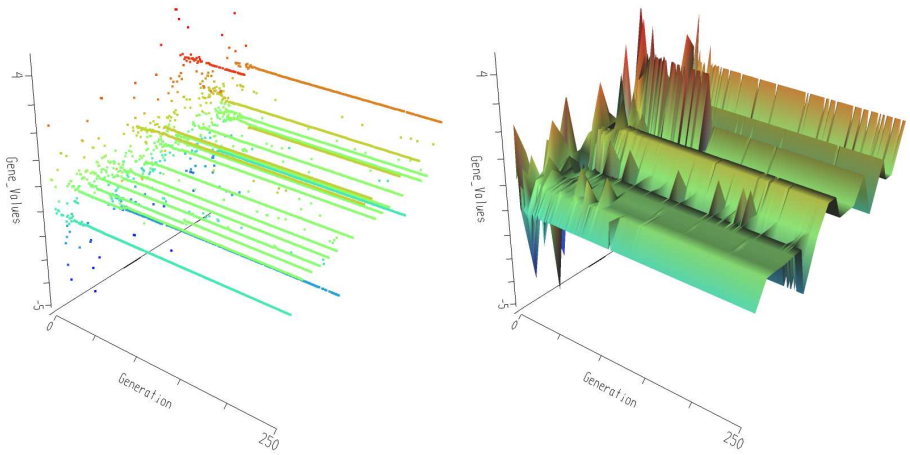
3.2 Γραφική απεικόνιση της κατάστασης του ΓΑ

Φεύγοντας από την συνολική εκτίμηση του ΓΑ, επικεντρωνόμαστε στην ανάλυση πληροφοριών που περιορίζονται σε έναν μόνο πληθυσμό. Τα δεδομένα που απαιτούνται για την παρατήρηση ενός ολόκληρου πληθυσμού μίας ορισμένης γενιάς, είναι τα εξής:

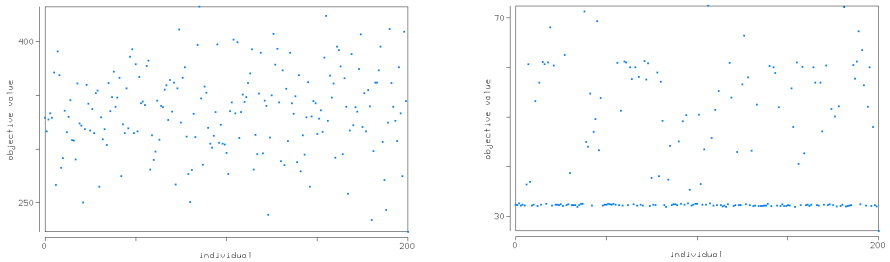
1. Οι τιμές ποιότητας όλων των ατόμων του πληθυσμού.
2. Οι τιμές των γονιδίων για κάθε άτομο του πληθυσμού.

3.2.1 Τιμές ποιότητας όλων των ατόμων

Για να μελετήσουμε την ανομοιομορφία του πληθυσμού, χρειαζόμαστε τις τιμές ποιότητας όλων των ατόμων. Ο ιδανικότερος τρόπος απεικόνισης, είναι ένα διάγραμμα που αντιστοιχίζει τις τιμές ποιότητας με όλα τα άτομα του πληθυσμού, όπως υποδεικνύεται στην εικόνα 3.4. Παράλληλα με την διαδοχική προσπέλαση των γενεών, παρατηρούμε τη διατήρηση ή τη μείωση της ανομοιομορφίας, καθώς συγκρίνουμε τους πληθυσμούς.



Σχήμα 3.3: *Αριστερά, τρισδιάστατο διάγραμμα σημείων. Δεξιά τρισδιάστατο διάγραμμα επιφάνειας. Αποτελούν επέκταση του διαγράμματος 3.2.*



Σχήμα 3.4: *Δισδιάστατο διάγραμμα με τις τιμές ποιότητας όλων των ατόμων ενός πληθυσμού. Αριστερά είναι η αρχική γενιά και δεξιά η 50η γενιά.*

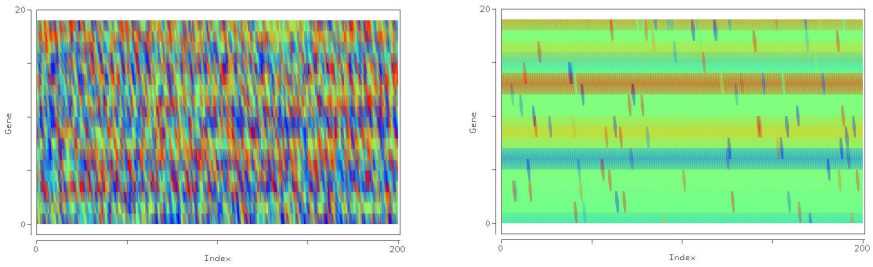
3.2.2 Τιμές των ατόμων

Με τη γραφική απεικόνιση των τιμών των γονιδίων από τα άτομα του πληθυσμού μίας συγκεκριμένης γενιάς, μπορούμε να μελετήσουμε πληροφορίες που αφορούν την κατανομή των ατόμων, τις διαφορές που παρουσιάζουν μεταξύ τους και την σύγκλιση των τιμών τους σε ορισμένες περιοχές του χώρου, από τις οποίες δίνονται καλά αποτελέσματα. Διακρίνουμε δύο κατηγορίες, στη πρώτη μελετάμε όλα τα άτομα του πληθυσμού, ενώ στη δεύτερη συγκρίνουμε το καλύτερο άτομο με τον υπόλοιπο πληθυσμό. Στο σημείο αυτό πρέπει να τονίσουμε ότι ο όγκος της πληροφορία που χρειάζεται είναι πολύ μεγάλος καθώς απαιτούνται δεδομένα από όλες τις τιμές των γονιδίων όλων των ατόμων του πληθυσμού από όλες τις γενιές που έχουν δημιουργηθεί από τον ΓΑ.

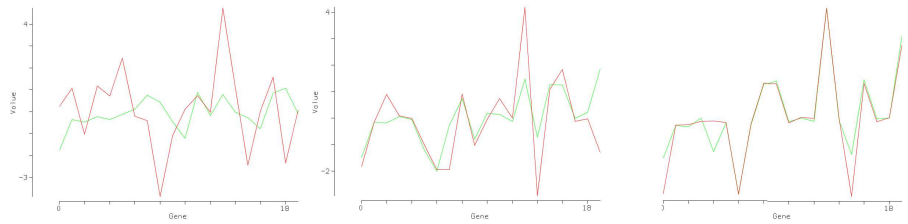
Όσον αφορά την πρώτη κατηγορία έχουμε δύο διαγράμματα, ένα δισδιάστατο και ένα τρισδιάστατο, τα οποία είναι παρόμοια με αυτά της παραγράφου 3.1.2.

Στο δισδιάστατο διάγραμμα εικόνας (εικόνα 3.5), ο άξονας $x'x$ αναπαριστά τη θέση του ατόμου στον πληθυσμό και στον άξονα $y'y$ τα γονίδια. Οι τιμές των γονιδίων του κάθε ατόμου αντιστοιχίζονται και απεικονίζονται με χρώματα. Τα χρώματα διαβαθμίζονται από το μπλε που αντιστοιχεί στην χαμηλότερη τιμή προς το κόκκινο που αντιστοιχεί στην υψηλότερη τιμή. Το τρισδιάστατο διάγραμμα βασίζεται στο δισδιάστατο, με μοναδική προσθήκη τον άξονα $z'z$ στον οποίο αντιστοιχίζονται αριθμητικά οι τιμές των γονιδίων (εικόνα 3.3).

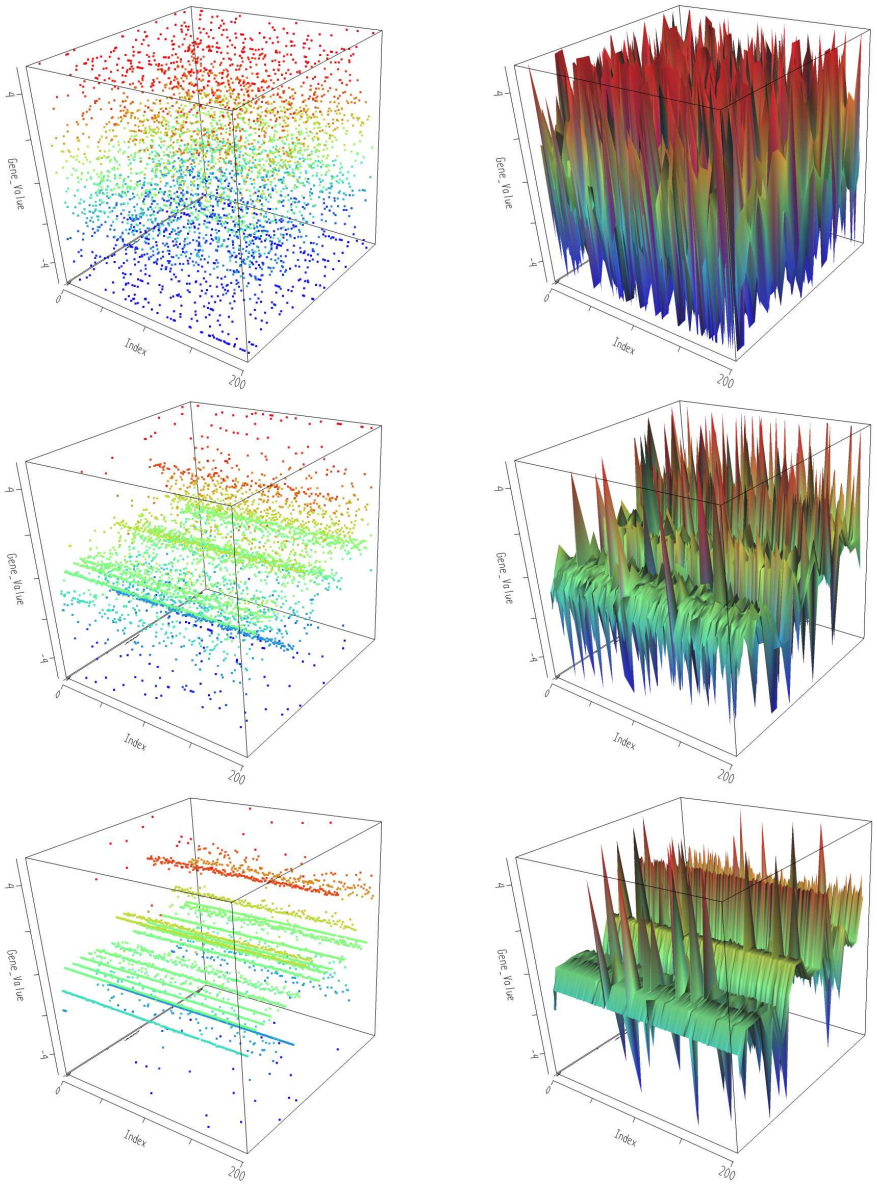
Για την δεύτερη κατηγορία, με την οποία συγκρίνουμε τον πληθυσμό με το καλύτερο άτομό του, χρησιμοποιούμε ένα δισδιάστατο διάγραμμα όπως φαίνεται στην εικόνα 3.6. Στον άξονα $x'x$ έχουμε τα γονίδια και στον άξονα $y'y$ τις τιμές τους. Σχηματίζονται δύο γραφικές παραστάσεις που εκφράζουν τις τιμές του καλύτερου ατόμου και τις μέσες τιμές όλου του υπόλοιπου πληθυσμού.



Σχήμα 3.5: *Δισδιάστατο διάγραμμα εικόνας για όλες τις τιμές των γονιδίων, από έναν ολόκληρο πληθυσμό. Αριστερά, η αρχική γενιά και δεξιά, η πεντηκοστή γενιά.*



Σχήμα 3.6: *Δισδιάστατο διάγραμμα. Με κόκκινο είναι οι τιμές των γονιδίων του καλύτερου και με πράσινο η μέση τιμή των γονιδίων όλου του υπόλοιπου πληθυσμού.*



Σχήμα 3.7: *Αριστερή στήλη, τρισδιάστατο διάγραμμα σημείων και δεξιά στήλη τρισδιάστατο διάγραμμα επιφάνειας, για όλες τις τιμές των γονιδίων, από έναν ολόκληρο πληθυσμό. Απεικονίζονται η αρχική γενιά, η γενιά 15 και 30.*

4

Περιγραφή Απαιτήσεων Περιβάλλοντος

Στα προηγούμενα κεφάλαια, παρουσιάσαμε το θεωρητικό μοντέλο των ΓΑ, καθώς και ένα σύνολο από γενικευμένες γραφικές μεθόδους μελέτης των αποτελεσμάτων και της συμπεριφοράς τους. Ωστόσο η θεωρητική μελέτη από μόνη της δεν επαρκεί για την κατανόηση του τρόπου λειτουργίας του ΓΑ. Η εργασία αυτή δεν περιορίζεται μόνο στη θεωρητική παρουσίαση περί ανάπτυξης και γραφικής αναπαράστασης των ΓΑ, αλλά έχει ως κύριο στόχο την υλοποίηση όλων των θεμάτων που αναπτύχθηκαν στα προηγούμενα κεφάλαια. Δηλαδή, την κατασκευή ενός ολοκληρωμένου περιβάλλοντος ανάπτυξης ΓΑ με δυνατότητες γραφικής αναπαράστασης των δεδομένων που προκύπτουν από την εφαρμογή του αλγόριθμου σε ένα συγκεκριμένο πρόβλημα.

Πρωταρχικό στάδιο για την σχεδίαση ενός προγράμματος ή μίας βιβλιοθήκης είναι η συλλογή των απαιτήσεων. Παρακάτω δίνονται οι απαιτήσεις:

1. Το περιβάλλον να αποτελείται από μία βιβλιοθήκη ΓΑ και από μία εφαρμογή γραφικού περιβάλλοντος που θα αξιοποιεί την βιβλιοθήκη.

2. Ανεξάρτητη από την αρχιτεκτονική του συστήματος (Cross-platform). Η βιβλιοθήκη και η εφαρμογή θα πρέπει να μπορούν να λειτουργούν σε συστήματα τα οποία διαφέρουν στην αρχιτεκτονική (πχ. PowerPC, x86, sparc, mips κα.) και στο λειτουργικό τους σύστημα. (πχ. Linux, Solaris, IRIX, MacOS X, Windows κ.α).

3. Κεντρικό σύστημα ρυθμίσεων του ΓΑ. Ο ΓΑ να παραμετροποιείται από ένα μόνο αρχείο ρυθμίσεων, το οποίο με την σειρά του, πρέπει να είναι εύκολο στη χρήση, κατανοητό από τον άνθρωπο (human readable) και να έχει δυνατότητες επέκτασης.

4. Η βιβλιοθήκη να είναι επεκτάσιμη. Δηλαδή να δίνεται η δυνατότητα συγγραφής κώδικα, από τον προγραμματιστή-χρήστη, με τον οποίο θα επεκτείνονται οι λειτουργίες της βιβλιοθήκης. Η δυνατότητα αυτή θα πρέπει να ισχύει σε όλα τα επίπεδα του ΓΑ (πχ. Τελεστές μετάλλαξης, ανασυνδυασμού, επιλογής κ.α), με εύκολο τρόπο, χωρίς να χρειάζεται η τροποποίηση του κώδικα της ίδιας της βιβλιοθήκης.

5. Η σχεδίαση να είναι αντικειμενοστραφείς (object-oriented). Οπότε, κρίνεται απαραίτητο να υλοποιηθεί σε μία σύγχρονη αντικειμενοστραφή γλώσσα προγραμματισμού.

6. Ταχύτητα. Είναι πολύ σημαντικό η εκτέλεση του αλγόριθμου να είναι γρήγορη, καθώς κατά τη λειτουργία του διατηρείται ένα μεγάλο πλήθος από δομές δεδομένων (γονίδια, άτομα, πληθυσμός, κα.) οι οποίες διαρκώς τροποποιούνται από πολλές επαναλαμβανόμενες μαθηματικές πράξεις. Παράλληλα γίνεται καταγραφή των δομών δεδομένων (χρωμοσώματα, πληθυσμός) του ΓΑ, οπότε εκτελούνται πολλές εντολές εισόδου-εξόδου (I/O).

7. Αναπαραστάσεις. Να υποστηρίζονται όλες οι αναπαραστάσεις που έχουν αναφερθεί. Δηλαδή δυαδική αναπαράσταση (binary representation), αναπαράσταση μετάθεσης (permutation), αναπαράσταση ακεραίων (integer reperesentation) και κινητής υποδιαστολής (floating point representation).

8. Τελεστές επιλογής. Ομοίως, να υποστηρίζονται όλοι οι τελεστές επιλογής που έχουν παρουσιαστεί στο δεύτερο κεφάλαιο.

9. Γενετικοί τελεστές. Ομοίως, να υποστηρίζονται όλοι οι γενετικοί τελεστές που περιγράφτηκαν στο δεύτερο κεφάλαιο.

10. Να υπάρχει η δυνατότητα κάθε στοιχείο του ΓΑ να παραμετροποιείται μεμονωμένα και από γραφικό περιβάλλον.

11. Ο προγραμματιστής να έχει την δυνατότητα να παρακολουθεί τον αλγόριθμο μέσο γεγονότων (events), έτσι ώστε να ξέρει όλες τις καταστάσεις του και να μπορεί να επέμβει ή να εκτελέσει λειτουργίες μεταξύ αυτών.

12. Γραφικές παραστάσεις. Όλες οι γραφικές παραστάσεις που έχουν παρουσιαστεί στο τρίτο κεφάλαιο, δηλαδή της πορείας και της κατάστασης του ΓΑ.

13. Οι τρισδιάστατες γραφικές παραστάσεις να επιταχύνονται από το υλικό (hardware) του συστήματος.

14. Η εφαρμογή θα πρέπει να αξιοποιεί πλήρως τις δυνατότητες της βιβλιοθήκης. Να υπάρχει πλήρης παραμετροποίηση όλων των στοιχείων της βιβλιοθήκης σε μορφή wizard. Επιπλέον να χρησιμοποιείται το ίδιο σύστημα ρυθμίσεων με την βιβλιοθήκη.

15. Ικανότητες φόρτωσης και παραμετροποίησης των επεκτάσεων της βιβλιοθήκης που έχουν γραφτεί από τον χρήστη, από το γραφικό περιβάλλον.

16. Κατά την εκτέλεση του αλγόριθμου να υπάρχει αλληλεπίδραση με τον χρήστη. Δηλαδή να εμφανίζεται ένα διάγραμμα καλύτερης - μέσης τιμής σε πραγματικό χρόνο, ώστε ο χρήστης να ενημερώνεται άμεσα για την εξέλιξη. Επιπλέον, ο χρήστης μπορεί να σταματάει προσωρινά τον αλγόριθμο, με δυνατότητες πρόσβασης σε όλα τα διαγράμματα.

17. Δυνατότητες εξαγωγής σε αρχεία εικόνων, είτε έγχρωμα είτε σε αποχρώσεις του γκρι.

18. Δυνατότητα εξαγωγής των δεδομένων σε αρχεία CSV (Comma Separated Values), έτσι ώστε να μπορούν να χρησιμοποιηθούν από προγράμματα για περαιτέρω στατιστική μελέτη.

Πέραν των απαιτήσεων που αναφέραμε παραπάνω, έναν ακόμα περιορισμό που θέτουμε για την υλοποίηση της εργασίας είναι η χρήση βοηθητικών βιβλιοθηκών, οι οποίες πρέπει να είναι ανοικτού κώδικα (open source) με ελεύθερη άδεια χρήσης.

5

Υλοποίηση

Ο στόχος της εργασίας είναι η κατασκευή ενός περιβάλλοντος για την ανάπτυξη Γενετικών Αλγόριθμων και τη γραφική μελέτη των αποτελεσμάτων και της διαδικασίας εξέλιξης. Ο χρήστης μέσω του περιβάλλοντος αυτού θα μπορεί να επιλύει προβλήματα με ΓΑ καθώς και να το επεκτείνει εύκολα, ανάλογα με τις ανάγκες του. Τα δεδομένα τα οποία παράγονται από τον ΓΑ, θα μπορούν να μελετηθούν με γραφικό τρόπο μέσα από το ίδιο το περιβάλλον. Για τους λόγους αυτούς κρίθηκε απαραίτητο το περιβάλλον να χωριστεί σε δύο τμήματα, ένα για την ανάπτυξη και ένα για την γραφική μελέτη.

Όσον αφορά το πρώτο τμήμα αναπτύχθηκε μία ολοκληρωμένη βιβλιοθήκη Γενετικών Αλγόριθμων, η οποία ονομάζεται GAJLib (από τα αρχικά Genetic Algorithms Java Library). Η βιβλιοθήκη σχεδιάστηκε με σκοπό να είναι εύχρηστη, εύκολα επεκτάσιμη και να ακολουθεί αυστηρά το θεωρητικό μοντέλο, έτσι όπως αυτό παρουσιάστηκε στο τρίτο κεφάλαιο. Σημαντικό ρόλο στη σχεδίαση κατέχει το θέμα της ταχύτητας. Η βιβλιοθήκη σχεδιάστηκε και υλοποιήθηκε με με τέτοιο τρόπο, έτσι ώστε να προσφέρει όλα αυτά τα χαρακτηριστικά χωρίς να υστερεί στον τομέα της ταχύτητας.

Γενικά οι εφαρμογές γραφικής αναπαράστασης (Software Visualization) βασίζονται στις αρχές της τυπογραφίας, των σχεδίων και την μοντέρνας αλληλεπίδρασης μεταξύ ανθρώπου και υπολογιστή, με στόχο να διευκολύνουν τον άνθρωπο να κατανοεί και να χρησιμοποιεί αποδοτικά τον Η/Υ. Η γραφική απεικόνιση δίνει τη δυνατότητα στο χρήστη, να αλληλεπιδρά και να παρατηρεί τα δεδομένα με περισσότερο φυσικό τρόπο. Έτσι λοιπόν για το δεύτερο τμήμα της εργασίας αναπτύχθηκε η εφαρμογή GAJVis (από τα αρχικά Genetic Algorithm Java Visualization), με την οποία ο χρήστης έχει τη δυνατότητα να μελετάει τα αποτελέσματα και όλη την πορεία της εξέλιξης ενός ΓΑ.

5.1 Σχεδίαση

Για την ανάπτυξη αυτής της εργασίας χρησιμοποιήθηκαν ορισμένες βιβλιοθήκες ανοικτού κώδικα. Με μία αναζήτηση στο Διαδίκτυο μπορεί οποιοσδήποτε να βρεθεί μπροστά σε μία πληθώρα από βιβλιοθήκες που εξυπηρετούν ίδιους σκοπούς. Παρόλα αυτά, οι κυριότερες διαφορές που εμφανίζουν μεταξύ τους δεν είναι μόνο η γλώσσα προγραμματισμού με την οποία αναπτύσσονται, αλλά η φιλοσοφία, η σχεδίαση, οι δυνατότητες και η ποιότητα κατασκευής τους. Επιπλέον, πέρα από τις διαφορές που έχουν άμεση ή έμμεση σχέση με την υλοποίησή τους, σημαντικό κριτήριο επιλογής αποτελεί η άδεια χρήσης με την οποία δίνονται. Έτσι λοιπόν οι βιβλιοθήκες θα πρέπει να είναι στα πρότυπα του ανοικτού κώδικα με ελεύθερη άδεια χρήσης, τουλάχιστον για τα πλαίσια μιας πτυχιακής εργασίας.

5.1.1 Περιβάλλον ανάπτυξης και βοηθητικές βιβλιοθήκες

Γλώσσα προγραμματισμού

Πρωταγωνιστικό ρόλο στην ανάπτυξη μίας εφαρμογής κατέχει η γλώσσα προγραμματισμού. Η γλώσσα προγραμματισμού επηρεάζει την εφαρμογή σε όλα τα επίπεδα. Είναι αυτή η οποία επηρεάζει έως ένα βαθμό την σχεδίαση. Παρέχει δυνατότητες στον προγραμματιστή, μέσω της γλώσσας και των βιβλιοθηκών της. Επίσης η γλώσσα καθορίζει και την συμβατότητα σε διαφορετικές πλατφόρμες συστημάτων (πχ. i386/Linux, Sparc/Solaris). Στο σημείο αυτό υπογραμμίζεται και η απαίτηση να μπορεί να εκτελείται η εφαρμογή σε πολλές πλατφόρμες. Γενικότερα υπάρχουν δύο προσεγγίσεις, η μία είναι η εφαρμογή να μπορεί να μεταφράζεται με σχετικά ελάχιστες μετατροπές σε κάθε μία πλατφόρμα ξεχωριστά, όπως για παράδειγμα συμβαίνει με τους GNU compilers (πχ GCC, GCJ) και τις βιβλιοθήκες τους. Η άλλη προσέγγιση είναι να γράφεται ο κώδικας σε μία γλώσσα η οποία εκτελείται σε περιβάλλον διερμηνευτή, από το οποίο καθορίζεται η συμβατότητα.

Στην εργασία αυτή η γλώσσα προγραμματισμού που επιλέχθηκε είναι η Java 2 της Sun Microsystems. Πρόκειται για μία σύγχρονη αντικειμενοστραφή γλώσσα προγραμματισμού που εκτελείται σε περιβάλλον διερμηνευτή με μία πραγματικά πλούσια βιβλιοθήκη. Πίσω από την γλώσσα υπάρχει ένα ολόκληρο framework με ισχυρό διερμηνευτή που είναι σε θέση να κάνει βελτιστοποιήσεις κατά τον χρόνο εκτέλεσης της εφαρμογής. Η Java δεν είναι απλώς μία γλώσσα προγραμματισμού, αλλά μία πλατφόρμα (Java Platform, Standard Edition) που προσφέρει ένα ολοκληρωμένο περιβάλλον ανάπτυξης και εκτέλεσης σε προσωπικούς υπολογιστές και εξυπηρετητές. Πιο συγκεκριμένα χρησιμοποιήθηκε το

πακέτο ανάπτυξης Java Development Kit 5.0 το οποίο περιέχει τον μεταγλωττιστή, τις βιβλιοθήκες και το περιβάλλον εκτέλεσης (Java Runtime Enviroment). Στην έκδοση αυτή εμφανίστηκαν για πρώτη φορά τα Generics και τα Annotations τα οποία χρησιμοποιήθηκαν ευρέως. Τα Generics παραπέμπουν στα πρότυπα (Templates) της γλώσσας C++, με παρόμοια σύνταξη αλλά με διαφορετική φιλοσοφία. Με τα Annotations μπορούν να εισαχθούν μεταδεδομένα (metadata) στις κλάσεις, τα οποία μπορούν να είναι διαθέσιμα ακόμα και κατά το χρόνο εκτέλεσης (run-time) της εφαρμογής.

Ταχύτητα

Όπως έχει επισημανθεί αρκετές φορές στην εργασία, οι ΕΑ είναι αρκετά απαιτητικοί αλγόριθμοι. Κατά την λειτουργία του αλγόριθμου σε κάθε επαναληπτική διαδικασία δημιουργίας γενιάς, σχηματίζονται και καταστρέφονται στη μνήμη, γονίδια, χρωμοσώματα και άλλες βοηθητικές δομές. Επιπλέον μία από τις απαιτήσεις που έχει τεθεί είναι η ταχύτητα εκτέλεσης του αλγόριθμου. Παρόλο που η πλατφόρμα της Java διαθέτει γρήγορο περιβάλλον εκτέλεσης, ωστόσο από μόνη της θεωρείται ανεπαρκής για εφαρμογές πραγματικού χρόνου, εξαιτίας της έλλειψης του ντετερμινισμού, δηλαδή του απρόβλεπτου χρόνου εκτέλεσης. Για παράδειγμα, ο Garbage Collector (GC), ο οποίος αφαιρεί τα αντικείμενα τα οποία δεν χρειάζονται με αυτόματο τρόπο, μπορεί να παγώσει την εφαρμογή για ορισμένα χρονικά διαστήματα. Μία τέτοια συμπεριφορά φυσικά δεν είναι αποδεκτή στον κόσμο των εφαρμογών πραγματικού χρόνου. Για να ξεπεραστούν τα προβλήματα αυτά έχουν αναπτυχθεί διάφορες υλοποιήσεις εικονικών μηχανών Java (Java Virtual Machines, πχ. JVM με Concurrent GC). Επιπλέον έχει ολοκληρωθεί ένα νέο πρότυπο για Java πραγματικού χρόνου, το RTSJ (Real-Time Specification Java, JSR-001). Δυστυχώς, οι προαναφερθείσες λύσεις επιτυγχάνουν τον απαιτούμενο ντετερμινισμό με κόστος την ταχύτητα. Για παράδειγμα, το μοντέλο διαχείρισης μνήμης που προβλέπει το RTSJ, απαιτεί πολλούς ελέγχους κατά τον χρόνο εκτέλεσης. Τέλος, ένα άλλο μειονέκτημα είναι ότι χρειάζεται διαφορετική υλοποίηση της JVM.

Υπάρχει και μία τρίτη λύση για την αντιμετώπιση των προβλημάτων που αφορούν την ταχύτητα και την έλλειψη του ντετερμινισμού κατά τον χρόνο εκτέλεσης. Η βιβλιοθήκη Javolution είναι ανοικτού κώδικα και η μόνη μέχρι στιγμής λύση που πληρεί το πρότυπο RTSJ για όλες τις JVM. Ο GC λειτουργεί όταν στη μνήμη πραγματοποιείται κατανομή (allocation), για παράδειγμα κατά τη δημιουργία και την αρχικοποίηση ενός αντικείμενου ή την διαγραφή του. Επομένως, αν τα «νέα»έτοιμα για χρήση αντικείμενα υπάρχουν, δεν χρειάζεται να πραγματοποιηθεί κατανομή στη μνήμη. Για να επιτευχθεί αυτό αρκεί να υπάρχει μία δομή η οποία θα αποθηκεύει και θα ανακυκλώνει τα αντικείμενα τα οποία παύουν να χρειάζονται, δηλαδή αυτά που ο GC θα διέγραφε. Και κάθε

φορά που θα χρειάζεται ένα νέο αντικείμενο, αντί να δημιουργηθεί, απλώς να επιστραφεί ένα ήδη υπάρχων από αυτήν την δομή. Με τον τρόπο αυτό αυξάνεται σημαντικά η ταχύτητα εκτέλεσης της εφαρμογής, διότι δεν διακόπτεται από τον GC. Επιπλέον η βιβλιοθήκη αυτή δεν περιορίζεται ως εδώ, προσφέρει ένα σύνολο από δομές δεδομένων, XML αναλυτές (parsers) και άλλες εναλλακτικές υλοποιήσεις απ΄ αυτές της Java και οι οποίες είναι κατά πολύ ταχύτερες, από δύο (2) έως και πέντε (5) φορές. Διευκολύνει τον προγραμματισμό παράλληλων συστημάτων και μπορεί να εκτελεστεί σε οποιαδήποτε πλατφόρμα από την απλούστερη J2ME CLDC 1.0 για μικρές συσκευές χωρίς GC έως και την τελευταία έκδοση της J2EE (Java 2 Enterprise Edition).

Γραφική αναπαράσταση

Για την γραφική αναπαράσταση των δεδομένων σε δύο και τρεις διαστάσεις, όπως έχουν περιγραφεί στο 3^o κεφάλαιο , χρησιμοποιήθηκαν οι βιβλιοθήκες VisAD και JChart2D. Το VisAD (Visualization for Algorithm Development) είναι μία βιβλιοθήκη που έχει αναπτυχθεί εξολοκλήρου σε Java για την γραφική παράσταση και ανάλυση αριθμητικών δεδομένων. Χρησιμοποιήθηκε για όλα τα διαγράμματα που απεικονίζουν την πορεία και την κατάσταση του ΓΑ. Η βιβλιοθήκη αυτή διαθέτει τα εξής χαρακτηριστικά:

1. Έχει υλοποιηθεί σε γλώσσα Java, για δυνατότητες ανεξαρτήτου πλατφόρμας εκτέλεσης και υποστήριξης κατανεμημένης λειτουργίας (μέσο Java RMI).

2. Παρέχει ένα γενικό μαθηματικό μοντέλο το οποίο υποστηρίζει οποιασδήποτε μορφής αριθμητικά δεδομένα. Το μοντέλο αυτό, μπορεί να μοιραστεί σε δικτυακό περιβάλλον μεταξύ διαφορετικών πηγών δεδομένων.

3. Διαθέτει ένα γενικό μοντέλο γραφικής αναπαράστασης που υποστηρίζει δισδιάστατα και τρισδιάστατα διαγράμματα, πολλαπλές όψεις δεδομένων, άμεση διαχείριση, εικονική πραγματικότητα κ.α. Το γραφικό μοντέλο αυτό, έχει υλοποιηθεί σε Java3D και Java2D, καθώς και στο σύστημα εικονικής πραγματικότητας ImmersaDesk.

4. Είναι επεκτάσιμη. Μπορεί ο προγραμματιστής εύκολα να γράψει δικές του κλάσεις που προσθέτουν επιπλέον δυνατότητες.

5. Μπορεί να χρησιμοποιηθεί από Java Applet.

6. Μπορεί να χρησιμοποιηθεί άμεσα από την script γλώσσα Jython.

Το VisAD για να εμφανίσει τα τρισδιάστατα διαγράμματα απαιτεί τη βιβλιοθήκη Java3D. Το Java3D είναι μία πολύ ισχυρή βιβλιοθήκη για την δημιουργία τρισδιάστατων γραφικών στο περιβάλλον της Java, που δημιουργήθηκε από την ίδια την Sun Microsystems. Ωστόσο σήμερα είναι ελεύθερη και ανοικτού κώδικα βιβλιοθήκη. Σε λειτουργικά τύπου Unix τα γραφικά επιταχύνονται από το υλικό μέσω του OpenGL ενώ στα Microsoft Windows μέσω Direct3D ή OpenGL. Όσον αφορά τα δισδιάστατα διαγράμματα χρησιμοποιεί το Java2D, το οποίο είναι βιβλιοθήκη διανυσματικών γραφικών και συμπεριλαμβάνεται στην καθιερωμένη βιβλιοθήκη της Java. Το VisAD έχει αναπτυχθεί από προγραμματιστές των:

- SSEC Visualization Project του πανεπιστημίου Wisconsin-Madison, τμήμα Space Science and Engineering
- Unidata Program Center
- National Center of Metereology
- National Center of Supercomputer Applications
- Austrian Bureau of Metereology
- National Center for Atmosphere Research
- Canadian National Research Council

Το JChart2D είναι μία μινιμαλιστική βιβλιοθήκη ανοικτού κώδικα, για την απεικόνιση απλών διαγραμμάτων σε πραγματικό χρόνο. Στην πτυχιακή εργασία χρησιμοποιείται για την γραφική απεικόνιση της μέσης και καλύτερης τιμής, κατά τον χρόνο εκτέλεσης του ΓΑ.

Γραφικό περιβάλλον

Όλο το γραφικό περιβάλλον στηρίζεται στο Swing το οποίο είναι μέρος του περιβάλλοντος της Java. Το Swing είναι σχεδιαστικά μία καθαρά αντικειμενοστραφής βιβλιοθήκη ανάπτυξης γραφικού περιβάλλοντος, η οποία είναι ανεξάρτητη αρχιτεκτονικής συστήματος. Περιέχει μεγάλο πλήθος από συστατικά στοιχεία (components) που συνεργάζονται μεταξύ τους και λειτουργούν με τον ίδιο τρόπο σε όλα τα λειτουργικά συστήματα. Επιπλέον η επικοινωνία όλων των στοιχείων του Swing πραγματοποιείται ασύγχρονα με γεγονότα (events). Το χαρακτηριστικό αυτό διευκολύνει πολύ τον προγραμματιστή να αναπτύξει πολύπλοκα γραφικά περιβάλλοντα. Τέλος, το Swing προσφέρει δυνατότητες παραμετροποίησης της εμφάνισής του (Swing look and feel), για

τον λόγο αυτό χρησιμοποιήθηκε η ελεύθερη βιβλιοθήκη Synthetica (`http://www.javasoft.de/jsf/public/products/synthetica`), έτσι ώστε η εφαρμογή να διαθέτει σε όλα τα λειτουργικά συστήματα την ίδια γραφική εμφάνιση.

5.1.2 Δυνατότητες

Έχοντας ως βάση τις απαιτήσεις που αναφέρθηκαν στο προηγούμενο κεφάλαιο, τις βοηθητικές βιβλιοθήκες και το περιβάλλον ανάπτυξης, παρουσιάζουμε τις δυνατότητες της βιβλιοθήκης GAJLib και της εφαρμογής GAJVis.

Δυνατότητες της GAJLib

Παρακάτω παρουσιάζονται οι δυνατότητες της βιβλιοθήκης GAJLib:

1. Αρχιτεκτονική με βάση τα πρότυπα του Real-Time Specification for Java (RTSJ).
2. Αναπαράσταση (Representation).
   i. Δυαδική
   ii. Ακέραιων αριθμών (Integer και Long, δηλαδή 32 και 64 bits αντίστοιχα)
   iii. Κινητής υποδιαστολής (Float και Double, δηλαδή 32 και 64 bits αντίστοιχα)
   iv. Μετάθεσης
3. Τελεστές ανασυνδυασμού (Recombination).
   i. Γενικοί τελεστές ανασυνδυασμού (ισχύουν για όλες τις αναπαραστάσεις)
      - Single Point
      - N Point
      - Uniform
   ii. Τελεστές ανασυνδυασμού μετάθεσης
      - PMX
      - Cycle
   iii. Τελεστές ανασυνδυασμού για δυαδική αναπαράσταση
      - Bit Single Point

- Bit N Point
- Bit Uniform

iv. Τελεστές ανασυνδυασμού για αναπαραστάσεις κινητής υποσδιαστολής

- BLX-a
- Flat
- Arithmetical
- Extended Line
- Heuristic
- Average

4. Τελεστές μετάλλαξης (Mutation)

i. Γενικοί τελεστές μετάλλαξης

- Random

ii. Τελεστές μετάλλαξης για αναπαράσταση μετάθεσης

- Swap
- Inversion

iii. Τελεστές μετάλλαξης για δυαδική αναπαράσταση

- Bit-Flip

iv. Τελεστές μετάλλαξης για αναπαράσταση αριθμών κινητής υποδιαστολής

- Creep
- Non-Uniform

5. Επιλογή (Selection)

i. Tournament

ii. Rank Roulette Wheel

iii. Cost Roulette Wheel

iv. Stochastic Universal Sampling

v. Random

6. Κλιμάκωση Ποιότητας (Fitness Scale)

i. Windowing

ii. Linear Scaling

iii. Sigma Scaling

iv. Power Law

7. Συνθήκες τερματισμού

   i. Απλές συνθήκες τερματισμού

      - Καλύτερη τιμή (Best Fitness). Δηλαδή όταν ένα άτομο του πληθυσμού αποκτήσει τιμή καλύτερη ή ίση της τιμής που έχει οριστεί, τότε η συνθήκη ικανοποιείται.
      - Μέγιστο πλήθος επαναλήψεων (Max Iterations) του ΓΑ.
      - Μέγιστος χρόνος εκτέλεσης (Max Execution Time) του ΓΑ, σε δευτερόλεπτα.

   ii. Σύνθετες συνθήκες τερματισμού

      - AND Γίνεται αληθής όταν ικανοποιούνται όλες οι συνθήκες που περιέχει.
      - OR Γίνεται αληθής όταν ικανοποιείται τουλάχιστο μία συνθήκη.
      - XOR Γίνεται αληθής όταν ικανοποιείται μόνο μία συνθήκη.

8. Αρχικοποίηση Πληθυσμού (Population Initialization)

   i. Τυχαία (Random). Οι τιμές επιλέγονται τυχαία από μία κατανομή, Uniform, Gaussian, Cauchy ή κάποια άλλη που έχει υλοποιηθεί από τον προγραμματιστή, κατασκευάζοντας δηλαδή ένα Random Number Generator.

   ii. Προσαρμοσμένη (Custom). Οι τιμές ορίζονται με τον τρόπο που επιθυμεί ο προγραμματιστής. Είναι ο μοναδικός τρόπος με τον οποίο μπορούν να εισαχθούν αρχικές τιμές όταν χρησιμοποιείται η αναπαράσταση μετάθεσης.

9. Αρχείο ρυθμίσεων βασισμένο στην XML

10. Εργαλεία

    i. GNUPlot writer. Εξαγωγή των αποτελεσμάτων (μέση και καλύτερη τιμή) σε ASCII αρχείο που διαβάζεται από το πρόγραμμα GNUPlot καθώς και αυτόματη δημιουργία του script που χρειάζεται για να εμφανιστεί το διάγραμμα.

    ii. Console writer. Τα αποτελέσματα (μέση και καλύτερη τιμή) τυπώνονται στο τερματικό.

    iii. Binary decoders. Ένα σύνολο από μεθόδους για την εύκολη αποκωδικοποίηση των ατόμων της δυαδικής αναπαράστασης.

iv. Random number generators. Ένα σύνολο από γεννήτριες τυχαίων αριθμών.

11. Evolution History File. Αρχείο καταγραφής όλων των δεδομένων που παράγονται κατά την διάρκεια εκτέλεσης του ΓΑ.

12. Εύκολη παρακολούθηση του ΓΑ, μέσο γεγονότων (events).

13. Εύκολα επεκτάσιμη. Ο προγραμματιστής μπορεί να επεκτείνει την βιβλιοθήκη σε όλα τα επίπεδα του ΓΑ.

Δυνατότητες του GAJVis

Παρακάτω παρουσιάζονται οι δυνατότητες της εφαρμογής GAJVis:

1. Γραφική απεικόνιση της Πορείας του ΓΑ
   i. Διάγραμμα καλύτερης και μέσης τιμής του ΓΑ
   ii. Δισδιάστατη απεικόνιση τιμών του καλύτερου ατόμου
   iii. Τρισδιάστατη απεικόνιση τιμών του καλύτερου ατόμου
2. Γραφική απεικόνιση της Κατάστασης του ΓΑ
   i. Τιμές ποιότητας όλων των ατόμων
   ii. Δισδιάστατη απεικόνιση των γονιδίων των ατόμων
   iii. Τρισδιάστατη απεικόνιση των γονιδίων των ατόμων
   iv. Μέσες τιμές των γονιδίων μαζί με τις τιμές των καλύτερων
3. Διάγραμμα πραγματικού χρόνου με τις τιμές του καλύτερου και του μέσου κατά την διάρκεια εκτέλεσης του ΓΑ.
4. Δυνατότητα προσωρινής παύσης του ΓΑ με άμεση πρόσβαση σε όλα τα δεδομένα που έχουν καταγραφεί.
5. Configuration Wizard. Ο χρήστης μπορεί εύκολα να δημιουργήσει ένα ΓΑ, μέσα από εύχρηστες διαδικασίες που προσφέρονται από το γραφικό περιβάλλον.
6. Αυτόματη φόρτωση των επεκτάσεων της GAJLib και των συναρτήσεων ποιότητας στο περιβάλλον, έτσι ώστε να υπάρχει πρόσβαση από τον Configuration Wizard.
7. Εξαγωγή δεδομένων σε αρχείο CSV για μελέτη των αποτελεσμάτων από άλλα προγράμματα.

8. Εξαγωγή όλων των γραφικών απεικονίσεων σε αρχεία εικόνας τύπου JPEG, PNG και BMP

5.2 Η βιβλιοθήκη GAJLib

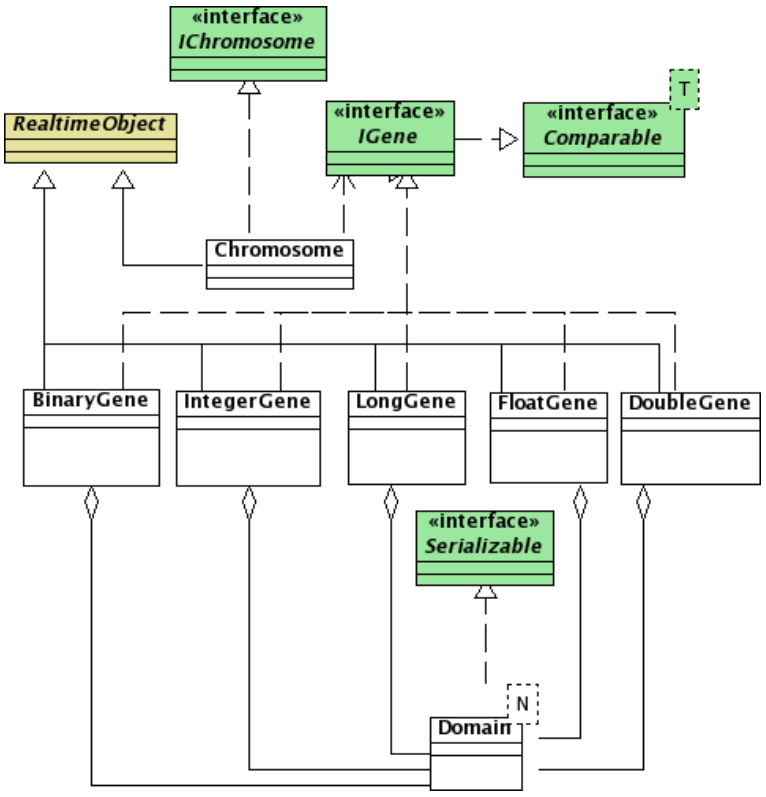
5.2.1 Χρωμόσωμα

Στο επίκεντρο του ΓΑ είναι το χρωμόσωμα. Το χρωμόσωμα αντιπροσωπεύει μία πιθανή λύση και αποτελείται από ένα πλήθος γονιδίων. Στη βιβλιοθήκη GAJLib το χρωμόσωμα περιέχει την τιμή ποιότητας (με μία μεταβλητή τύπου double) και ένα πίνακα από γονίδια. Κάθε γονίδιο είναι ένα αντικείμενο που περιέχει μία τιμή και στο οποίο αντιστοιχίζεται ένα επιτρεπτό εύρος τιμών (Domain). Το χρωμόσωμα μπορεί να περιέχει οποιοδήποτε τύπο ή συνδυασμό γονιδίων, αυτό καθορίζεται από την αναπαράσταση του ΓΑ. Το χρωμόσωμα και όλες οι υλοποιήσεις των γονιδίων που δίνονται μέσα από τη βιβλιοθήκη, είναι κατασκευασμένα σύμφωνα με τα πρότυπα των real-time αντικειμένων, όπως αυτά ορίζονται στη βιβλιοθήκη Javolution. Η βιβλιοθήκη επιτρέπει στον προγραμματιστή να δημιουργήσει δικές του διαφορετικές υλοποιήσεις γονιδίων και χρωμοσωμάτων.

5.2.2 Αναπαράσταση

Για την επίλυση προβλημάτων με ΓΑ, το πρώτο πράγμα που καλούμαστε να αποφασίσουμε, είναι η αναπαράσταση. Είναι αυτή οι οποία επηρεάζει όλα τα στοιχεία του ΓΑ, ακόμη και την συνάρτηση ποιότητας. Από την πλευρά της βιβλιοθήκης η αναπαράσταση (σχήμα 5.2) αποτελεί τον ακρογωνιαίο λίθο. Μέσω της αναπαράστασης δημιουργούνται τα χρωμοσώματα, καθώς είναι αυτή η οποία ορίζει τη δομή των δεδομένων τους. Πολλοί γενετικοί τελεστές εξαρτώνται από την αναπαράσταση, διότι τροποποιούν τις τιμές των γονιδίων.

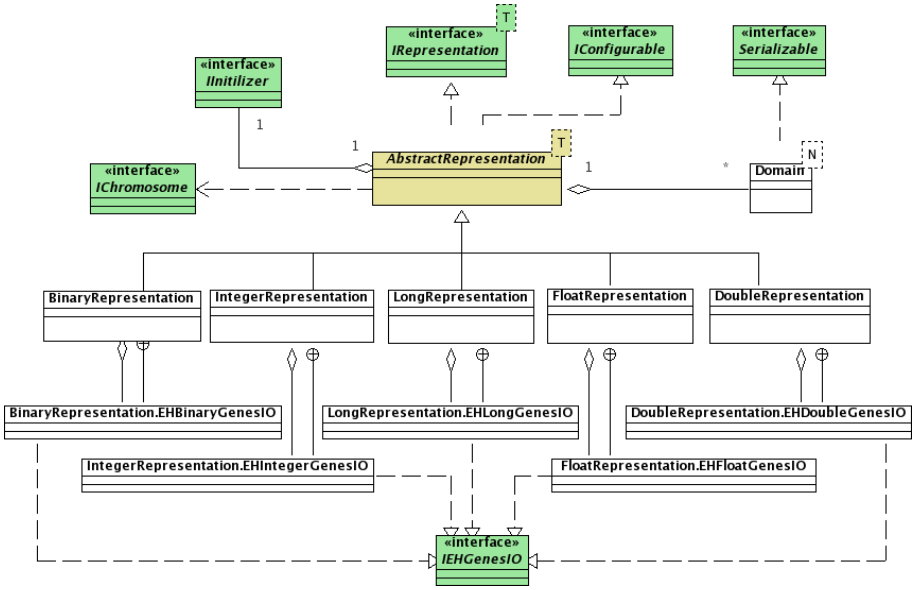
Από την αναπαράσταση μπορούμε να ορίσουμε τα γονίδια του χρωμοσώματος, είτε ένα προς ένα, είτε μαζικά (όταν ισχύουν τα ίδια για όλα). Δηλαδή τον τύπο δεδομένων τους, το επιτρεπτό εύρος τιμών και την ακρίβεια σε δεκαδικά ψηφία. Όμως πέραν του ορισμού της δομής δεδομένων του χρωμοσώματος, το αντικείμενο της αναπαράστασης είναι υπεύθυνο για την αποθήκευση ή την φόρτωση του χρωμοσώματος σε αρχείο. Σχεδόν όλα τα στοιχεία του ΓΑ χρειάζονται πληροφορίες από την αναπαράσταση για να λειτουργήσουν.



Σχήμα 5.1: *Διάγραμμα Τάξεων, χρωμόσωμα και γονίδια.*

5.2.3 Σύστημα Ρυθμίσεων

Χαρακτηριστικό γνώρισμα των ΓΑ είναι ο μεγάλος αριθμός των ρυθμίσεων που χρειάζονται κατά τον σχεδιασμό τους. Πέρα από τις βασικές ρυθμίσεις, όπως για παράδειγμα η πιθανότητα μετάλλαξης ή το μέγεθος του πληθυσμού, πρέπει να οριστούν τα στοιχεία του ΓΑ. Ως στοιχείο του ΓΑ, ορίζουμε κάθε γενετικό τελεστή, τελεστή επιλογής, αναπαράσταση και γενικότερα κάθε οντότητα του ΓΑ και της βιβλιοθήκης, που έχει παρουσία στο αρχείο ρυθμίσεων. Κάθε στοιχείο μπορεί να έχει διαφορετικές παραμέτρους ή να μην έχει καθόλου. Επιπλέον υπάρχει η περίπτωση ένα στοιχείο να περιέχει άλλα υποστοιχεία, τα οποία κι αυτά με τη σειρά τους να χρειάζονται ρυθμίσεις. Παράλληλα είναι πολύ πιθανό, μερικά από αυτά να χρειάζονται πληροφορίες από άλλα στοιχεία, με τα οποία δεν έχουν άμεση σχέση (σχέση στοιχείο-υποστοιχείο). Είναι πολύ σημαντικό ένα αρχείο ρυθμίσεων να παραμένει όσο το δυνατόν κατανοητό από τον άνθρωπο. Δηλαδή να διαβάζεται και να τροποποιείται χωρίς να είναι απαραίτητη η χρήση κάποιου ειδικού προγράμματος. Πάνω στις ιδέες αυτές βασίζεται η σχεδίαση του συστήματος ρυθμίσεων της βιβλιοθήκης GAJLib, το



Σχήμα 5.2: *Διάγραμμα Τάξεων, αναπαραστάσεις.*

οποίο αποτελείται από δύο τμήματα, το αρχείο ρυθμίσεων και τον μηχανισμό παραμετροποίησης.

Το αρχείο ρυθμίσεων είναι ουσιαστικά η εικόνα των ίδιων των στοιχείων της βιβλιοθήκης. Στηρίζεται στο πρότυπο της XML (eXtended Metadata Language) έτσι ώστε να είναι κατανοητό από τον άνθρωπο, ευέλικτο και επεκτάσιμο. Η XML είναι ένα υποσύνολο της SGML (Standard Generalized Markup Langage) και συνιστάται από το W3C (World Wide Web Consortium), ως μία γενικού σκοπού γλώσσα σήμανσης (markup language) για την κατασκευή άλλων εξειδικευμένων, ικανών για την περιγραφή δεδομένων. Η XML είναι ένας τρόπος περιγραφής των δεδομένων, ενώ ταυτόχρονα μπορεί να περιέχει τα ίδια τα δεδομένα, όπως συμβαίνει με μία βάση δεδομένων. Για να περιγράψει την πληροφορία χρησιμοποιεί έννοιες βασισμένες σε κείμενο, οι οποίες εφαρμόζονται σε δενδρική μορφή. Η ιεραρχία με στη οποία εμφανίζονται οι σημάνσεις της XML είναι πανομοιότυπη με τις S-εκφράσεις (S-expressions) της γλώσσας LISP, οι οποίες περιγράφουν δενδρικές δομές, όπου μέσα σ΄ αυτές κάθε κόμβος έχει το δικαίωμα να διαθέτει τις δικές του παραμέτρους. Για τους λόγους αυτούς κρίθηκε απαραίτητη η χρήση της XML για το αρχείο ρυθμίσεων.

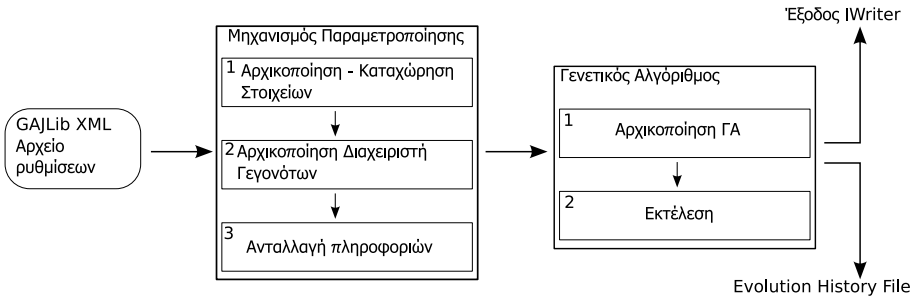
Ο μηχανισμός παραμετροποίησης στη βιβλιοθήκη δεν αποτελεί ένα ενιαίο τμήμα ρυθμίσεων, αλλά περισσότερο στηρίζεται σε μία αποκεντρωμένη λογική. Σύμφωνα με την οποία κάθε στοιχείο είναι το ίδιο υπεύθυνο για το πέρασμα και την καταχώριση των παραμέτρων του. Πιο συγκεκριμένα χρησιμοποιείται ο μηχανισμός σειριακοποίησης (serialization) αντικειμένου από τη βιβλιοθήκη Ja-

volution σε XML. Παρόλα αυτά, ο μηχανισμός προσφέρει ένα κεντρικό σημείο στο οποίο καταχωρούνται όλα τα στοιχεία. Είναι υπεύθυνος και για την μεταξύ ανταλλαγή πληροφορίας από ανεξάρτητα στοιχεία, όταν αυτό είναι επιθυμητό. Ενώ παράλληλα ελέγχει το σύνολο των ρυθμίσεων και είναι υπεύθυνος για την αποθήκευση ή την επαναφορά τους σε αρχείο XML. Παρακάτω περιγράφονται τα τρία στάδια (σχήμα 5.3) του μηχανισμού παραμετροποίησης του ΓΑ:

1. Αρχικοποίηση - Καταχώρηση Στοιχείων. Στο πρώτο στάδιο το αρχείο ρυθμίσεων διαβάζεται ανά στοιχείο. Κάθε στοιχείο φορτώνεται και ρυθμίζεται με βάση τις παραμέτρους που ορίζει μόνο το αρχείο. Όταν ολοκληρώνεται επιτυχώς η διαδικασία αυτή, το στοιχείο καταχωρείται σε μία ενιαία δομή. Αν το τρέχον στοιχείο χρειάζεται πληροφορίες από κάποιο ή κάποια άλλα, τότε σημειώνεται έτσι ώστε στο τρίτο στάδιο να κληθεί ξανά.

2. Αρχικοποίηση Διαχειριστή Γεγονότων. Στο στάδιο αυτό φορτώνεται ο διαχειριστής γεγονότων έτσι ώστε να είναι διαθέσιμος. Όσα από τα στοιχεία χρειάζονται να ενημερώνονται από κάποια γεγονότα κατά τον χρόνο εκτέλεσης του ΓΑ, δηλώνονται στον διαχειριστή.

3. Ανταλλαγή Πληροφοριών. Όσα στοιχεία έχουν σημειωθεί στο πρώτο στάδιο καλούνται ξανά ώστε να ζητήσουν τις πληροφορίες που χρειάζονται (πχ. τύπος αναπαράστασης) από τα υπόλοιπα.



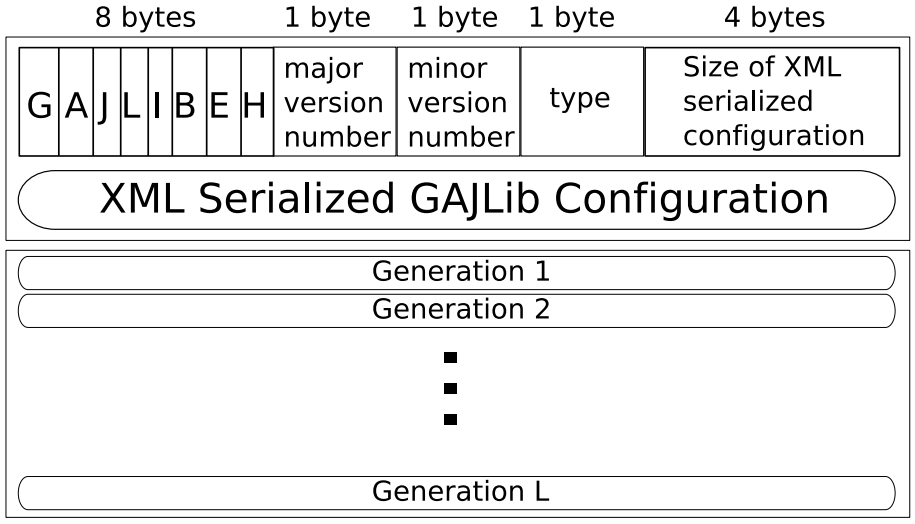
Σχήμα 5.3: *Ρύθμιση και εκτέλεση.*

5.2.4 Αρχείο Evolution History

Κατά την λειτουργία του ΓΑ, ο όγκος των δεδομένων που παράγονται από την επαναληπτική διαδικασία της εξέλιξης είναι αρκετά μεγάλος. Συνεπώς για να γίνει εφικτή η μελέτη τους θα πρέπει να καταγραφούν σε μορφή αρχείου.

Δύο είναι τα σημεία τα οποία χρήζουν ιδιαίτερης σημασίας, η ταχύτητα σε I/O και το μέγεθος του αρχείου. Πιο συγκεκριμένα κατά τη διαδικασία καταγραφής, εκτελούνται εντολές I/O, οι οποίες μπορούν να επιβαρύνουν τόσο πολύ τον αλγόριθμο, ώστε να μειωθούν δραματικά οι επιδώσεις του. Επίσης, δεδομένου ότι ο όγκος των πληροφοριών είναι μεγάλος, τα μόνα στοιχεία στα οποία μπορούμε να εστιάσουμε την προσοχή μας, είναι η μορφή (format) με την οποία αποθηκεύονται οι πληροφορίες και η εσωτερική σχεδίαση του αρχείου.

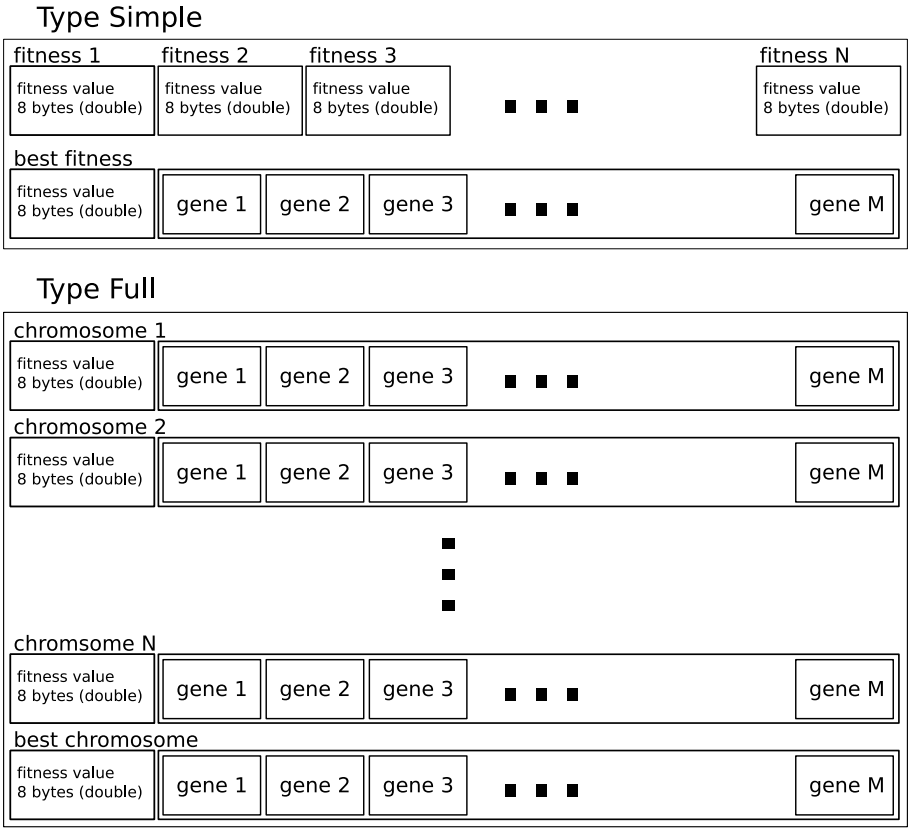
Το αρχείο Evolution History είναι μία απλή μορφή δυαδικού αρχείου για την αποθήκευση όλων των δεδομένων που παράγονται κατά την διάρκεια εκτέλεσης του ΓΑ. Από τα δεδομένα αυτά μπορούμε να εξάγουμε τα διαγράμματα που αφορούν την πορεία και την κατάσταση του ΓΑ, όπως αυτά παρουσιάστηκαν στο 3ο κεφάλαιο. Το αρχείο χωρίζεται σε δύο τμήματα (σχήμα 5.4), την κεφαλίδα (header) και τα δεδομένα του ΕΑ για όλες τις γενιές που εξελίχθηκαν. Η κεφαλίδα του αρχείου περιέχει στα πρώτα 11 bytes τον τίτλο, την έκδοση και τον τύπο του αρχείου. Το υπόλοιπο κομμάτι, περιέχει ένα πεδίο με το μέγεθος του αρχείου ρυθμίσεων σε bytes και ένα άλλο πεδίο με ολόκληρο το αρχείο ρυθμίσεων. Έτσι με τα δεδομένα της κεφαλίδας επιτυγχάνεται η αναγνώριση του αρχείου και ο τύπος της αναπαράστασης



Σχήμα 5.4: *Αρχείο Evolution History.*

Διακρίνουμε δύο τύπους αρχείων, έναν απλό (TYPE_SIMPLE) που περιέχει μόνο τα δεδομένα της πορείας του ΓΑ για λόγους οικονομίας, και έναν ολοκληρωμένο (TYPE_FULL). Οπότε το δεύτερο τμήμα του αρχείου που περιέχει τα δεδομένα του ΕΑ, διαμορφώνεται ανάλογα με τον τύπο. Στην απλή μορφή, όπως διακρίνεται στο σχήμα 5.5 τα δεδομένα που καταχωρούνται σε κάθε γενιά είναι οι τιμές ποιότητας όλων των ατόμων του πληθυσμού και ολό-

κληρο το καλύτερο χρωμόσωμα που έχει βρεθεί. Στην ολοκληρωμένη μορφή, όπως απεικονίζεται στο σχήμα 5.5, τα δεδομένα που καταχωρούνται σε κάθε γενιά είναι ολόκληρα τα χρωμοσώματα όλων των ατόμων του πληθυσμού, με τελευταίο το καλύτερο.



Σχήμα 5.5: *Οι δύο τύποι Evolution History αρχείων.*

Από το σχήμα 5.5, παρατηρούμε ότι στο αρχείο το περιεχόμενο του χρωμοσώματος διαφοροποιείται. Όπως γνωρίζουμε, ο τύπος των δεδομένων στο εσωτερικό των γονιδίων ορίζεται από την αναπαράσταση. Το στοιχείο που γνωρίζει ολόκληρη τη δομή του χρωμοσώματος, είναι η αναπαράσταση. Για τον λόγο αυτό, κάθε αναπαράσταση είναι επίσης υπεύθυνη για τον τρόπο με τον οποίο αποθηκεύονται και διαβάζονται οι πληροφορίες αυτές (σχήμα 5.2).

Για την κατασκευή του αρχείου Evolution History χρησιμοποιείται το πακέτο Java NIO (New Input Output), έτσι ώστε να επιτυγχάνεται η μέγιστη δυνατή ταχύτητα από το περιβάλλον της JVM. Το σημαντικότερο που προσφέρει αυτό το πακέτο, είναι το I/O κατά τμήματα (block). Δηλαδή οι I/O λειτουργίες, εφαρμόζονται (όταν είναι δυνατό) σε μεγάλα τμήματα δεδομένων

με ένα μόνο βήμα, αντί ενός byte ή χαρακτήρα κάθε φορά. Συνεπώς με τη χρήση του Java NIO πακέτου μειώνεται το πλήθος των I/O.

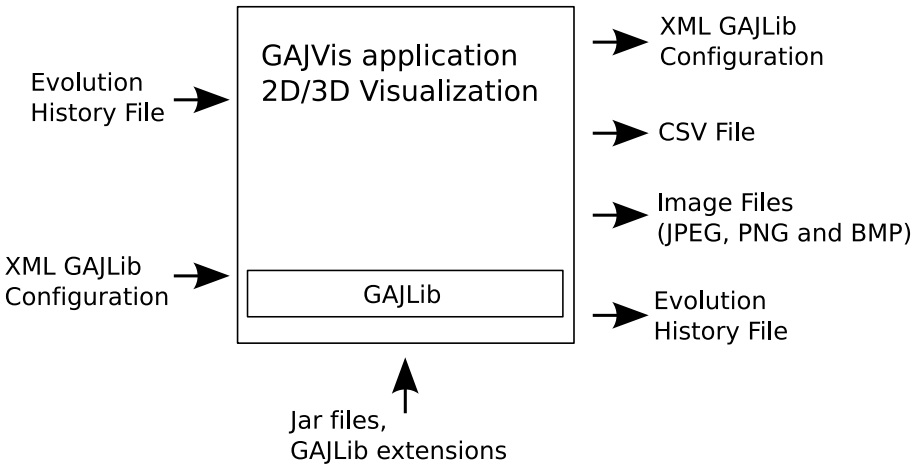
Όσον αφορά την μορφή με την οποία καταγράφονται οι πληροφορίες, έχει ήδη αναφερθεί ότι επιλέχθηκε η δυαδική μορφή (binary format). Με τη δυαδική μορφή, η πληροφορία (π.χ. η τιμή της ποιότητας του χρωμοσώματος) δεν χρειάζεται να κωδικοποιηθεί σε άλλο τύπο (π.χ. ASCII αναπαράσταση). Δηλαδή δεν είναι απαραίτητη η κωδικοποίηση και η αποκωδικοποίηση της πληροφορίας, με αποτέλεσμα να κερδίζουμε ταχύτητα. Επιπλέον καταφέρνουμε να μην επιβαρύνουμε κι άλλο το μέγεθος του αρχείου, διότι εάν καταχωρούσαμε την πληροφορία σε ASCII μορφή, θα χρειαζόμασταν περισσότερα bytes για να περιγράψουμε ένα αριθμό σε σχέση με τη φυσική του μορφή (native format).

Υπάρχει βέβαια και αντίλογος στη χρήση αρχείων δυαδικής μορφής. Η μορφή αναπαράστασής τους είναι κατανοητή μόνο από τον Η/Υ. Για την ανάγνωση των πληροφοριών τους, ο μόνος τρόπος είναι η χρήση ειδικευμένων προγραμμάτων, το αντίθετο δηλαδή με το αρχείο ρυθμίσεων. Φυσικά οι στόχοι των δύο αρχείων είναι παντελώς διαφορετικοί. Όμως το παραπάνω μειονέκτημα ξεπερνιέται εύκολα, καθώς η σχεδίαση του αρχείου είναι ανοικτή και ταυτόχρονα πολύ απλή. Επομένως μπορούν να γραφτούν πολύ εύκολα προγράμματα που να αναγνωρίζουν την μορφή του Evolution History αρχείου.

5.3 Η εφαρμογή GAJVis

Η εφαρμογή GAJVis αποτελεί ένα περιβάλλον γραφικής αναπαράστασης και μελέτης του συνόλου του ΓΑ. Συμπληρώνει τη βιβλιοθήκη GAJLib, κάνοντας περισσότερο εύκολη τη χρήση των ΓΑ. Μέσα από το περιβάλλον αυτό, ο χρήστης έχει τη δυνατότητα, να εκτελέσει τα πειράματά του και να μελετήσει τα αποτελέσματα ακόμη και κατά τη διάρκεια της εξέλιξης. Οι πληροφορίες απεικονίζονται σε δύο και τρεις διαστάσεις, με πολλές δυνατότητες γραφικής εξερεύνησης. Επιπλέον ο χρήστης μπορεί να ρυθμίσει όλα τα στοιχεία του ΓΑ, αλλά και των δικών του υλοποιήσεων, μέσα από ένα εύχρηστο και δυναμικό γραφικό περιβάλλον.

Για να γίνουν πραγματικότητα όλα τα παραπάνω, χρειάστηκε να αναπτυχθεί ένας μηχανισμός διαχείρισης των επεκτάσεων της βιβλιοθήκης GAJLib, ένα διάγραμμα πραγματικού χρόνου, αρκετά διαγράμματα που απεικονίζουν όλα τα δεδομένα του Evolution History, ένας δυναμικός wizard και πολλά components που χαρίζουν στην εφαρμογή περισσότερη ευχρηστία. Γενικότερα η εφαρμογή χρησιμοποιεί όλες τις δυνατότητες της βιβλιοθήκης και βασίζεται επάνω στο αρχείο Evolution History.



Σχήμα 5.6: *Genetic Algorithm Java Visualization application*

5.3.1 Μηχανισμός διαχείρισης των Επεκτάσεων

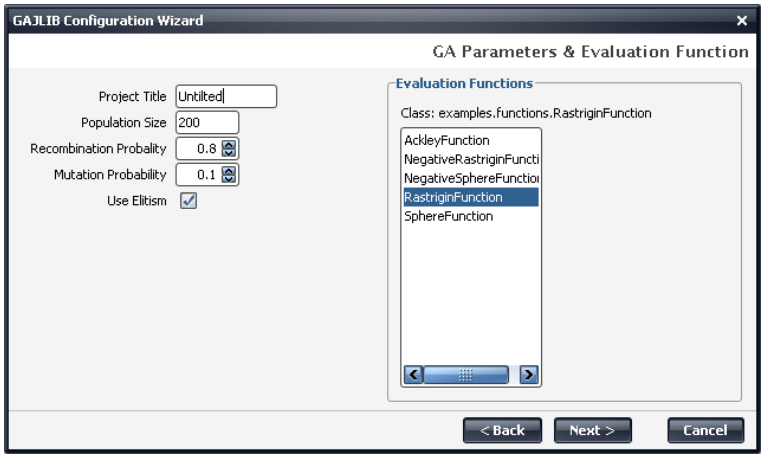
Στο επίπεδο της ρύθμισης και εκτέλεσης των πειραμάτων, το σημαντικότερο ρόλο κατέχει ο μηχανισμός διαχείρισης των επεκτάσεων. Είναι ένας εσωτερικός μηχανισμός, σύμφωνα με τον οποίο όλες οι επεκτάσεις του χρήστη (π.χ. τελεστές συνδυασμού, συναρτήσεις ποιότητας κ.α), εμφανίζονται στον wizard των ρυθμίσεων. Μάλιστα, στην περίπτωση όπου οι επεκτάσεις αυτές απαιτούν ρυθμίσεις από το γραφικό περιβάλλον, θα πρέπει να υλοποιούν τα κατάλληλα Interfaces και Annotations της βιβλιοθήκης. Με τον τρόπο αυτό, εμφανίζονται τα γραφικά components των επεκτάσεων, έτσι ώστε η παραμετροποίηση τους να γίνεται μέσα στο ίδιο το περιβάλλον. Επίσης ο χρήστης θα πρέπει να έχει συγκεντρώσει τις επεκτάσεις μέσα σε αρχεία Jar και να τα τοποθετήσει στο κατάλογο extensions. Έτσι λοιπόν κατά την εκκίνηση της εφαρμογής, ανοίγονται όλα τα αρχεία τύπου Jar και σαρώνονται όλες οι κλάσεις, ώστε να αναγνωριστούν αυτές που επεκτείνουν την βιβλιοθήκη.

5.3.2 Ρύθμιση και Εκτέλεση Πειραμάτων

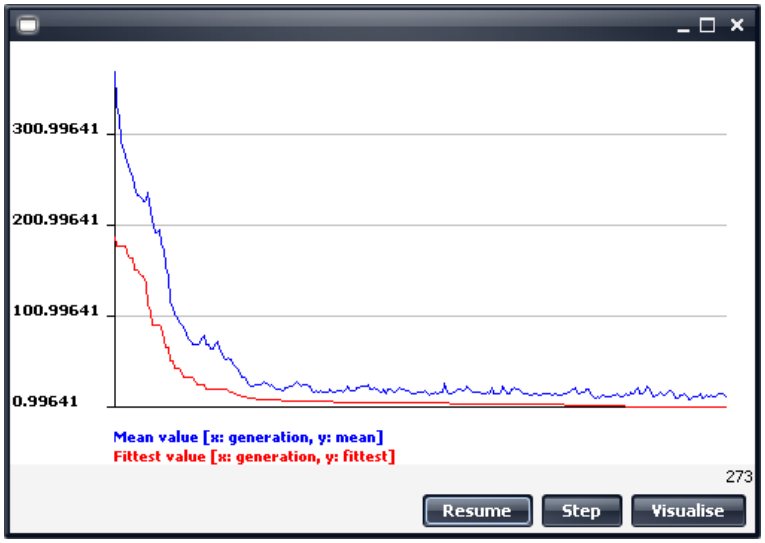
Η ρύθμιση των πειραμάτων από το γραφικό περιβάλλον, βασίζεται στον μηχανισμό διαχείρισης των επεκτάσεων που περιγράψαμε προηγουμένως. Η εφαρμογή GAJVis διαθέτει ένα wizard (εικόνα 5.7), ο οποίος λειτουργεί με δυναμικό τρόπο. Γενικά σε ένα wizard, συναντάμε ένα σύνολο από καρτέλες, οι οποίες εμφανίζονται με κάποια σειρά. Η σειρά αυτή αλλάζει ανάλογα με τις επιλογές της προηγούμενης καρτέλας. Επιπλέον, είναι δυνατό το περιεχόμενο μίας καρτέλας, να επηρεάζεται από τα δεδομένα των προηγούμενων.

Με τον τρόπο αυτό λειτουργεί και ο wizard της εφαρμογής. Όμως το ιδιαί-

τερο χαρακτηριστικό που διαθέτει, είναι η δυναμική εμφάνιση των περιεχομένων του. Δηλαδή ανάλογα με το επιλεγόμενο στοιχείο του ΓΑ, αλλάζει το περιεχόμενο της καρτέλας, έτσι ώστε να εμφανίζονται οι ιδιαίτερες ρυθμίσεις του στοιχείου. Για τις ρυθμίσεις αυτές και τον έλεγχό τους, είναι υπεύθυνο το ίδιο το στοιχείο, όπως συμβαίνει και με τις παραμέτρους στο XML αρχείο ρυθμίσεων της βιβλιοθήκης GAJLib. Κατά την εισαγωγή των παραμέτρων σε κάθε στοιχείο είναι δυνατή η εμφάνιση οπουδήποτε αντικειμένου της βιβλιοθήκης Swing.



Σχήμα 5.7: *Wizard ρυθμίσεων*



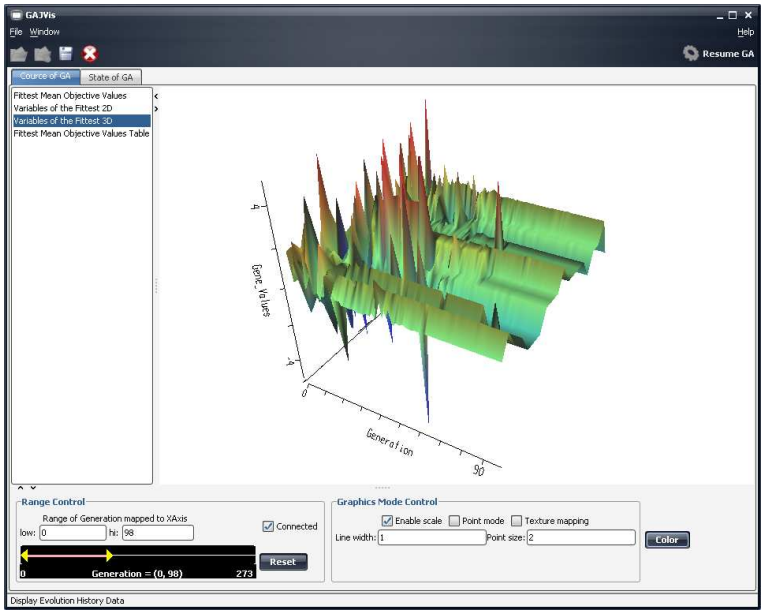
Σχήμα 5.8: *Διάγραμμα πραγματικού χρόνου*

Μετά την διαδικασία των ρυθμίσεων ακολουθεί η εκτέλεση του ΓΑ, η οποία εκτελείται από την βιβλιοθήκη GAJLib. Κατά την διάρκεια της εκτέλεσης, ο χρήστης μπορεί να παρακολουθήσει σε πραγματικό χρόνο την πορεία της εξέλιξης, μέσα από ένα διάγραμμα (εικόνα 5.8) που απεικονίσει την μέση και την καλύτερη τιμή του πληθυσμού ανά γενιά. Επιπλέον ο χρήστης μπορεί να σταματήσει προσωρινά την εκτέλεση του αλγόριθμου και να μελετήσει όλα τα διαγράμματα της 'Πορείας του ΓΑ' και της 'Κατάστασης του ΓΑ'.

5.3.3 Γραφική Αναπαράσταση Δεδομένων

Η εφαρμογή σχεδιάστηκε έτσι ώστε να εκμεταλλεύεται πλήρως την δομή του Evolution History της βιβλιοθήκης GAJLib. Απεικονίζει όλες τις γραφικές μεθόδους που παρουσιάστηκαν στο 3ο κεφάλαιο, με τον πλέον εύχρηστο τρόπο. Όσον αφορά το θέμα της ευχρηστίας κατασκευάστηκαν components, όπου μερικά από αυτά βασίζονται στα ήδη υπάρχοντα των SWING και VisAD. Επιπλέον για αποδοτικότερη λειτουργία της εφαρμογής, αρκετά τμήματά της εκτελούνται σε ξεχωριστά νήματα επεξεργασίας (Threads) και η πρόσβαση στα δεδομένα του αρχείου είναι τυχαίας προσπέλασης. Πιο συγκεκριμένα ο μηχανισμός με τον οποίο το αρχείο φορτώνεται στη μνήμη προέρχεται από την βιβλιοθήκη GAJLib. Ο μηχανισμός αυτός εκμεταλλεύεται αποτελεσματικά τη μνήμη του συστήματος, καθώς λειτουργεί με σκοπό να ελαττώσει όσο το δυνατό το πλήθος των εντολών I/O.

Η γραφική αναπαράσταση (εικόνα 5.9) χωρίζεται σε δύο ομάδες, την «Πορεία του ΓΑ»και την «Κατάσταση του ΓΑ». Κατά την χρήση του προγράμματος, ο χρήστης έχει την δυνατότητα να μελετήσει τα αποτελέσματα με γραφικό τρόπο, μέσω δισδιάστατων και τρισδιάστατων διαγραμμάτων, ή να περιηγηθεί με άμεσο τρόπο στα αριθμητικά δεδομένα. Επιπλέον οι πληροφορίες αυτές μπορούν να εξαχθούν σε αρχεία εικόνας ή σε αρχείο δεδομένων τύπου CSV.



Σχήμα 5.9: *Γραφική αναπαράσταση δεδομένων*

6

Συμπεράσματα και Περαιτέρω Ανάπτυξη

Στα προηγούμενα κεφάλαια παρουσιάσαμε την θεωρία των ΕΑ, επικεντρώνοντας την προσοχή μας στους ΓΑ. Ασχοληθήκαμε με την γραφική μελέτη τους, απεικονίζοντας τη δομή των λύσεων και τις περιοχές του χώρου στις οποίες επικεντρώνεται η εξερεύνηση, με στόχο την ευκολότερη κατανόηση της συμπεριφοράς του αλγόριθμου. Τέλος, σχεδιάστηκε και κατασκευάστηκε μία βιβλιοθήκη και μία εφαρμογή που πραγματοποιούν όλα αυτά. Έτσι λοιπόν ολοκληρώνοντας την εργασία αυτή, θεωρούμε απαραίτητο να αναπτύξουμε τα συμπεράσματα και τις ιδέες για περαιτέρω βελτίωση και προσθήκη νέων δυνατοτήτων.

Η βιβλιοθήκη σχεδιαστικά και λειτουργικά βρίσκεται κοντά στο θεωρητικό-μαθηματικό μοντέλο των ΓΑ. Το γεγονός αυτό οφείλεται στην καθαρά αντικειμενοστραφής σχεδίαση η οποία μιμείται το μαθηματικό μοντέλο. Σε σχέση με άλλες υπάρχουσες βιβλιοθήκες καταφέρνει να υποστηρίζει ταυτόχρονα και με τον ίδιο τρόπο χρήσης πολλές διαφορετικές αναπαραστάσεις. Παράλληλα δίνει την δυνατότητα στον προγραμματιστή να μπορέσει να την επεκτείνει σε όλα τα επίπεδά της. Τα δυνατά χαρακτηριστικά της είναι η μεγάλη πληθώρα γενετικών τελεστών, το σύστημα ρυθμίσεων, η γρήγορη ταχύτητα εκτέλεσης του ΓΑ και η δυνατότητα εκτέλεσης σε διαφορετικές αρχιτεκτονικές Η/Υ συστημάτων. Όμως δεν περιοριζόμαστε εδώ. Το περιβάλλον επιπλέον προσφέρει και μία εφαρμογή γραφικού περιβάλλοντος η οποία επεκτείνει τις δυνατότητες της βιβλιοθήκης. Με την εφαρμογή αυτή, ο χρήστης έχει την δυνατότητα να ρυθμίσει και να εκτελέσει τα πειράματά του από γραφικό περιβάλλον, καθώς επίσης κει να μελετήσει τα αποτελέσματα και τη συμπεριφορά του ΓΑ. Συνεπώς, ο συνδυασμός της βιβλιοθήκης και της γραφικής εφαρμογής, διευκολύνει την ανάπτυξη και την μελέτη των ΓΑ, ενώ ταυτόχρονα βοηθάει σε μεγάλο βαθμό στην κατανόηση της διαδικασίας της εξέλιξης.

Παρόλο που δόθηκε ιδιαίτερη έμφαση στις δυνατότητες και στην ταχύτητα, το περιβάλλον επικεντρώνεται μόνο στους σειριακούς ΓΑ. Πέραν των σειριακών υπάρχουν και οι παράλληλοι ΓΑ, οι οποίοι έχουν ως στόχο την ταχύτερη εκτέλεση και την αποτελεσματικότερη αναζήτηση λύσεων. Επιπλέον η γλώσσα προγραμματισμού όπως αναφέραμε είναι η Java. Η γλώσσα αυτή παρέχει το ιδανικό περιβάλλον για εφαρμογές παράλληλης και κατανεμημένης επεξεργασίας.

Όπως έχει παρουσιαστεί στο 2^o κεφάλαιο, υπάρχουν κι άλλες μεθοδολογίες ΕΑ, συνεπώς το περιβάλλον μπορεί να επεκταθεί σ' αυτές (π.χ. Εξελικτικές Στρατηγικές). Στη περίπτωση αυτή θα πρέπει να υποστηριχθούν διαφορετικές αναπαραστάσεις, τόσο στο επίπεδο της βιβλιοθήκης όσο και στο επίπεδο της εφαρμογής, όπως για παράδειγμα δένδρικές δομές δεδομένων (Γενετικός Προγραμματισμός).

Κλείνοντας, ένα πραγματικό πρόβλημα μπορεί να απαιτεί την βελτίωση περισσότερων του ενός κριτηρίων. Για την επίλυση τέτοιων προβλημάτων έχουν αναπτυχθεί πολλές διαφορετικές μορφές ΓΑ πολλαπλών κριτηρίων (Multiple Objective GA). Η βιβλιοθήκη θα μπορούσε να επεκταθεί και να υποστηρίξει επίλυση προβλημάτων βελτιστοποίησης πολλαπλών κριτηρίων.

Α′

Οδηγίες Εγκατάστασης

Στο CD εγκατάστασης περιέχεται ο πηγαίος κώδικας και η μεταγλωττισμένη (σε bytecode) έκδοση του περιβάλλοντος. Ο χρήστης μπορεί είτε να μεταγλωττίσει τον κώδικα είτε να χρησιμοποιήσει την έτοιμη έκδοση. Η βιβλιοθήκη GAJLib και η εφαρμογή GAJVis χρειάζονται για την μεταγλώττιση και την εκτέλεσή τους τουλάχιστον το Java 2 SDK 5.0 SE (προτείνεται να χρησιμοποιείται πάντοτε η τελευταία έκδοση) το οποίο μπορεί να βρεθεί στην διεύθυνση http://java.sun.com/javase/downloads/ και οι παρακάτω βιβλιοθήκες:

- Javolution v3.7.10 (http://www.javolution.org).
- VisAD v2.0 [build date Wed Feb 15 14:21:41 CST 2006] (http://www.ssec.wisc.edu/ billh/visad.html).
- JChart2D v1.03 (http://jchart2d.sourceforge.net).
- Java3D 1.4.0 (https://java3d.dev.java.net).
- JAMA 1.0.2 (http://math.nist.gov/javanumerics/jama/).
- Synthetica LnF 1.3 (http://www.javasoft.de/jsf/public/products/synthetica).
- TableLayout (https://tablelayout.dev.java.net).
- JavaHelp 2.0_02 (http://java.sun.com/products/javahelp/).

Επιπλέον για την ευκολότερη μεταγλώττιση προτείνεται η χρήση του Apache Ant (http://ant.apache.org/), έκδοση 1.6.5 και άνω.

Α΄.1 Μεταγλώττιση πηγαίου κώδικα και εγκατάσταση

Ο πηγαίος κώδικας της πτυχιακής υπάρχει στον κατάλογο ProjectGAJ, όπου περιέχει τα εξής:

- Κατάλογος GAJLib. Περιέχει τον πηγαίο κώδικα της βιβλιοθήκης GAJLib στον υποκατάλογο src, καθώς και όλες τις μεταγλωττισμένες κλάσεις στον υποκατάλογο classes.

- Κατάλογος GAJLib_examples. Περιέχει τον πηγαίο κώδικα των παραδειγμάτων της βιβλιοθήκης GAJLib στον υποκατάλογο src, καθώς και όλες τις μεταγλωττισμένες κλάσεις στον υποκατάλογο classes.

- Κατάλογος GAJVis. Περιέχει τον πηγαίο κώδικα της εφαρμογής GAJVis στον υποκατάλογο src, καθώς και όλες τις μεταγλωττισμένες κλάσεις στον υποκατάλογο classes.

- Κατάλογος GAJVis_help. Περιέχει τις οδηγίες χρήσης του GAJVis σε μορφή JavaHelp.

- Τα αρχεία build.xml και projectgaj.properties του Apache Ant.

- Κατάλογος libs. Περιέχει όλες τις βιβλιοθήκες που χρειάζονται τα GAJVis και GAJLib.

- Κατάλογος scripts. Περιέχει τα αρχεία κελύφους για λειτουργικά τύπου UNIX και Windows.

Για την μεταγλώττιση της βιβλιοθήκης GAJLib και της εφαρμογής GAJVis, ο προγραμματιστής είναι ελεύθερος να χρησιμοποιήσει οποιοδήποτε περιβάλλον ανάπτυξης (π.χ. Idea IntelliJ, Netbeans, Eclipse κ.α) ή ακόμα και κάποιον απλό συντάκτη κειμένων όπως ο vi σε λειτουργικά συστήματα UNIX. Επιπλέον, για μεγαλύτερη ευκολία στη μεταγλώττιση συνιστάται η χρήση του εργαλείου Apache Ant. Το Apache Ant είναι ένα εργαλείο αυτοματοποιημένης κατασκευής για το περιβάλλον της Java, όπως τα make, gnumake, nmake κ.α. Η χρήση του έχει καθιερωθεί και υποστηρίζεται σχεδόν από όλα τα γνωστά περιβάλλοντα ανάπτυξης. Έτσι λοιπόν, με την προϋπόθεση ότι έχουμε εγκαταστήσει το Ant, για την μεταγλώττιση του κώδικα πραγματοποιούμε τα εξής:

- Από το Command Line Interface (CLI) του λειτουργικού συστήματος, μεταφερόμαστε μέσα στον κατάλογο ProjectGAJ.

- Στη συνέχεια εκτελούμε το Ant, δίνοντας απλώς την εντολή ant f build.xml dist. Στο σημείο αυτό μεταγλωττίζονται η βιβλιοθήκη GAJLib, η εφαρμογή GAJVis και τα παραδείγματα GAJLib_examples. Στη συνέχεια δημιουργείται ο κατάλογος GAJ και μέσα σε αυτόν αντιγράφονται οι βιβλιοθήκες, η εφαρμογή και τα παραδείγματα σε μορφή Jar αρχείων. Τέλος αντιγράφονται και τα βοηθητικά script εκτέλεσης της εφαρμογής.

Συνεπώς, με μία μόνο εντολή κατασκευάζεται ολόκληρη η διανομή του περιβάλλοντος. Όσον αφορά την εγκατάσταση απλός αντιγράφουμε τον κατάλογο GAJ όπου επιθυμούμε, αρκεί να φροντίσουμε να έχουμε δικαιώματα ανάγνωσης και εγγραφής.

Α΄.2 Εγκατάσταση μεταγλωττισμένου κώδικα

Εφόσον έχουμε εγκατεστημένα στον υπολογιστή μας το Java2 SDK 5.0 SE και το Java3D 1.4.0, μπορούμε να προχωρήσουμε στην διαδικασία της εγκατάστασης. Για την εγκατάσταση απλώς αντιγράφουμε τον κατάλογο GAJ από το CD της πτυχιακής στον σκληρό δίσκο του συστήματος και φροντίζουμε να έχουμε δικαιώματα ανάγνωσης όλων των αρχείων του.

Α΄.3 Εγκατάσταση Επεκτάσεων

Για τις επεκτάσεις συνιστάται η κατασκευή τους σε μορφή αρχείου jar. Στον κατάλογο ext θα πρέπει να περιέχονται οι επεκτάσεις είτε σε μορφή jar είτε σε μορφή αρχείων class. Επιπλέον θα πρέπει να δηλωθούν στο classpath της εφαρμογής. Για τα λειτουργικά συστήματα τύπου UNIX και Windows αρκεί να δηλώσουμε το classpath στο πεδίο GAJLIB_EXT (π.χ. GAJLIB_EXT="ext/gajlib_examples.jar") μέσα στα αρχεία κελύφους, τα οποία δίνονται μαζί με την εφαρμογή. Δηλαδή τα αρχεία gajvis, evolve και ehinfo για περιβάλλον κελύφους Bash των λειτουργικών συστημάτων τύπου UNIX και gajvis.bat, evolve.bat και ehinfo.bat για τα Windows.

Β′

Οδηγίες χρήσης

Β′.1 Εκτέλεση Γενετικών Αλγορίθμων

Μέσα στον κατάλογο της εφαρμογής υπάρχουν τα προγράμματα κελύφους evolve, gajvis και ehinfo για λειτουργικά τύπου UNIX με κέλυφος bash. Επιπλέον υπάρχουν και τα αντίστοιχα αρχεία με κατάληξη .bat για λειτουργικά Windows.

Για την εκτέλεση ΓΑ σε περιβάλλον κελύφους χρησιμοποιείται το script evolve ή evolve.bat. Με *evolve -h* τυπώνεται ο σύντομος οδηγός χρήσης όπως φαίνεται παρακάτω:

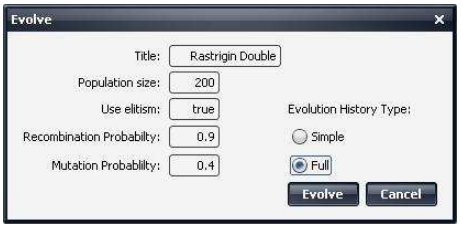
```
GAJLib v0.1 (2005-2006)
Anastasios Skarlatidis

Usage:
 evolve -f=<file> [--disable-all-writers] [-type=<simple or full>] conf.xml
 or evolve [--disable-all-writers] file.xml
 or evolve -check [--disable-all-writers] file.xml
 or evolve -h
 Where file.xml is a GAJLib XML configuration file.
Options:
 -h                     Display usage information.
 -f=<file>              Store Evolutionary History data into the specified
                        <file>.
 -type=<simple or full> Specify Evolutionary History file type
                        (Default type=simple).
 --disable-all-writers  Disable all the configuration's specified IWriter
                        implementations.
 -check                 Check if the configuration file (conf.xml) is correct.
```

Έστω ότι το αρχείο ρυθμίσεων ονομάζεται conf.xml. Τότε για να εκτελέσουμε τον ΓΑ πληκτρολογούμε *evole conf.xml*. Αν όμως θέλουμε να καταγραφούν οι πληροφορίες της εξέλιξης σε ένα αρχείο Evolution History (EH)

με ονομασία out.eh, τότε πληκτρολογούμε *evolve -f=out.eh conf.xml*. Στο σημείο αυτό θα πρέπει να τονίσουμε ότι, για να εκτελέσουμε ένα πρόβλημα θα πρέπει να το έχουμε δηλώσει στο πεδίο GAJLIB_EXT του αρχείου evolve 'h evolve.bat. Επιπλέον οι κλάσεις ή το αρχείο jar του προβλήματος, πρέπει να βρίσκονται μέσα στον κατάλογο ext.

Μπορούμε να εκτελέσουμε ένα πρόβλημα από το γραφικό περιβάλλον. Πρώτα απ΄ όλα χρειάζεται να το δηλώσουμε στο πεδίο GAJLIB_EXT του αρχείου gajvis ή gajvis.bat, τις κλάσεις ή το αρχείο jar του προβλήματος, τα οποία θα πρέπει να βρίσκονται στον κατάλογο ext. Στη συνέχεια εκτελούμε το script gajvis (ή gajvis.bat) και ανοίγουμε ένα αρχείο ρυθμίσεων επιλέγοντας *File → Open Configuration*. Αν το αρχείο ρυθμίσεων είναι σωστό θα εμφανιστεί το παράθυρο διαλόγου (εικόνα Β΄.1) από το οποίο επιλέγουμε τον τύπο του EH αρχείου.



Σχήμα Β΄.1: *Παράθυρο επιλογής τύπου αρχείου EH*

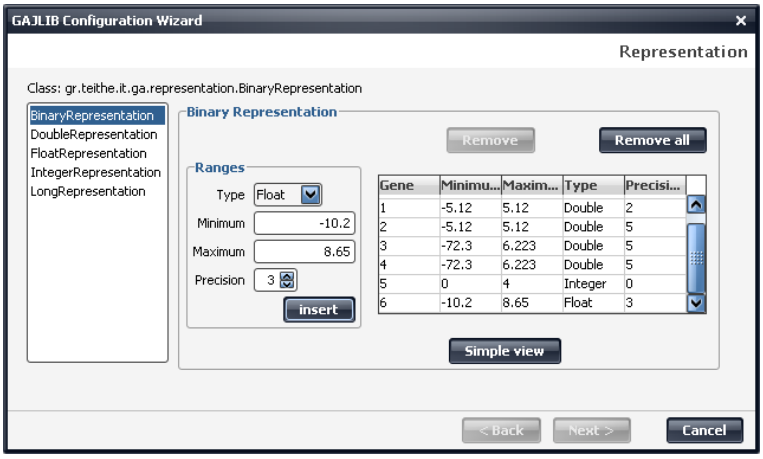
Στο γραφικό περιβάλλον έχουμε την δυνατότητα να εκτελέσουμε ένα πρόβλημα χωρίς να απαιτείται η σύνταξη αρχείου ρυθμίσεων. Αυτό επιτυγχάνεται με την χρήση του wizard. Επιλέγουμε *File→New GA* και εμφανίσεται ο wizard (εικόνα Β΄.2) δημιουργίας ρυθμίσεων. Μετά την εκτέλεση του προβλήματος, μπορούμε να εξάγουμε τις ρυθμίσεις που επιλέξαμε σε ένα αρχείο ρυθμίσεων GAJLib XML, επιλέγοντας *File→Save Configuration*.

Τέλος το script ehinfo (ή ehinfo.bat) δέχεται ως παράμετρο ένα αρχείο EH και τυπώνει βοηθητικές πληροφορίες γι΄ αυτό.

Β΄.2 Διαγράμματα στο GAJVis

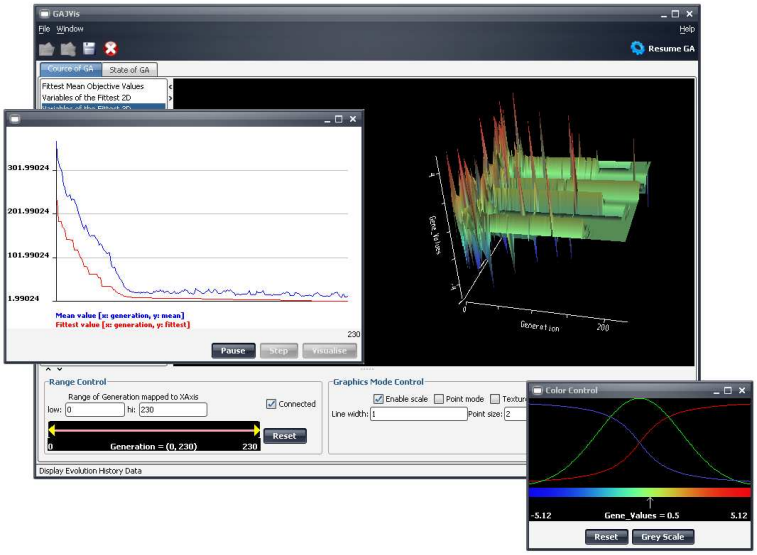
Παρακάτω θα παρουσιάσουμε τον τρόπο χρήσης των διαγραμμάτων της εφαρμογής GAJVis (εικόνα Β΄.3). Σε κάθε διάγραμμα έχουμε την δυνατότητα να εκτελέσουμε τις παρακάτω ενέργειες:

- Εξαγωγή εικόνας. Επιλέγουμε *File→Export Image* και έχουμε την δυνατότητα να επιλέξουμε τρεις τύπους εικόνας, PNG, JPEG και BMP.

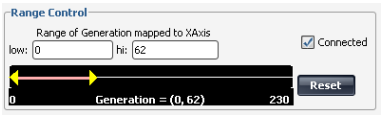


Σχήμα Β΄.2: *Παράθυρο wizard*

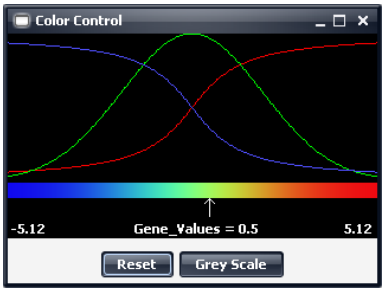
- Μεγέθυνση. Διατηρώντας το πλήκτρο Ctrl πιεσμένο, πιέζουμε το αριστερό κουμπί του ποντικιού και μετακινούμε το ποντίκι προς τα επάνω.
- Σμίκρυνση. Διατηρώντας το πλήκτρο Ctrl πιεσμένο, πιέζουμε το αριστερό κουμπί του ποντικιού και μετακινούμε το ποντίκι προς τα κάτω.
- Μετακίνηση ολόκληρου του διαγράμματος. Διατηρώντας το πλήκτρο Alt πιεσμένο, πιέζουμε το αριστερό κουμπί του ποντικιού και μετακινούμε το ποντίκι προς την κατεύθυνση που επιθυμούμε.
- Περιστροφή τρισδιάστατου διαγράμματος. Πιέζουμε το αριστερό κουμπί του ποντικιού και μετακινούμε το ποντίκι προς την κατεύθυνση που επιθυμούμε.
- Εμφάνιση τιμών στο σημείο που βρίσκεται το ποντίκι. Απλός διατηρούμε πιεσμένο το μεσαίο κουμπί του ποντικού και το μετακινούμε προς την κατεύθυνση που επιθυμούμε.
- Μπορούμε να μετακινηθούμε στον επόμενο ή τον προηγούμενο πληθυσμό με τα κουμπιά Previous και Next αντιστοίχως.
- Μπορούμε να περιοριστούμε σε ένα υποσύνολο του διαγράμματος, όπως φαίνεται στην εικόνα Β΄.4.
- Να τροποποιήσουμε τα χρώματα του διαγράμματος (εικόνα Β΄.5).



Σχήμα Β΄.3: *Η εφαρμογή GAJVis κατά την εκτέλεση ενός ΓΑ*



Σχήμα Β΄.4: *Επιλογή υποσυνόλου από το σύνολο των γενεών.*



Σχήμα Β΄.5: *Διαχείριση χρωμάτων*

Γ′

Παραδείγματα Χρήσης

Γ′.1 Παραδείγματα χρήσης της βιβλιοθήκης GAJLib

Μαζί με την πτυχιακή εργασία δίνονται και μερικά παραδείγματα χρήσης της βιβλιοθήκης. Στην εφαρμογή βρίσκονται μέσα στον κατάλογο ext, στο αρχείο gajlib_examples.jar. Τα παραδείγματα περιέχουν δοκιμαστικές συναρτήσεις, στις οποίες καλείται ο ΓΑ να εντοπίσει το ολικό βέλτιστο.

Γ′.1.1 Συναρτήσεις

Συνάρτηση Ackley

Η συνάρτηση Ackley [22] είναι ένα πρόβλημα ελαχιστοποίησης. Αρχικά η συνάρτηση είχε οριστεί μόνο για δύο διαστάσεις, αλλά αργότερα γενικεύτηκε σε N διαστάσεις [2]. Τυπικά, το πρόβλημα μπορεί να περιγραφεί ως η εύρεση ενός διανύσματος $\overrightarrow{x} = (x_1, x_2, \ldots, x_N)$, με $x_i \epsilon [-30, 30]$, το οποίο ελαχιστοποιεί την παρακάτω εξίσωση:

$$F(\vec{x}) = -20 \cdot exp(-0.2\sqrt{\frac{1}{n} \cdot \sum_{i=1}^{n} x_i^2}) - exp(\frac{1}{n} \cdot \sum_{i=1}^{n} \cos(2\pi x_i)) + 20 + exp(1) \tag{Γ′.1}$$

$$\vec{x^*} = (0, ..., 0)^T \ ; \ f(x^*) = 0 \ ; \ n = 30$$

Συνάρτηση Rastrigin

Η γενικευμένη συνάρτηση Rastrigin (συνάρτηση Γ΄.2) είναι ένα τυπικό παράδειγμα μη-γραμμικής πολυκόρυφης συνάρτησης (non-linear multimodal function). Αρχικά προτάθηκε από τον Rastrigin [23] ως μία συνάρτηση δύο διαστάσεων και αργότερα γενικεύτηκε από τον Mühlenbein et al [24]. Η συνάρτηση αυτή αποτελεί δύσκολο πρόβλημα, εξαιτίας του μεγάλου χώρου τιμών και του μεγάλου πλήθους τοπικών ελάχιστων. Η συνάρτηση έχει πολυπλοκότητα $O(n \cdot \ln(n))$, όπου n είναι η διάσταση του προβλήματος. Η επιφάνεια της συνάρτησης ορίζεται από τις εξωτερικές μεταβλητές A και ω, οι οποίες ελέγχουν το πλάτος και την συχνότητα διαμόρφωσης αντίστοίχα.

$$F(\vec{x}) = A \cdot n + \sum_{i=1}^{n} x_i^2 - A \cdot \cos(\omega \cdot x_i) \qquad (\Gamma'.2)$$
$$A = 10 \ ; \ \omega = 2 \cdot \pi \ ; \ x_i \in [-5.12, 5.12]$$

Η βέλτιστη λύση της συνάρτησης είναι το διάνυσμα $\vec{x}^* = (0, ..., 0)$ με $F(\vec{x}^*) = 0$.

Συνάρτηση Sphere

Η συνάρτηση Sphere χρησιμοποιήθηκε κατά την ανάπτυξη της θεωρίας των ΕΣ [20]. Επίσης είναι η πρώτη από τις δοκιμαστικές συναρτήσεις που χρησιμοποιήθηκαν από τον De Jong [21] για την εκτίμηση των ΓΑ.

$$F(\vec{x}) = \sum_{i=1}^{n} x_i^2 \qquad (\Gamma'.3)$$

$$\vec{x}^* = (0, ..., 0)^T \ ; \ f(x^*) = 0 \ ; \ n = 30; \ x_i \epsilon [-5.12, 5.12]$$

Γ΄.1.2 Παράδειγμα ελαχιστοποίησης συνάρτησης

Στην προηγούμενη παράγραφο παρουσιάστηκαν οι δοκιμαστικές συναρτήσεις που δίνονται ως παραδείγματα. Για να δείξουμε τον τρόπο χρήσης της βιβλιοθήκης, παρακάτω θα ασχοληθούμε με την ελαχιστοποίηση της συνάρτησης Rastrigin με $n = 20$. Το πρώτο βήμα για την επίλυση ενός προβλήματος με ΓΑ, είναι η επιλογή της αναπαράστασης. Η συγκεκριμένη συνάρτηση μπορεί να επιλυθεί είτε με δυαδική αναπαράσταση είτε με αναπαράσταση αριθμών κινητής υποδιαστολής. Το παράδειγμα αυτό θα το επιλύσουμε και με τις δύο αναπαραστάσεις.

Α΄ τρόπος Αναπαράσταση αριθμών κινητής υποδιαστολής

Η αναπαράσταση είναι κινητής υποδιαστολής, με κάθε χρωμόσωμα να έχει 20 γονίδια τα οποία δέχονται τιμές από -5.12 έως 5.12. Στη συνέχεια χρειάζεται να γραφτεί κώδικας για την συνάρτηση ποιότητας. Ο κώδικας είναι ο παρακάτω:

```
package examples.functions;

import gr.teithe.it.ga.IChromosome;
import gr.teithe.it.ga.eval.AbstractMinimizationEvaluator;
import gr.teithe.it.ga.representation.DoubleRepresentation;
import gr.teithe.it.ga.representation.genes.DoubleGene;
import gr.teithe.it.ga.representation.genes.IGene;
import gr.teithe.it.util.annotations.Require;

@Require(value = DoubleRepresentation.class)
public final class RastriginFunction extends AbstractMinimizationEvaluator {

    public void evaluate(IChromosome c) {

        IGene[] genes = c.getGenes();
        double fitness = 0.0;
        double curr = 0.0;

        for (int i = 0; i < genes.length; i++) {
            curr = ((DoubleGene) genes[i]).getValue();
            fitness += (Math.pow(curr, 2)
                - 10.0 * Math.cos(2.0 * Math.PI * curr)) + 10.0;
        }
        c.setFitnessValue(fitness);
    }
}
```

Για να δηλώσουμε ότι είναι πρόβλημα ελαχιστοποίησης, απλώς η κλάση θα πρέπει να επεκτείνει την αφηρημένη (abstract) κλάση *AbstractMinimizationEvaluator*. Χρειάζεται μόνο μία μέθοδος να υλοποιηθεί από τον προγραμματιστή (*public void evaluate(IChromosome c)*), από την οποία δίνεται ένα χρωμόσωμα για εκτίμηση της ποιότητάς του. Από το χρωμόσωμα έχουμε πρόσβαση στα γονίδιά του (*IGene[] c.getGenes()*), από τα οποία διαβάζουμε τις τιμές τους και θέτουμε την τιμή ποιότητας του χρωμοσώματος. (*c.setFitnessValue(double value)*). Τέλος εισάγοντας το *Annotation @Require(value = Class)*, δηλώνουμε ότι η συγκεκριμένη τάξη απαιτεί αναπαράσταση αριθμών κινητής υποδιαστολής.

Το επόμενο βήμα είναι η σύνταξη του αρχείου ρυθμίσεων, σύμφωνα με το οποίο επιλέγουμε όλα τα στοιχεία του ΓΑ. Έστω ότι θέλουμε τις εξής ρυθμίσεις:

- Μέγεθος πληθυσμού ίσο με 200.
- Πιθανότητα ανασυνδυασμού 0.9.
- Πιθανότητα μετάλλαξης 0.4.
- Να χρησιμοποιηθεί ελιτισμός.

- Να χρησιμοποιηθεί για ανασυνδυασμό ο τελεστής BLX-a με τιμή $\alpha = 0.5$.
- Για μετάλλαξη να χρησιμοποιηθεί ο τελεστής Creep Mutation.
- Για επιλογή η μέθοδος Tournament Selection με αριθμό υποψήφιων χρωμοσωμάτων 10.
- Η αναπαράσταση να είναι αριθμών κινητής υποδιαστολή με αριθμό γονιδίων 20 σε κάθε χρωμόσωμα. Κάθε χρωμόσωμα δέχεται τιμές από -5.12 έως και 5.12.
- Η συνθήκη τερματισμού να είναι 2000 γενιές το πολύ ή χρόνος εκτέλεσης 5 δευτερόλεπτα.
- Τα αποτελέσματα να τυπώνονται στην οθόνη.

Οπότε σύμφωνα με τα παραπάνω το αρχείο ρυθμίσεων θα έχει την παρακάτω μορφή:

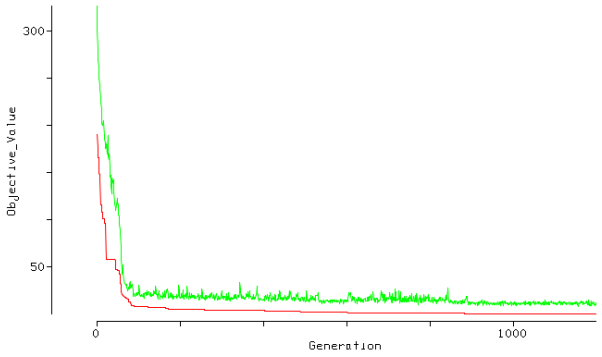
```
<?xml version="1.0" encoding="UTF-8"?>
<gr.teithe.it.ga.configuration.Configurator
    xmlns:j="http://javolution.org" xmlns:t="http://www.it.teithe.gr"
    xmlns:g="java:gr.teithe.it.ga">
  <Title value="Rastrigin Double"/>
  <PopulationSize value="200"/>
  <RecombinationProbability value="0.9"/>
  <MutationProbability value="0.4"/>
  <UseElitism value="true"/>
  <Recombinator j:class="gr.teithe.it.ga.operators.real64.BLXaRecombinator"
     Alpha="0.5"/>
  <Mutator j:class="gr.teithe.it.ga.operators.real64.CreepMutator"/>
  <NaturalSelector j:class="gr.teithe.it.ga.selectors.DefaultNaturalSelector"/>
  <Selector j:class="gr.teithe.it.ga.selectors.TournamentSelector"
     Candidates="10"/>
  <Evaluator j:class="examples.functions.RastriginFunction"/>
  <Initializer j:class="gr.teithe.it.ga.RandomInitializer">
    <Random j:class="gr.teithe.it.util.random.DefaultRandomGenerator"/>
  </Initializer>
  <Representation j:class="gr.teithe.it.ga.representation.DoubleRepresentation"
      Length="20">
    <Domain j:class="gr.teithe.it.ga.Domain" Type="java.lang.Double"
      LowerBound="-5.12" UpperBound="5.12"/>
  </Representation>
  <TerminationCondition j:class="gr.teithe.it.ga.termination.OrCondition">
    <TerminationCondition j:class="gr.teithe.it.ga.termination.MaxIterations"
       Iterations="2000"/>
    <TerminationCondition j:class="gr.teithe.it.ga.termination.MaxExecutionTime"
      Time="5"/>
  </TerminationCondition>
  <Writer j:class="gr.teithe.it.ga.audit.ConsoleWriter"/>
</gr.teithe.it.ga.configuration.Configurator>
```

Παρατηρώντας το αρχείο, είναι φανερή η ιεραρχική δομή του και η εύκολη επεκτασιμότητά του. Οι γενικές ρυθμίσεις του ΓΑ (π.χ. Title, PopulationSize

κ.α) βρίσκονται στην αρχή και μετά ακολουθούν οι δηλώσεις και οι ρυθμίσεις των στοιχείων του ΓΑ.

Ένα στοιχείο του ΓΑ διαθέτει ένα όνομα, όπως Recombinator, ολόκληρο το classpath της τάξης *(j:class="gr.teithe.it.ga.operators.real64.BLXaRecombinator")* και προαιρετικά τις παραμέτρους του (πχ *Alpha="0.5"*).

Είναι δυνατό να οριστεί για το κάθε γονίδιο διαφορετικό πεδίο τιμών, στην περίπτωση αυτή θα πρέπει να δηλωθούν ξεχωριστά όλα τα πεδία για όλα τα γονίδια. Όταν όμως το πεδίο τιμών είναι το ίδιο σε όλα τα γονίδια, αρκεί μία μόνο δήλωση. Τέλος ο ΓΑ μπορεί να εκτελεστεί είτε από το γραφικό περιβάλλον, είτε μέσω της εντολής evolve (evolve.bat σε περιβάλλον Windows).



Σχήμα Γ΄.1: *Διαγραμμα μέσης και καλήτερης τιμής, συνάρτηση Rastrigin*

Β΄ τρόπος Δυαδική αναπαράσταση

Για την επίλυση του προβλήματος με δυαδική αναπαράσταση η προσέγγιση είναι παρόμοια. Παρακάτω δίνεται ο πηγαίος κώδικας της συνάρτησης Rastrigin για την δυαδική αναπαράσταση:

```
package examples.functions;

import gr.teithe.it.ga.IChromosome;
import gr.teithe.it.ga.eval.AbstractMinimizationEvaluator;
import gr.teithe.it.ga.representation.BinaryRepresentation;
import gr.teithe.it.ga.representation.genes.BinaryGene;
import gr.teithe.it.ga.representation.genes.IGene;
import gr.teithe.it.util.BinaryUtils;

import gr.teithe.it.util.annotations.Require;

@Require(value = BinaryRepresentation.class)
public final class RastriginFunctionBinary extends AbstractMinimizationEvaluator {
```

```
    public void evaluate(IChromosome c) {

        IGene[] genes = c.getGenes();
        double fitness = 0.0;
        double curr = 0.0;

        for (int i = 0; i < genes.length; i++) {
            curr = BinaryUtils.parseDouble((BinaryGene) genes[i]);
            fitness += (Math.pow(curr, 2) - 10.0 * Math.cos(2.0 * Math.PI * curr)) + 10.0;
        }
        c.setFitnessValue(fitness);
    }
}
```

Εύκολα διαπιστώνεται πως οι διαφορές είναι η τιμή στο *Annotation @Require* και η χρήση της βοηθητικής τάξης *BinaryUtils* με την οποία μπορούμε να αποκωδικοποιούμε την δυαδική τιμή κάθε γονιδίου. Επιπλέον, θέτοντας για τελεστή ανασυνδυασμού τον ανασυνδυασμό ενός σημείου, για μετάλλαξη την *BitFlip* μετάλλαξη και την ακρίβεια των αριθμών σε 8 δεκαδικά ψηφία, το αρχείο ρυθμίσεων διατυπώνεται ως εξής:

```
<?xml version="1.0" encoding="UTF-8"?>
<gr.teithe.it.ga.configuration.Configurator
    xmlns:j="http://javolution.org" xmlns:t="http://www.it.teithe.gr"
    xmlns:g="java:gr.teithe.it.ga">
  <Title value="Rastrigin Binary"/>
  <PopulationSize value="200"/>
  <RecombinationProbability value="0.9"/>
  <MutationProbability value="0.1"/>
  <UseElitism value="true"/>
  <Recombinator
    j:class="gr.teithe.it.ga.operators.binary.BitSinglePointRecombinator"/>
  <Mutator j:class="gr.teithe.it.ga.operators.binary.BitFlipMutator"/>
  <NaturalSelector j:class="gr.teithe.it.ga.selectors.DefaultNaturalSelector"/>
  <Selector j:class="gr.teithe.it.ga.selectors.TournamentSelector" Candidates="10"/>
  <Evaluator j:class="examples.functions.RastriginFunctionBinary"/>
  <Initializer j:class="gr.teithe.it.ga.RandomInitializer">
    <Random j:class="gr.teithe.it.util.random.DefaultRandomGenerator"/>
  </Initializer>
  <Representation j:class="gr.teithe.it.ga.representation.BinaryRepresentation"
          Length="20">
     <Domain j:class="gr.teithe.it.ga.Domain" Type="java.lang.Double"
          LowerBound="-5.12" UpperBound="5.12" Precision="8"/>
  </Representation>
  <TerminationCondition j:class="gr.teithe.it.ga.termination.OrCondition">
    <TerminationCondition j:class="gr.teithe.it.ga.termination.MaxIterations"
      Iterations="2000"/>
    <TerminationCondition j:class="gr.teithe.it.ga.termination.MaxExecutionTime"
      Time="5"/>
  </TerminationCondition>
  <Writer j:class="gr.teithe.it.ga.audit.ConsoleWriter"/>
</gr.teithe.it.ga.configuration.Configurator>
```

Γ΄.1.3 Το πρόβλημα με τις σφηκοφωλιές

Παρακάτω θα επιλύσουμε ένα πρόβλημα το οποίο προέρχεται από το μάθημα των Ευφυών Συστημάτων, που διδάσκεται στο έβδομο εξάμηνο του τμήματος Πληροφορικής.

Εκφώνηση

Μόλις αγοράσατε ένα σπίτι και ανακαλύπτετε ότι η σοφίτα του είναι γεμάτη από σφηκοφωλιές. Πριν μετακομίσετε στο νέο σας σπίτι αποφασίζετε να εξοντώσετε τις σφήκες. Επισκέπτεστε το κατάστημα της περιοχής σας το οποίο διαθέτει εντομοκτόνα αλλά βρίσκεται μόνο τρία (3) δοχεία τύπου «εντομοβόμβασ»τα οποία έχουν συγκεκριμένη ακτίνα δράσης και πρέπει να τοποθετηθούν πολύ κοντά στη φωλιά για να εξοντώσουν τις σφήκες που βρίσκονται μέσα. Δυστυχώς τα 3 δοχεία δεν είναι αρκετά να εξοντώσουν όλες τις σφήκες της σοφίτας. Ευτυχώς η τύχη σας βοηθάει και βρίσκετε:

1. ένα χάρτη που άφησε ο προηγούμενος ιδιοκτήτης και ο οποίος περιγράφει την θέση που βρίσκονται οι φωλιές όπως επίσης και τον αριθμό σφηκών που διαθέτει κάθε φωλιά (χρησιμοποιώντας ένα πίνακα 100×100, Πίνακας 1).

Αριθμός φωλιάς	Αριθμός Σφηκών	Άξονας Χ	Άξονας Υ
1	100	25	65
2	200	23	8
3	327	7	13
4	440	95	53
5	450	3	3
6	639	54	56
7	650	67	78
8	678	32	4
9	750	24	76
10	801	66	89
11	945	84	4
12	967	34	23

2. ένα τύπο πάνω στο δοχείο «εντομο-βόμβασ»ο οποίος δίνει την σχέση απόστασης από την φωλιά και του ποσοστού των σφηκών οι οποίες εξοντώνονται (Εξίς. Γ΄.4).

$$K = n * \frac{d_{max}}{20 * d + 0.00001} \qquad (\Gamma'.4)$$

όπου:
K: Πλήθος σφηκών που θα σκοτωθούν σε μία φωλιά
n: Πλήθος υπαρχόντων σφηκών σε αυτή τη φωλιά
d: Απόσταση βόμβας από αυτή τη φωλιά.
d_{max}: Η μέγιστη απόσταση μεταξύ δύο φωλιών (141.42 για τον χάρτη του Πίνακα 1, με διαστάσεις 100×100)
Η απόσταση μεταξύ δύο θέσεων του χάρτη υπολογίζεται από την εξίσωση Γ΄.5:

$$d = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2} \qquad (\Gamma'.5)$$

Στόχος είναι να βρεθεί η καλύτερη δυνατή τοποθέτηση των δοχείων έτσι ώστε να εξοντωθεί ο μεγαλύτερος αριθμός σφηκών.

Υποδείξεις:

1. Για να βρείτε πόσες συνολικά σφήκες σκότωσε μία βόμβα θα πρέπει να υπολογίσετε πόσες σκότωσε σε όλες τις φωλιές. Για τον υπολογισμό του συνόλου των σφηκών που σκοτώνει μία 'εντομο-βόμβα' θα χρησιμοποιήσετε την εξίσωση Γ΄.6.

$$T = \sum_{i=1}^{n} K_i \qquad (\Gamma'.6)$$

όπου:
K: Πλήθος σφηκών που θα σκοτωθούν σε μία φωλιά
n: Πλήθος φωλιών

2. Και οι τρεις «εντομο-βόμβεσ»δεν μπορούν να σκοτώσουν σε μία φωλιά περισσότερες σφήκες από όσες συνολικά υπάρχουν σε αυτή. Εάν δηλαδή μία βόμβα έχει σκοτώσει όλες τις σφήκες κάποιας φωλιάς, τότε αφού δεν υπάρχουν άλλες σφήκες σε αυτή τη φωλιά, οι άλλες δύο δεν θα σκοτώσουν ούτε μία σε αυτή τη φωλιά.

3. Η λύση που θα δώσετε να λειτουργεί για οποιαδήποτε θέση των φωλιών. Μην την περιορίζετε μόνο στις θέσεις του παραπάνω χάρτη του Πίνακα 1, αλλά να μπορεί να λειτουργεί και για άλλους πίνακες.

Λύση

Το συγκεκριμένο πρόβλημα μπορεί να λυθεί είτε με δυαδική αναπαράσταση, είτε με αναπαράσταση ακεραίων αριθμών. Παρακάτω θα παρουσιάσουμε μόνο την λύση με ακέραια αναπαράσταση, καθώς η λύση με την δυαδική αναπαράσταση είναι παρόμοια.

Για την λύση χρειάστηκαν να κατασκευαστούν τέσσερις κλάσεις, η *WaspBombsIntEvaluator* που είναι η συνάρτηση ποιότητας, η εσωτερική (inner) κλάση *WaspBombsIntEvaluator.GUIConf* που είναι η γραφική ρύθμιση της συνάρτησης ποιότητας, η *WaspsNestLoader* που είναι υπεύθυνη για την φόρτωση του αρχείου με τις συντεταγμένες και η *WaspsNest*, είναι μία τάξη σχεδιασμένη με τα πρότυπα πραγματικού χρόνου της βιβλιοθήκης Javolution και περιέχει τις συντεταγμένες της φωλιάς και τον αριθμό των σφηκών. Η αναπαράσταση του χρωμοσώματος είναι έξη (6) γονίδια που περιέχουν τις συντεταγμένες (ακέραιες τιμές) τοποθέτησης των βομβών. Παρακάτω δίνεται ο κώδικας της συνάρτησης ποιότητας:

```
package examples.wasps;

import gr.teithe.it.util.annotations.*;
import gr.teithe.it.ga.representation.*;
import gr.teithe.it.ga.eval.AbstractMinimizationEvaluator;
import gr.teithe.it.ga.IChromosome;
import gr.teithe.it.ga.configuration.IGUIConfigurator;
import javolution.util.FastList;
import javolution.realtime.LocalContext;
import javolution.xml.*;
import javax.swing.*;
import java.io.*;
import java.awt.event.*;

@Require( value = IntegerRepresentation.class)
@GUIConfiguration(value = WaspBombsIntEvaluator.GUIConf.class)
public final class WaspBombsIntEvaluator extends AbstractMinimizationEvaluator {

    private static final String ATTR_DATA_FILE = "DataFile";

    public static final XmlFormat<WaspBombsIntEvaluator> XML =
        new XmlFormat<WaspBombsIntEvaluator>(WaspBombsIntEvaluator.class) {
        public void format(WaspBombsIntEvaluator waspBombsIntEvaluator,
            XmlElement xmlElement) {
            xmlElement.setAttribute(ATTR_DATA_FILE,waspBombsIntEvaluator.file.getAbsolutePath());
        }

        public WaspBombsIntEvaluator parse(XmlElement xmlElement) {
            return new WaspBombsIntEvaluator(new File(
              xmlElement.getAttribute(ATTR_DATA_FILE, "nests.dat")));
        }
    };

    private FastList<WaspsNest> list=null;
    private WaspNestsLoader loader=null;
    private File file = null;
    private static final double dmax = 141.42;

    public WaspBombsIntEvaluator(File file){
        super();

        if(file==null)
            this.file = new File("nests.dat");
        else
            this.file = file;
```

```
        try {
            loader = new WaspNestsLoader(file);
        } catch (FileNotFoundException e) {
            e.printStackTrace();
        }
        try {
            list = loader.load();
        } catch (IOException e) {
            e.printStackTrace();
        }
    }

    public void evaluate(IChromosome c) {
        IGene[] genes = c.getGenes();
        try{
            LocalContext.enter();
            FastList<WaspsNest> currList= FastList.newInstance();
            for (FastList.Node<WaspsNest> n = list.head(), end = list.tail();
               (n = n.getNext()) != end;) currList.addLast(n.getValue().copy());

            double fitness = 0.0;
            double k;

            for(int i=0; i<genes.length; i+=2){
                k = 0.0;
                int bombX = ((IntegerGene) genes[i]).getValue();
                int bombY = ((IntegerGene) genes[i+1]).getValue();
                for(WaspsNest node: currList){
                    k = node.numberOfWasps*(WaspBombsIntEvaluator.dmax /
                            ((20*distance(bombX, bombY, node.x, node.y))
                                    +0.00001));
                    if(node.numberOfWasps!=0){
                        if(k>node.numberOfWasps) node.numberOfWasps=0;
                        else node.numberOfWasps-=k;
                    }
                }
            }

            for(WaspsNest node: currList){
                fitness+=node.numberOfWasps;
            }
            c.setFitnessValue(fitness);
        }finally{
            LocalContext.exit();
        }
    }

    private final double distance(int x1, int y1, int x2, int y2){
        return Math.sqrt(Math.pow(x1-x2,2)+Math.pow(y1-y2, 2));
    }

    public final static class GUIConf extends JPanel implements
        IGUIConfigurator<WaspBombsIntEvaluator> {

        private final JButton btnOpenFile = new JButton("Open File");
        private final JFileChooser fileChooser = new JFileChooser();
        private File file = null;

        public GUIConf(){
            fileChooser.setDialogTitle("Open wasp nests file");
            btnOpenFile.addActionListener(new ActionListener() {
                public void actionPerformed(ActionEvent e) {
```

```
                    if(fileChooser.showOpenDialog(null) ==
                           JFileChooser.APPROVE_OPTION){
                        file = fileChooser.getSelectedFile();
                    }
                }
            });
            add(btnOpenFile);
        }

        public JComponent getJComponent() {
            return this;
        }

        public WaspBombsIntEvaluator getConfiguredType() {
            return new WaspBombsIntEvaluator(file);
        }
}
```

Παρατηρώντας τον κώδικα διαπιστώνουμε ότι είναι πρόβλημα ελαχιστοποίησης του πλήθους των σφηκών, για τον λόγο αυτό η τάξη της συνάρτησης ποιότητας επεκτείνει την τάξη *AbstractMinimizationEvaluator*. Σε κάθε κλήση της μεθόδου *evaluate(IChromosome c)* δίνεται μία προσωρινή λίστα με τις φωλιές (η λίστα επαναχρησιμοποιείται με βάση τους μηχανισμούς του Javolution) η οποία χρησιμοποιείται για την εκτίμηση της ποιότητας του χρωμοσώματος.

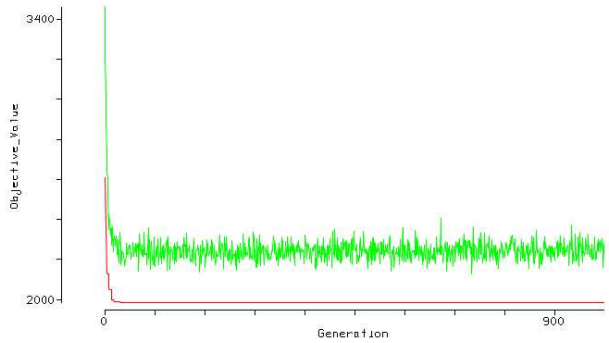
Το αντικείμενο *XmlFormat* υλοποιείται απευθείας κατά την δήλωσή του, έτσι ώστε από τις μεθόδους format και parse να ορίζει ο προγραμματιστής τον τρόπο με τον οποίο θα σώζεται και θα φορτώνεται αντίστοιχα, η τάξη *WaspBombsIntEvaluator* στο αρχείο ρυθμίσεων. Με τον τρόπο αυτό ο προγραμματιστής δηλώνει και χειρίζεται τις δικές του παραμέτρους.

Η εσωτερική τάξη *GUIConf* υλοποιεί το inteface *IGUIConfigurator* και ο σκοπός της είναι, όταν το πρόβλημα επιλεχθεί από τον *wizard* του γραφικού περιβάλλοντος να εμφανιστεί ένα παράθυρο από το οποίο ο χρήστης επιλέγει το αρχείο με τις φωλιές των σφηκών. Τέλος για να γίνει αντιληπτή αυτή η τάξη από τον wizard πρέπει να οριστεί το *Annotation @GUIConfiguration*.

Συνεπώς, με βάση τα παραπάνω το αρχείο ρυθμίσεων θα είναι το παρακάτω:

```
<?xml version="1.0" encoding="UTF-8"?>
<gr.teithe.it.ga.configuration.Configurator xmlns:j="http://javolution.org"
  xmlns:t="http://www.it.teithe.gr" xmlns:g="java:gr.teithe.it.ga">
  <Title value="Wasp Bombs Problem"/>
  <PopulationSize value="100"/>
  <RecombinationProbability value="0.9"/>
  <MutationProbability value="0.4"/>
  <UseElitism value="false"/>
  <Recombinator j:class="gr.teithe.it.ga.operators.SinglePointRecombinator"/>
  <Mutator j:class="gr.teithe.it.ga.operators.RandomMutator"/>
  <NaturalSelector j:class="gr.teithe.it.ga.selectors.DefaultNaturalSelector"/>
  <Selector j:class="gr.teithe.it.ga.selectors.TournamentSelector" Candidates="7"/>
  <Evaluator j:class="examples.wasps.WaspBombsIntEvaluator" DataFile="nests.dat"/>
  <Initializer j:class="gr.teithe.it.ga.RandomInitializer">
    <Random j:class="gr.teithe.it.util.random.DefaultRandomGenerator"/>
  </Initializer>
```

```
  <Representation j:class="gr.teithe.it.ga.representation.IntegerRepresentation"
     Length="6">
    <Domain j:class="gr.teithe.it.ga.Domain" Type="java.lang.Integer" LowerBound="1"
     UpperBound="100"/>
  </Representation>
  <TerminationCondition j:class="gr.teithe.it.ga.termination.MaxIterations"
     Iterations="1000"/>
  <Writer j:class="gr.teithe.it.ga.audit.ConsoleWriter"/>
</gr.teithe.it.ga.configuration.Configurator>
```



Σχήμα Γ΄.2: *Διάγραμμα μέσης και καλύτερης τιμής, πρόβλημα με τις σφηκοφωλιές*

Βιβλιογραφία

[1] Goldberg D. E. Genetic Algorithms in Search, Optimization & Machine Learning. Addison-Wesley Publishing Company, 1989.

[2] Thomas Bäck. Evolutionary Algorithms in Theory and Practice. Oxford University Press, 1996

[3] A.E. Eiben, J.E. Smith. Introduction to Evolutionary Computing (Natural Computing Series). Springer, 2003.

[4] William M. Spears. Evolutionary Algorithms, The Role of Mutation and Recombinarion. Springer, 2000.

[5] Zbigniew Michlewicz. Genetic Algorithms + Data Structures = Evolution Programs 3ed. Springer, 1996.

[6] David A. Coley. An Introduction to Genetic Algorithms for Scientists and Engineers. World Scientific, 1997.

[7] Randy L. Haupt, Sue Ellen Haupt. Practical Genetic Algorithms 2ed. Wiley-Interscience, 2004.

[8] Melanie Mitchell. An Introduction to Genetic Algorithms. MIT Press, 1998.

[9] Thomas Bäck, David B. Fogel, Zbigniew Michlewicz. Evolutionary Computation 1, Basic Algorithms and Operators. IOP Publishing Ltd, 2000.

[10] Thomas Bäck, David B. Fogel, Zbigniew Michlewicz. Evolutionary Computation 2, Advanced Algorithms and Operators. IOP Publishing Ltd, 2000.

[11] Chambers D. Lance. Practical Handbook of Genetic Algorithms, New Frontiers, Volume II. CRC Press Inc, 1995.

[12] Chambers D. Lance. Practical Handbook of Genetic Algorithms, Complex Coding Systems, Volume III. CRC Press LLC, 1999.

[13] Andreas Kerren, Thomas Egger. EAVis: A Visualization Toll for Evolutionary Algorithms. Proceedings of the IEEE Symposium on Visual Languages and Human-Centric Computing, VL/HCC 2005, pages 299-301, Dallas, TX, USA, 2005. IEEE Computer Society Press, 2005.

[14] Hartmut Pohlheim. Visualization of Evolutionary Algorithms - Set of Standard Technigues and Multidimensional Visualization. In Banzhaf, W.(ed): GECCO'99 - Proceedings of the Genetic and Evolutionary Computation Conference, San Francisco, CA: Morgan Kaufmann, pp. 533-540, 1999.

[15] Darrell Whitley. An overview of Evolutionary Algorithms: Practical issues and common pitfalls. Elsevier Science B.V, 2001.

[16] Darrell Whitley. A Genetic Algorithm Tutorial. Computer Science Deparment, Colorado State University.

[17] Kai-min Kevin Chang. A Real-Coded Genetic Algorithm Approach to Data Segmentation Problem. University of Waterloo, Faculty of Mathematics, 2002.

[18] F. Herrera, M. Lozano, J.L. Verdegay. Tackling Real-Coded Genetic Algorithms: Operators and Tools for Behavioural Analysis. 1998.

[19] Π. Αδαμίδης. Εξελικτικοί Αλγόριθμοι, Εισαγωγή. Α.Τ.Ε.Ι. Θεσσαλονίκης, Τμήμα Πληροφορικής, 1999.

[20] I. Rechenberg. EvolutionsstrategieOptimierum technischer Systeme nach Prinzipien der biologischen Evolution. PhD thesis, StuttgartBad Cannstatt: Frommann-Holzboog, 1973.

[21] K. D. De Jong.An analysis of the behavior of a class of genetic adaptive systems. PhD thesis, Departament of Computer and Communication Sciences, University of Michigan, Ann Arbor, 1975.

[22] D. H. Ackley. "A connectionist machine for genetic hillclimbing". Boston: Kluwer Academic Publishers, 1987.

[23] A. Törn and A. Zilinskas. "Global Optimization". Lecture Notes in Computer Science, No 350,Springer-Verlag,Berlin,1989.

[24] H. Mühlenbein, D. Schomisch and J. Born. "The Parallel Genetic Algorithm as Function Optimizer". Parallel Computing, 17, pages 619-632,1991.

[25] David R. Wilkins. Getting Started with LaTeX 2nd Edition. 1995.

[26] Apostolos Syropoulos, Antonis Tsolomitis, Nick Sofroniou. Digital Typografy Using LaTeX. Springer, 2003.

[27] Jack Shirazi. Java Performance Tuning, 2nd Edition. O' Reilly, 2003.

[28] David Flanagan, Brett McLaughlin. Java 1.5 Tiger: A Developer's Notebook. O'Reilly 2004.

[29] Ron Hitchens. Developing High Performance Applications, Java NIO. O'Reilly, 2002.

[30] Steve Holzner. Ant: The Definitive Guide, 2nd Edition. O'-Reilly, 2005.

[31] Matthew Robinson, Pavel Vorobiev. Swing 2nd Edition, Foreword by James Gosling, "Father" of Java. Manning, 2003.

[32] Ira R. Forman, Nate Forman. Java Reflection in Action. Manning, 2005.

[33] Jean-Marie Dautelle. Javolution, the Java solution for Real-Time and Embended Systems.
`http://www.javolution.org/`

[34] Jean-Marie Dautelle. Collections Classes for Real-Time and High-Performance Applications. 2005.

[35] Jean-Marie Dautelle. Validating Java for Safety-Critical Applications. AIAA Space 2005 Conference.

[36] Jean-Marie Dautelle. Turbo-Charging Java for Real-Time Applications. July 2004 issue of Java Developer Journal.

[37] Gregory Bollella. The Real-Time Specification for Java. Addison Wesley Longman, 2000

[38] Ugo Taddei. The VisAD Tutorial. 2003.
`http://www.ssec.wisc.edu/billh/tutorial/index.html`

[39] The VisAD Java Component Library, Developers Guide. 2000. `http://www.ssec.wisc.edu/~billh/guide.html`