ΠΑΝΕΠΙΣΤΗΜΙΟ ΠΑΤΡΩΝ
ΤΜΗΜΑ ΜΗΧΑΝΟΛΟΓΩΝ ΚΑΙ ΑΕΡΟΝΑΥΠΗΓΩΝ ΜΗΧΑΝΙΚΩΝ
ΚΑΤΑΣΚΕΥΑΣΤΙΚΟΣ ΤΟΜΕΑΣ
ΕΡΓΑΣΤΗΡΙΟ ΥΠΟΛΟΓΙΣΜΟΥ ΚΑΙ ΣΧΕΔΙΑΣΕΩΣ ΣΤΟΙΧΕΙΩΝ ΜΗΧΑΝΩΝ

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Ανάπτυξη Περιβάλλοντος για την Υλοποίηση Σχεδιαστικής Βελτιστοποίησης με Χρήση Parser και Γενετικών Αλγορίθμων

Άγγελος **Μαυρογιάννης**
6387

Επιβλέπων : Καθηγητής Αργύρης Δέντσορας

Διπλωματική εργασία υποβληθείσα στο Τμήμα Μηχανολόγων και Αεροναυπηγών του Πανεπιστημίου Πατρών

ΠΑΤΡΑ, Ιούλιος, 2017

ΠΑΝΕΠΙΣΤΗΜΙΟ ΠΑΤΡΩΝ
ΤΜΗΜΑ ΜΗΧΑΝΟΛΟΓΩΝ ΚΑΙ ΑΕΡΟΝΑΥΠΗΓΩΝ ΜΗΧΑΝΙΚΩΝ
[ΤΟΜΕΑΣ]
[ΟΝΟΜΑ ΕΡΓΑΣΤΗΡΙΟΥ]

Η παρούσα διπλωματική εργασία παρουσιάστηκε

από τον

Άγγελο Μαυρογιάννη

6387

την 7/7/2017

Η έγκριση της διπλωματικής εργασίας δεν υποδηλοί την αποδοχή των γνωμών του συγγραφέα. Κατά τη συγγραφή τηρήθηκαν οι αρχές της ακαδημαϊκής δεοντολογίας.

Ανάπτυξη Περιβάλλοντος για την Υλοποίηση Σχεδιαστικής Βελτιστοποίησης με Χρήση Parser και Γενετικών Αλγορίθμων

Άγγελος Μαυρογιάννης

ΠΕΡΙΛΗΨΗ

Το αντικείμενο της παρούσης εργασίας είναι η έρευνα, ο σχεδιασμός και η ανάπτυξη ενός προγράμματος για την εισαγωγή, ανάλυση – μέσω κατάλληλου parser – και βελτιστοποίηση υπολογιστικών σχεδιαστικών σχέσεων με τη βοήθεια γενετικών αλγορίθμων. Η χρήση του parser γίνεται κατά την εισαγωγή των μαθηματικών σχέσεων και επιτρέπει την εισαγωγή υπολογιστικών σχέσεων υπό μορφή κειμένου και την αναγνώριση μεταβλητών και συμβόλων. Έτσι παρέχεται η δυνατότητα σχεδιασμού, δόμησης και δημιουργίας ενός παραμετρικού υπολογιστικού περιβάλλοντος, το οποίο θα μπορεί να λαμβάνει ως είσοδο οποιαδήποτε μαθηματική σχέση και να πραγματοποιεί βελτιστοποίηση σε πραγματικό χρόνο. Όσον αφορά στη μέθοδο βελτιστοποίησης, επιλέχθηκε η χρήση των γενετικών αλγορίθμων. Οι γενετικοί αλγόριθμοι χρησιμοποιούνται σε εφαρμογές ανάλυσης και βελτιστοποίησης και υπερβαίνουν πολλά εμπόδια και περιορισμούς που υπάρχουν στις συμβατικές μεθόδους βελτιστοποίησης. Ως παραδείγματα εφαρμογής της προτεινόμενης μεθοδολογίας επιλέχθηκαν το απλό πρόβλημα βελτιστοποίησης της ακαμψίας μιας τυπικής πρόβολης δοκού και το πρόβλημα ελαχιστοποίησης των δυνάμεων που ασκεί ένα ρομποτικό χέρι σε ένα αντικείμενο που πιάνει.

Λέξεις κλειδιά
Υπολογιστικό Περιβάλλον, Βελτιστοποίηση, Parser, Γενετικοί Αλγόριθμοι

Environment Development for Implementing Design Optimization Using Parsers and Genetic Algorithms

Angelos Mavrogiannis

ABSTRACT

The present study examines the design and creation of a program, capable of receiving computational design equations as input, analyzing them using a suitable parser and optimizing them using Genetic Algorithms. The parser is used during the input of the equations. It allows the input of equations as text and, at a next step, the recognition of the variables and the operational symbols in the text body. Hence, it is possible to design, assemble and create an abstract computational environment, which can receive any mathematical equation as input and optimize it in real time. As far as the optimization method is concerned, genetic algorithms will be the technique to be used. Genetic algorithms are used widely on analysis and optimization problems, since they overcome common obstacles and limitations posed by other conventional optimization techniques. To demonstrate the aforementioned methodology, the simple problem of the stiffness of a cantilever beam and the problem of the minimization of the forces applied onto an object grasped by a robot hand will be the selected case studies.

Keywords
Computational Environment, Optimization, Parser, Genetic Algorithms

ΚΑΤΑΛΟΓΟΣ ΠΙΝΑΚΩΝ

ΚΑΤΑΛΟΓΟΣ ΣΧΗΜΑΤΩΝ ΚΑΙ ΕΙΚΟΝΩΝ

ΣΥΜΒΟΛΙΣΜΟΙ

Σύμβολο	Μονάδα Μέτρησης	Σημασία
k	N/m	Σταθερά ελατηρίου
F	N	Εφαρμοζόμενη δύναμη
δ	m	Μεταβολή μήκους
σ	Pa	Τάση
ε	-	Παραμόρφωση
E	Pa	Μέτρο ελαστικότητας
A	m^2	Διατομή
l	m	Μήκος
q1	m	Διάνυσμα θέσης δαχτύλου #1
q2	m	Διάνυσμα θέσης δαχτύλου #2
c	-	Κέντρο μάζας αντικειμένου
f1	N	Δύναμη από δάχτυλο #1 στο αντικείμενο
f2	N	Δύναμη από δάχτυλο #2 στο αντικείμενο
f1x	N	Οριζόντια δύναμη από δάχτυλο #1 στο αντικείμενο
f2x	N	Οριζόντια δύναμη από δάχτυλο #2 στο αντικείμενο
f1y	N	Κατακόρυφη δύναμη από δάχτυλο #1 στο αντικείμενο
f2y	N	Κατακόρυφη δύναμη από δάχτυλο #2 στο αντικείμενο
W	N	Βάρος αντικειμένου
m	kg	Μάζα Αντικειμένου
g	m/s^2	Επιτάχυνση της βαρύτητας
μ	-	Συντελεστής τριβής
rc	m	Διάνυσμα θέσης κέντρου μάζας αντικειμένου
a	m	Οριζόντια διάσταση του αντικειμένου
b	m	Κατακόρυφη διάσταση του αντικειμένου
y1	m	Το μήκος στην κατακόρυφη πλευρά όπου τοποθετείται το δάχτυλο #1
y1	m	Το μήκος στην κατακόρυφη πλευρά όπου τοποθετείται το δάχτυλο #2

ΠΕΡΙΕΧΟΜΕΝΑ

1. ΕΙΣΑΓΩΓΗ

1.1 ΤΟΠΟΘΕΤΗΣΗ ΚΑΙ ΑΝΑΛΥΣΗ ΤΟΥ ΠΡΟΒΛΗΜΑΤΟΣ

Ο σκοπός αυτής της εργασίας είναι η αντιμετώπιση σε πραγματικό χρόνο ενός προβλήματος βελτιστοποίησης χρησιμοποιώντας parser και γενετικούς αλγορίθμους. Το πρόβλημα αυτό προϋποθέτει την ύπαρξη ενός περιβάλλοντος όπου θα μπορεί να γίνει η εισαγωγή μαθηματικών σχέσεων, η ανάλυσή τους βάσει δοθέντων περιορισμών και η βελτιστοποίησή τους με χρήση ενός γενετικού αλγορίθμου.

Το πρώτο στάδιο του προβλήματος είναι η εισαγωγή και η ανάλυση των μαθηματικών σχέσεων. Θα πρέπει με κάποιο τρόπο να μεταφραστούν οι υπολογιστικές σχέσεις από τη μορφή που εισάγονται σε μία μορφή κατανοητή από το περιβάλλον. Για τη λειτουργία αυτή χρησιμοποιείται *ο συντακτικός αναλυτής* (parser). Επειδή ο ελληνικός αυτός όρος δεν αποδίδει πλήρως τη σχετική λειτουργία, θα χρησιμοποιηθεί η σχετική λέξη στα αγγλικά σε όλη την υπόλοιπη εργασία.

Όσον αφορά στη διαδικασία της βελτιστοποίησης, θα πρέπει να επιλεχθεί και να καταστρωθεί ο γενετικός αλγόριθμος που θα χρησιμοποιηθεί, δηλαδή να καθοριστούν οι παράμετροι και οι τεχνικές σύμφωνα με τις οποίες θα λειτουργήσουν οι *γενετικοί τελεστές* (genetic operators). Οι γενετικοί τελεστές είναι η *φυσική επιλογή* (selection), η *διασταύρωση* (crossover) και η *μετάλλαξη* (mutation) και ουσιαστικά αποτελούν τα εργαλεία υλοποίησης των *γενετικών αλγορίθμων* (Genetic Algorithms).

Ο parser και ο γενετικός αλγόριθμος αποτελούν τα εργαλεία υλοποίησης του υπολογιστικού περιβάλλοντος, το οποίο στόχο έχει να επιλύει απλά προβλήματα μεγιστοποίησης και ελαχιστοποίησης και βέβαια να είναι απλό και κατανοητό στη χρήση.

1.2 ΑΝΑΓΚΑΙΟΤΗΤΑ ΕΠΙΛΥΣΗΣ ΤΟΥ ΠΡΟΒΛΗΜΑΤΟΣ

Η αναγκαιότητα επίλυσης του προβλήματος αυτού είναι και η αφορμή της σύλληψης της ιδέας για το σχεδιασμό ενός προγραμματιστικού περιβάλλοντος και έγκειται στη δυνατότητα δήλωσης μεταβλητών (σχεδιαστικών παραμέτρων) και αντικειμενικής συνάρτησης “online”, δηλαδή τη στιγμή που τρέχει το περιβάλλον, και επίλυσης προβλημάτων *βελτιστοποίησης σε πραγματικό χρόνο* (Real Time Optimization). Σκοπός δηλαδή είναι να μπορεί ένας σχεδιαστής να λύσει πολλών ειδών προβλήματα βελτιστοποίησης μέσα από αυτό το περιβάλλον άμεσα και με απλό τρόπο.

Ένας άλλος πολύ σημαντικός παράγοντας που πρέπει να αναφερθεί είναι η συγκριτική αξιολόγηση εναλλακτικών σχεδιασμών και σχεδιαστικών προτάσεων που αποδίδει η επίλυσή του προβλήματος αυτού. Το στάδιο της αξιολόγησης και αποτίμησης εναλλακτικών σχεδιασμών αποτελεί ένα πολύ σημαντικό κομμάτι της διαδικασίας του σχεδιασμού ενός προϊόντος, καθώς οι αποφάσεις και οι επιλογές που γίνονται στο σημείο αυτό επηρεάζουν σε πολύ μεγάλο βαθμό όλα τα επόμενα βήματα της διαδικασίας, αλλά και γενικότερα την τελική εικόνα, ποιότητα και εν τέλει την αποτελεσματικότητα του τελικού αποτελέσματος σχετικά με τους στόχους, τις σχεδιαστικές προδιαγραφές και τους σχεδιαστικούς περιορισμούς που είχαν τεθεί εξ’αρχής.

Η τελική επιλογή της καλύτερης σχεδιαστικής πρότασης σχετίζεται προφανώς με τις σχεδιαστικές προδιαγραφές. Οι σχεδιαστικές προδιαγραφές επιβάλλουν συγκεκριμένα κριτήρια βάσει των οποίων γίνεται η σύγκριση των εναλλακτικών σχεδιασμών και βέβαια μπορεί μία πρόταση να υπερέχει των άλλων συγκεκριμένα ως προς ένα κριτήριο, ωστόσο να είναι παράλληλα υποδεέστερη των άλλων αναφορικά με τα λοιπά κριτήρια. Η διαδικασία λοιπόν της αποτίμησης αυτής θα πρέπει να είναι – και είναι συνήθως - πολυκριτηριακή.

1.3 ΤΡΟΠΟΣ – ΜΕΘΟΔΟΣ ΕΠΙΛΥΣΗΣ

Η εισαγωγή των υπολογιστικών σχέσεων του προβλήματος προς βελτιστοποίηση στον υπολογιστή γίνεται με τη βοήθεια ενός ήδη υπάρχοντος parser [22], ο οποίος λαμβάνει ως είσοδο μαθηματικές σχέσεις σε μορφή χαρακτήρων (strings), αποκωδικοποιεί τις σχέσεις αυτές, αναγνωρίζοντας τις μεταβλητές και τα σύμβολα και ύστερα υλοποιεί τις ζητούμενες μαθηματικές πράξεις. Η χρήση του parser επιτρέπει τη δήλωση μεταβλητών και την εισαγωγή υπολογιστικών σχέσεων σε πραγματικό χρόνο, τη στιγμή δηλαδή που τρέχει το πρόγραμμα. Αυτό σημαίνει ότι ο χρήστης μπορεί να εισάγει οποιαδήποτε σχέση επιθυμεί δηλώνοντας παράλληλα τις μεταβλητές. Έτσι, μπορεί να λυθεί μεγάλο εύρος προβλημάτων και όχι μόνο προβλήματα με συγκεκριμένο αριθμό μεταβλητών, συγκεκριμένο όνομα μεταβλητών κτλ.

Η μέθοδος που επιλέχθηκε για τη λύση του προβλήματος της βελτιστοποίησης καθαρά είναι η βελτιστοποίηση με χρήση γενετικών αλγορίθμων. Οι γενετικοί αλγόριθμοι συνιστούν μια γνωστή έξυπνη μέθοδο βελτιστοποίησης, που εφαρμόζεται σε απλές και σύνθετες εφαρμογές, βασίζεται στην ιδέα της φυσικής επιλογής και αποδίδει βέλτιστες λύσεις με μεγάλη ακρίβεια. Ως εξελικτικοί αλγόριθμοι, οι γενετικοί αλγόριθμοι δημιουργούν έναν αρχικό τυχαίο πληθυσμό λύσεων του προβλήματος και μέσω διαδικασιών εμπνευσμένων από τον κλάδο της βιολογίας και της γενετικής, όπως πχ διασταύρωση, φυσική επιλογή, προχωρούν δημιουργώντας νέες γενιές του πληθυσμού αυτού, με την κάθε επόμενη γενιά να δίνει όλο και καλύτερες λύσεις στο πρόβλημα στο οποίο χρησιμοποιούνται. Όταν η διαδικασία φτάσει στην καλύτερη λύση, ο αλγόριθμος τερματίζει. Η διαδικασία αυτή θα εξηγηθεί αναλυτικότερα στη συνέχεια της εργασίας.

Όλα αυτά τα χαρακτηριστικά θα αναπτυχθούν αναλυτικότερα στο κυρίως μέρος της εργασίας, όπου θα διαμορφωθεί ένα σχέδιο ανάπτυξης ενός προγραμματιστικού περιβάλλοντος για την εισαγωγή και ανάλυση υπολογιστικών σχέσεων υπό μορφή κειμένου με χρήση parser και τη βελτιστοποίησή τους με γενετικό αλγόριθμο που θα επιλεχθεί. Στη συνέχεια το περιβάλλον αυτό θα υλοποιηθεί και θα χρησιμοποιηθεί για την υλοποίηση βελτιστοποίησης σε δύο εφαρμογές - παραδείγματα.

ΚΕΦΑΛΑΙΟ 1 ΒΙΒΛΙΟΓΡΑΦΙΚΗ ΑΝΑΛΥΣΗ

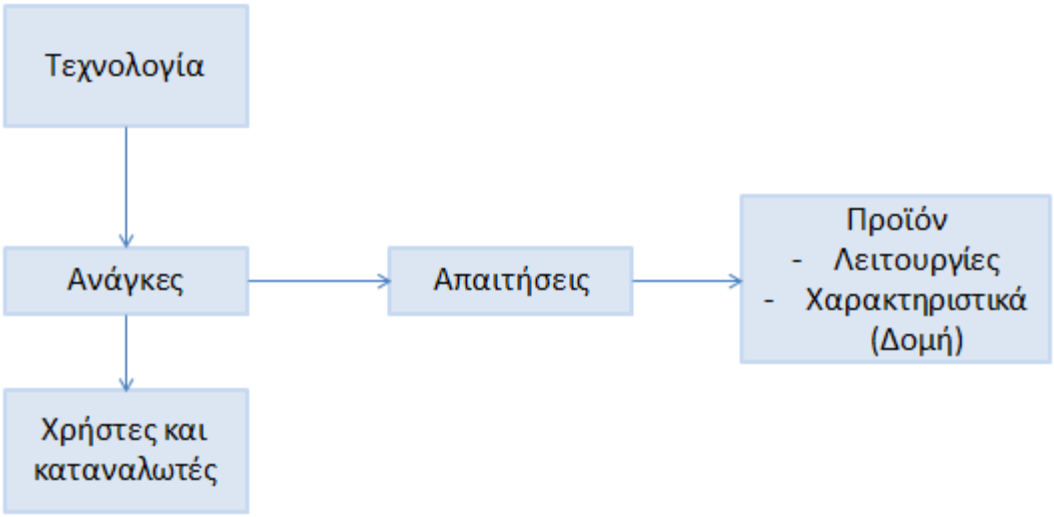
2.1 ΣΧΕΔΙΑΣΜΟΣ

Ξεκινώντας, θα πρέπει να αναφερθεί η γενική έννοια και ο ορισμός του σχεδιασμού.

Ορισμός :

«Σχεδιασμός είναι ένα σύνολο ενεργειών που έχουν σαν αφετηρία και απαραίτητη αρχική προϋπόθεση την αναγνώριση της ύπαρξης μιας αδιαμφισβήτητης ανάγκης ανάπτυξης ενός νέου αντικειμένου (product) για την ικανοποίηση μίας ή περισσοτέρων αναγκών και έχει σαν τελικό στόχο τον αναλυτικό προσδιορισμό όλων των χαρακτηριστικών και ιδιοτήτων του ώστε να είναι δυνατή η παραγωγή του». [1]

Συνεπώς, σύμφωνα με τον Δέντσορα (2014) [1] ο σχεδιασμός πηγάζει από ανάγκες και απαιτήσεις της αγοράς, μερικές από τις οποίες δημιουργούνται ή εκφράζονται από τους καταναλωτές ενώ άλλες ‘επιβάλλονται’ ουσιαστικά από την τεχνολογική εξέλιξη. Ξεκινάει λοιπόν έτσι η σχεδιαστική διαδικασία, η οποία με γνώμονα τις προαναφερθείσες ανάγκες και απαιτήσεις προσπαθεί να δημιουργήσει ένα προϊόν που να υλοποιεί συγκεκριμένες λειτουργίες και να έχει συγκεκριμένα χαρακτηριστικά. Οι λειτουργίες και τα χαρακτηριστικά αυτά προορίζονται βέβαια να ικανοποιούν τις ανάγκες και απαιτήσεις που έχουν διαμορφωθεί αρχικά.



Σχήμα 2.1. Η συσχέτιση ανάμεσα στις ανάγκες, τις απαιτήσεις και το προϊόν. [1]

Για το σχεδιαστικό πρόβλημα διαμορφώνονται διάφορες σχεδιαστικές προτάσεις από τη σχεδιαστική ομάδα, λαμβάνοντας υπόψιν τις σχεδιαστικές προδιαγραφές καθώς και τους σχεδιαστικούς περιορισμούς που έχουν τεθεί εξ' αρχής βάσει της φύσης του προϊόντος καθώς και των λειτουργιών που θα υλοποιεί. Οι σχεδιαστικές αυτές προτάσεις είναι οι εναλλακτικοί σχεδιασμοί που αναφέρθηκαν στο προηγούμενο κεφάλαιο, οι οποίοι εν συνεχεία θα συγκριθούν και θα επιλεχθεί ο καλύτερος με βάση τα δεδομένα του προβλήματος μέσω της διαδικασίας της βελτιστοποίησης.

Το σχεδιαστικό μοντέλο της παρούσης εργασίας βασίζεται στη χρήση του Η/Υ (Computer – based design model). Τα μοντέλα αυτά χρησιμοποιούν μεθόδους και τεχνικές που καθιστούν εφικτή :

i) την παροχή σχεδιαστικής βοήθειας από τον υπολογιστή στο σχεδιαστή μηχανικό και
ii) σε ορισμένες περιπτώσεις και κάτω από συγκεκριμένες προϋποθέσεις, τον αυτοδύναμο σχεδιασμό από την πλευρά του Η/Υ (πχ Autocad, Catia, Solidworks και λοιπά γνωστά σχεδιαστικά πακέτα απευθυνόμενα σε μηχανικούς και όχι μόνο).

Το μεγάλο πλεονέκτημα των μοντέλων που βασίζονται στη χρήση του Η/Υ είναι ότι επιτυγχάνουν καλύτερα αποτελέσματα σε ύστερα στάδια του σχεδιασμού αφού υλοποιούν παραμετρική και λεπτομερή ανάπτυξη του νέου προϊόντος και βέβαια αναφέρονται στις επιμέρους εργασίες που τα στάδια αυτά περιλαμβάνουν. Για τη βελτιστοποίηση της σχεδιαστικής λύσης, το οποίο είναι και το ζητούμενο, τα μοντέλα αυτά χρησιμοποιούν συμβατικές μεθόδους αλλά και μεθόδους της *υπολογιστικής νοημοσύνης* (computational intelligence). Μια τέτοια μέθοδος είναι και οι γενετικοί αλγόριθμοι.

2.2 ΒΕΛΤΙΣΤΟΠΟΙΗΣΗ ΤΟΥ ΣΧΕΔΙΑΣΜΟΥ

Πολλές φορές, στην πορεία του σχεδιασμού προκύπτει η ανάγκη μεγιστοποίησης ή ελαχιστοποίησης κάποιου μεγέθους, π.χ ελαχιστοποίηση του βάρους κάποιου εξαρτήματος, μεγιστοποίηση του μήκους του κτλ. Σε αυτήν την περίπτωση χρησιμοποιείται μία μέθοδος βελτιστοποίησης που χρησιμοποιεί μαθηματικές τεχνικές και μεθόδους προς επίτευξη ακρότατου σημείου, μεγίστου ή ελαχίστου. Συνήθως χρησιμοποιείται μία κατάλληλη αντικειμενική συνάρτηση, δηλαδή μια μαθηματική έκφραση που μας δίνει τη συσχέτιση του προαναφερθέντος μεγέθους με όλα τα μεγέθη που το επηρεάζουν, δηλαδή με τις σχεδιαστικές παραμέτρους. Έτσι η βελτιστοποίηση εδώ συνίσταται στην αναζήτηση των βέλτιστων τιμών των παραμέτρων που επηρεάζουν την αντικειμενική συνάρτηση, ώστε να πάρει τη μέγιστη ή ελάχιστη τιμή του, ανάλογα με το πρόβλημα. Η αναζήτηση αυτή γίνεται σύμφωνα με συγκεκριμένους περιορισμούς που έχουν τεθεί είτε από το σχεδιαστή είτε από άλλους παράγοντες. Οι περιορισμοί θέτουν όρια στις τιμές που μπορούν να πάρουν οι σχεδιαστικές παράμετροι και καθιστούν έτσι τη λύση του προβλήματος πιο συγκεκριμένη.

Είναι πάρα πολλές οι μέθοδοι βελτιστοποίησης που έχουν αναπτυχθεί ως τώρα. Πολύ γνωστή και χρήσιμη είναι η μέθοδος Simplex του George Dantzig [7], η οποία αποτελεί αλγόριθμο γραμμικού προγραμματισμού και χρησιμοποιείται κατά κόρον στην επιστήμη της Επιχειρησιακής Έρευνας και όχι μόνο. Μια μεγάλη κατηγορία μεθόδων βελτιστοποίησης είναι γενικά οι επαναληπτικές μέθοδοι, συμπεριλαμβανομένου των μεθόδων που χρησιμοποιούν

πεπερασμένες διαφορές, όπως η Newton –Raphson, και άλλων πολλών όπως οι quasi-Newton μέθοδοι κλπ [7]. Άλλη μεγάλη κατηγορία μεθόδων βελτιστοποίησης του σχεδιασμού αποτελούν οι ευρετικοί (ευριστικοί) αλγόριθμοι. Στην κατηγορία αυτή συμπεριλαμβάνονται πολλών ειδών αλγόριθμοι και κατά κύριο λόγο οι *Εξελικτικοί Αλγόριθμοι* (Evolutionary Algorithms). Ένα μικρό υποσύνολο των εξελικτικών αλγορίθμων είναι και οι Γενετικοί Αλγόριθμοι, τους οποίους χρησιμοποιεί η παρούσα εργασία.

Πρέπει να σημειωθεί εδώ επίσης ότι σε κάποια προβλήματα επαρκούν οι απλές αλγεβρικές μέθοδοι βελτιστοποίησης, οι οποίες είναι συνήθως και πιο απλές στην υλοποίηση, ωστόσο πολλές είναι οι περιπτώσεις στις οποίες κρίνεται αναγκαία η χρήση μη συμβατικών μεθόδων βελτιστοποίησης, όπως είναι και οι γενετικοί αλγόριθμοι.

2.3 ΓΕΝΕΤΙΚΟΙ ΑΛΓΟΡΙΘΜΟΙ

Πρόκειται για προσαρμόσιμους ευριστικούς αλγορίθμους οι οποίοι βασίζονται στον κλάδο της γενετικής και συνιστούν έναν έξυπνο τρόπο βελτιστοποίησης. Χρησιμοποιούν την ιδέα της εξέλιξης μέσω φυσικής επιλογής, γενετικής μετάλλαξης και διασταύρωσης και ουσιαστικά μιμούνται τις διαδικασίες αυτές σε υπολογιστικό – αλγοριθμικό επίπεδο. Άλλωστε ονομάζονται γενετικοί, και ανήκουν στην ευρύτερη κατηγορία των Εξελικτικών Αλγορίθμων. Ιδανικοί για πολυκριτηριακή – πολυπαραμετρική βελτιστοποίηση, οι γενετικοί αλγόριθμοι είναι ιδανικοί για απλά αλλά και για σύνθετα προβλήματα, για τα οποία δεν υπάρχει αναλυτική μέθοδος που να μπορεί να υπολογίσει το βέλτιστο συνδυασμό τιμών για τις μεταβλητές του υπό εξέταση συστήματος.

Ένας από τους πρώτους ερευνητές που ασχολήθηκαν με την ανάπτυξη της θεωρίας των γενετικών αλγορίθμων ήταν ο Holland (Holland 1975). Σημαντική έρευνα διεξήγε επίσης ο Goldberg (Goldberg 1989), ο οποίος διερεύνησε τη δυνατότητα αξιοποίησης των γενετικών αλγορίθμων σε βελτιστοποίηση πολύπλοκων συστημάτων σε μεγάλη κλίμακα. Αξίζει να σημειωθεί επίσης ο Bullock (Bullock et al. 1995), ο οποίος εστίασε στη χρήση των γενετικών αλγορίθμων συγκεκριμένα στο Engineering Design και τόνισε τα πλεονεκτήματά τους σε σχέση με άλλες μεθόδους. Οι γενετικοί αλγόριθμοι έχουν χρησιμοποιηθεί στο παρελθόν ως

σχεδιαστικά εργαλεία (Goldberg 2002) [2], αλλά επίσης και ως εργαλεία υποστήριξης σε συστήματα που βασίζονται στον Η/Υ και υλοποιούν λεπτομερή σχεδιασμό (Renner and Ekart 2003) [3].

Παρακάτω παρατίθενται τα πλεονεκτήματα των γενετικών αλγορίθμων: [4]

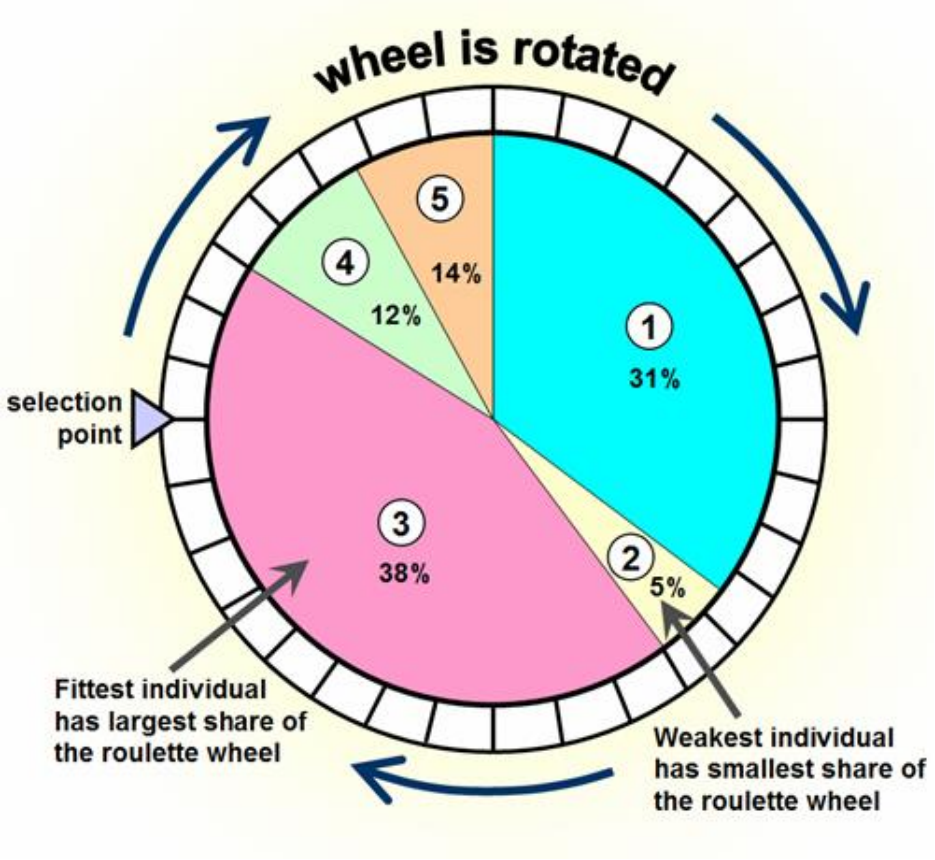
- Μπορούν να λύσουν δύσκολα προβλήματα γρήγορα και αξιόπιστα.
- Μπορούν να συνεργαστούν εύκολα με τα υπάρχοντα μοντέλα και συστήματα.
- Είναι εύκολα επεκτάσιμοι και εξελίξιμοι.
- Μπορούν να συμμετέχουν σε υβριδικές μορφές με άλλες μεθόδους.
- Εφαρμόζονται σε πολύ περισσότερα πεδία από κάθε άλλη μέθοδο.
- Δεν απαιτούν περιορισμούς στις συναρτήσεις που επεξεργάζονται.
- Δεν ενδιαφέρει η σημασία της υπό εξέτασης πληροφορίας.
- Έχουν από τη φύση τους το στοιχείο του παραλληλισμού.
- Είναι η μόνη μέθοδος που κάνει ταυτόχρονα εξερεύνηση του χώρου αναζήτησης και εκμετάλλευση της ήδη επεξεργασμένης πληροφορίας.
- Επιδέχονται παράλληλη υλοποίηση.

Πρέπει να τονιστεί εδώ το πολύ σημαντικό πλεονέκτημα της μη απαίτησης περιορισμών στις συναρτήσεις υπό επεξεργασία. Αυτό σημαίνει ότι δεν είναι απαραίτητη η ύπαρξη παραγώγου μιας συνάρτησης, η συνέχεια της κτλ. σε όλα τα σημεία του πεδίου ορισμού της, πράγμα που απαιτείται σε άλλες μεθόδους βελτιστοποίησης.

2.4 ΒΕΛΤΙΣΤΟΠΟΙΗΣΗ ΜΕ ΤΗ ΧΡΗΣΗ ΓΕΝΕΤΙΚΩΝ ΑΛΓΟΡΙΘΜΩΝ

Στην ενότητα αυτή περιγράφεται η μέθοδος των γενετικών αλγορίθμων που υλοποιεί τη βελτιστοποίηση, ενώ παρατίθεται και μια σχηματική αναπαράσταση των βημάτων εκτέλεσης του αλγορίθμου.

Πιο συγκεκριμένα, η μέθοδος ξεκινάει με ένα τυχαίο, μέσα σε κάποια προκαθορισμένα όρια, σύνολο λύσεων υπό δυαδική αναπαράσταση οι οποίες ονομάζονται *χρωμοσώματα* (chromosomes). Σχηματίζεται έτσι ένας πληθυσμός από λύσεις (population initialization) και υπολογίζεται για την κάθε μία η αντικειμενική συνάρτηση ή αλλιώς *συνάρτηση καταλληλότητας* (fitness function). Από τις λύσεις αυτές επιλέγονται οι βέλτιστες (selection) βάσει ενός κριτηρίου που εξαρτάται από το εκάστοτε πρόβλημα. Η συνηθέστερη μέθοδος Επιλογής είναι η *Επιλογή Ρουλέτας* (Roulette Wheel Selection).

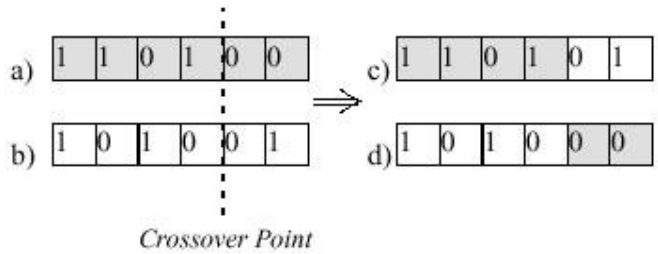


Σχήμα 2.2. Σχηματική Αναπαράσταση του Roulette Wheel Selection [5]

Στη μέθοδο Επιλογής Ρουλέτας, εφόσον οι τιμές της συνάρτησης καταλληλότητας του υπάρχοντος πληθυσμού έχουν υπολογιστεί, υπολογίζονται οι *κανονικοποιημένες τιμές*

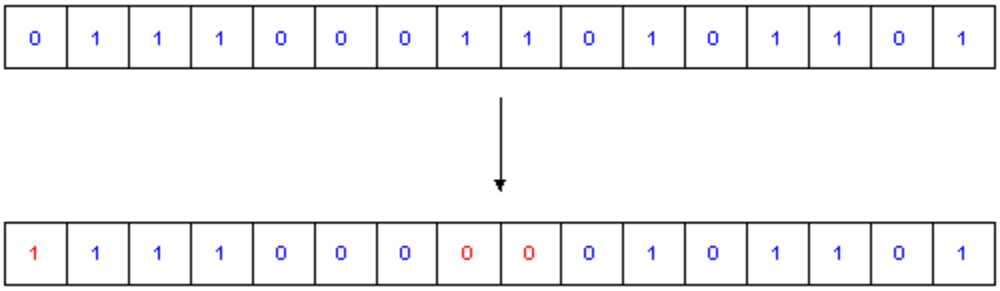
(normalized values) της συνάρτησης καταλληλότητας. Αυτές ισούνται με τις κανονικές τιμές καταλληλότητας διαιρεμένες με το συνολικό άθροισμα της συνάρτησης καταλληλότητας όλων των μελών του πληθυσμού, ώστε τελικά το άθροισμα όλων των κανονικοποιημένων τιμών να ισούται με τη μονάδα. Στη συνέχεια οι κανονικοποιημένες τιμές ταξινομούνται από τη μεγαλύτερη προς τη μικρότερη και υπολογίζονται οι *συσσωρευμένες τιμές* (accumulated values). Συσσωρευμένη τιμή μιας λύσης είναι η κανονικοποιημένη της τιμή προστιθέμενη στις κανονικοποιημένες τιμές καταλληλότητας όλων των προηγούμενων μελών του πληθυσμού, εφόσον βέβαια έχει γίνει η προαναφερθείσα ταξινόμηση. Στο τέλος της διαδικασίας υπολογισμού των συσσωρευμένων τιμών, η τελευταία τιμή θα πρέπει να ισούται με τη μονάδα, εάν η διαδικασία έχει γίνει σωστά. Στο σημείο αυτό επιλέγεται ένας τυχαίος πραγματικός αριθμός R στο διάστημα [0,1] και ξεκινάει έτσι μια διαδικασία που μοιάζει πάρα πολύ με τη γνωστή ρουλέτα. Παίρνοντας ένα σταθερό (fixed) σημείο, γίνεται σειριακά προσπέλαση όλων των συσσωρευμένων κανονικοποιημένων τιμών καταλληλότητας και η λύση της οποίας η τιμή είναι η τελευταία μικρότερη από τον αριθμό R είναι και η επιλεχθείσα λύση [6]. Με τη λογική αυτή βέβαια, όπως φαίνεται και στο παραπάνω σχήμα, λύσεις με μεγαλύτερες αλγεβρικά τιμές καταλληλότητας έχουν αναλογικά μεγαλύτερη πιθανότητα να επιλεγούν στη διαδικασία, ενώ λύσεις με μικρές τιμές καταλληλότητας αρκετά μικρή αλλά και όχι μηδαμινή πιθανότητα, αφού η όλη διαδικασία είναι ως γνωστόν βασισμένη στην τύχη. Η διαδικασία αυτή επαναλαμβάνεται για διαφορετικούς τυχαίους κάθε φορά αριθμούς R, μέχρις ότου έχει επιλεχθεί ο ζητούμενος ή επιθυμητός αριθμός λύσεων.

Εν συνεχεία οι επιλεχθείσες αυτές λύσεις υπόκεινται στη διαδικασία της διασταύρωσης, ανταλλάσσουν δηλαδή ανά ζεύγη μερικά από τα 0 και 1 (ανάλογα με το *επιλεγμένο σημείο διασταύρωσης* - crossover point) που έχουν στα χρωμοσώματά τους και έτσι σχηματίζονται νέες λύσεις. Το σημείο διασταύρωσης συνήθως επιλέγεται αυθαίρετα, ενώ ο ρόλος που παίζει στη φάση της Διασταύρωσης φαίνεται στην παρακάτω εικόνα.



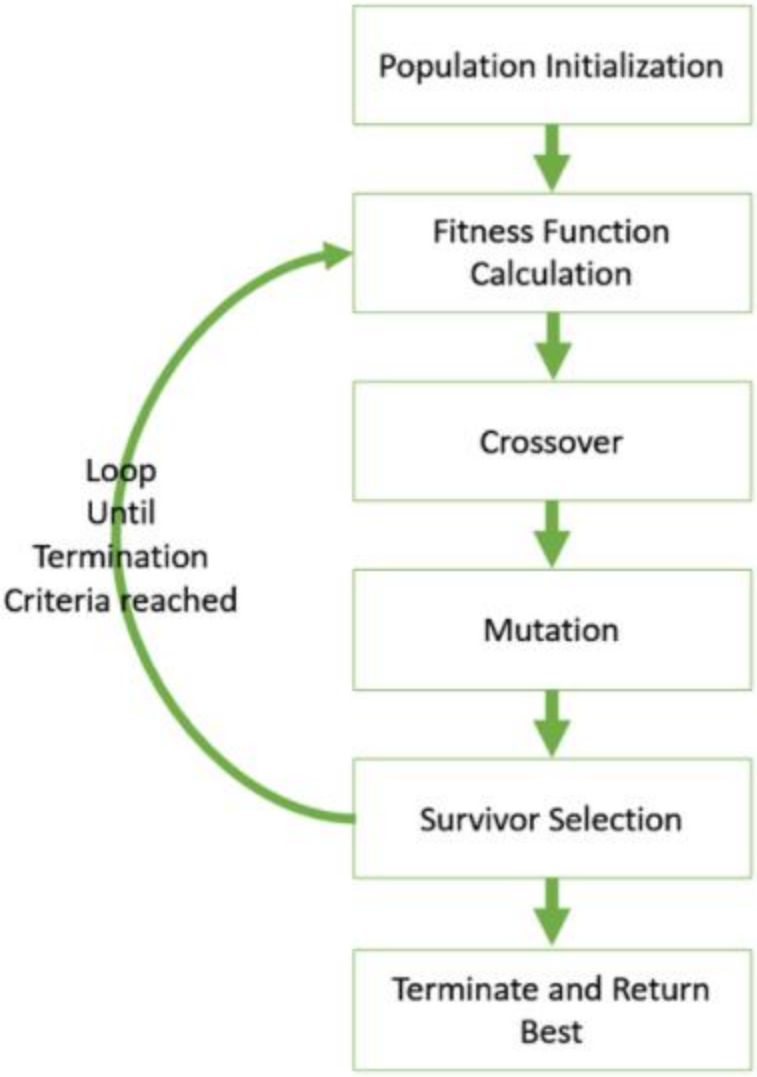
Σχήμα 2.3. Η διαδικασία της Διασταύρωσης με ένα σημείο (Single – Point Crossover) ανάμεσα σε 2 τυχαία χρωμοσώματα [7]

Έπειτα ακολουθεί η διαδικασίας της Μετάλλαξης, η οποία χρησιμοποιείται ώστε να διατηρηθεί η διαφορετικότητα από γενιά σε γενιά. Ένα μικρό ποσοστό επιλέγεται ανά γενιά για του οποίου τις λύσεις γίνεται η ακόλουθη διαδικασία: Κάθε ψηφίο (bit) του χρωμοσώματος έχει μια πιθανότητα P να αντιστραφεί, δηλαδή να αλλάξει το 0 σε 1 ή το 1 σε 0 . Η τυχαιότητα της επιλογής των ψηφίων που θα διαφοροποιηθούν εξασφαλίζει τη διαφορετικότητα των πληθυσμών από γενιά σε γενιά (population diversity).



Σχήμα 2.4. Παράδειγμα Μετάλλαξης όπου 2 bits ανετράπησαν και άλλαξαν τη δυαδική τους τιμή [8]

Η διαδικασία αυτή, από το selection και έπειτα, επαναλαμβάνεται μέχρις ότου βρεθεί η βέλτιστη ή πιο ικανοποιητική λύση ή για συγκεκριμένο αριθμό επαναλήψεων. Δημιουργούνται έτσι *«γενιές»* (generations) λύσεων και σκοπός είναι, η κάθε επόμενη γενιά να έχει λύσεις πιο κοντά στη βέλτιστη. Στο τέλος της όλης διαδικασίας, εάν αυτή γίνει σωστά, θα «επιβιώσουν» οι καλύτερες – βέλτιστες λύσεις (survival of the fittest).



Σχήμα 2.5. Ο γενετικός αλγόριθμος σε βήματα [9]

2.5 PARSING - ΟΡΙΣΜΟΣ, ΚΑΤΗΓΟΡΙΕΣ

Αρχικά θα πρέπει να δοθεί ένας συνοπτικός ορισμός του parser, καθώς είναι μια έννοια άγνωστη σε αρκετό κόσμο.

Ο parser είναι ένας συντακτικός αναλυτής, ένα πρόγραμμα το οποίο παίρνει ως είσοδο ένα σύνολο από χαρακτήρες, νούμερα, σύμβολα κλπ. και τα χωρίζει μεταξύ τους ώστε αυτά να αποκτούν νόημα ως ξεχωριστές οντότητες υπό το πλαίσιο της γραμματικής και της σύνταξης της γλώσσας στην οποία είναι ενταγμένος ή χρησιμοποιείται εν προκειμένω. Πιο συγκεκριμένα, μπορεί να λάβει ως είσοδο μαθηματικές σχέσεις υπό μορφή κειμένου, να αναγνωρίσει τις μεταβλητές και τα σύμβολα και να υλοποιήσει τις σχετικές πράξεις.

Πολλοί ορίζουν τον parser ως *διαχωριστή* (separator), με την έννοια ότι διαχωρίζει την πρόταση που παίρνει ως είσοδο σε λέξεις, φράσεις, αριθμούς και γενικά γνωστές δομές δεδομένων ώστε τελικά να κατασκευάσει ένα parse tree στο οποίο τα απεικονίζει όλα αυτά, τακτοποιημένα και ιεραρχημένα. Δίνει λοιπόν μια εποπτική ματιά της συμβολοσειράς (string) που εισάγεται από το χρήστη, μία δομική αναπαράσταση, δίνοντας νόημα στα κομμάτια που απαρτίζουν την πρόταση αυτή μέσα στο πλαίσιο της γλώσσας προγραμματισμού που χρησιμοποιείται. [10]

Οι κύριες λειτουργίες ενός parser είναι γενικά οι παρακάτω:

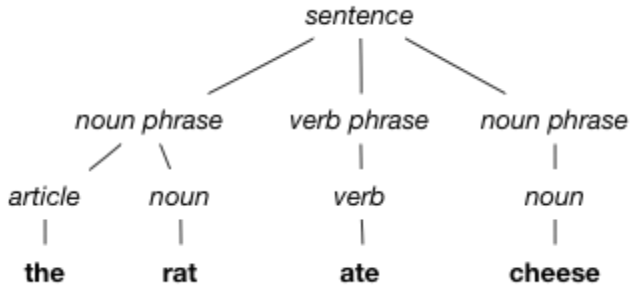
- *Λεξιλογική Ανάλυση* (Lexical Analysis), η οποία έχει την έννοια που εξηγήσαμε παραπάνω, της διαμόρφωσης εκφράσεων που έχουν νόημα, χωρίζοντας σε κομμάτια το string που δίνεται ως είσοδος.
- *Συντακτική Ανάλυση* (Syntactic Analysis). Εδώ γίνεται έλεγχος για το αν οι εκφράσεις που καταχωρήθηκαν από το προηγούμενο στάδιο βγάζουν στην πραγματικότητα νόημα ως σύνολο και μπαίνουν σε μια λογική σειρά.
- *Σημασιολογικό Parsing* (Semantic Parsing), όπου καθορίζεται και εδραιώνεται το νόημα και οι ιδιότητες των διαμορφωμένων από τα προηγούμενα στάδια εκφράσεων ενώ λαμβάνουν χώρα οι απαραίτητες ενέργειες. [11]

Υπάρχουν 2 ειδών parsers.

- Εκείνοι που υλοποιούν τη σχετική διαδικασία *από επάνω προς τα κάτω* (top-down), δηλαδή προσπελαύνουν τη δοθείσα σχέση χαρακτήρων από τα αριστερά προς τα δεξιά και τη σπάνε σε κομμάτια. Σε αυτήν την κατηγορία συμπεριλαμβάνονται οι LL parsers και οι recursive-descent parsers.
- Εκείνοι που υλοποιούν τη σχετική διαδικασία *από κάτω προς τα επάνω* (bottom-up) ή αλλιώς Shift-Reduce parsing, δηλαδή ξεκινούν βρίσκοντας τα πιο βασικά στοιχεία της σχέσης που παίρνουν ως input και εν συνεχεία στοιχεία που περιέχουν τα προηγούμενα στοιχεία. Τέτοιου είδους parsers είναι οι LR parsers. [10],[11],[14]

Υπάρχει η γενική πεποίθηση ότι οι από επάνω προς τα κάτω parsers απαιτούν μεγάλο υπολογιστικό χρόνο όταν προσπελαύνουν μια σχέση από τα αριστερά προς τα δεξιά, ωστόσο οι Frost, Hafiz και Callaghan (Frost, Hafiz, Callaghan, 2008) [12] έχουν δημιουργήσει πολυσύνθετους αλγορίθμους που βελτιώνουν την ταχύτητα των υπολογισμών, καθιστώντας τους χρόνους υπολογισμού από εκθετικούς σε πολυωνυμικούς, αυξάνοντας παράλληλα το επίπεδο αφαιρετικότητας.

Μία πολύ ενδιαφέρουσα ερμηνεία του parsing δίνει το Cornell University μέσα από ένα σχετικό μάθημά του [13]. Αξίζει να σημειωθεί η παρομοίωση του ανθρώπινου μυαλού με parser μέσα από την επόμενη εικόνα. Όταν ένας άνθρωπος ακούσει παραδείγματος χάρη την πρόταση “the rat ate cheese”, τότε θα σχηματίσει στο μυαλό του το ακόλουθο σχεδιάγραμμα σχετικά με τις λέξεις που άκουσε.



Σχήμα 2.6. Παρομοίωση του ανθρώπινου μυαλού με parser [13]

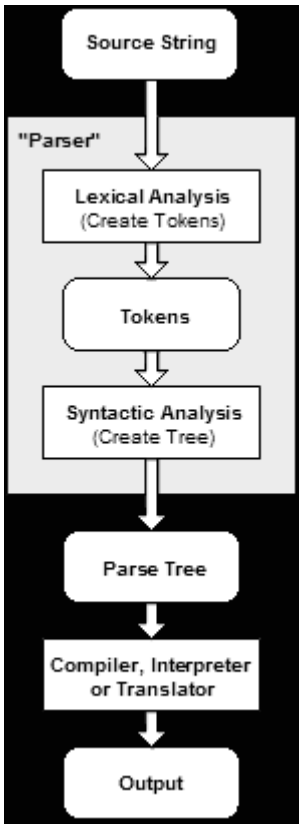
2.6 PARSING – ΑΝΑΛΥΣΗ ΚΑΙ ΕΚΤΕΛΕΣΗ ΥΠΟΛΟΓΙΣΜΩΝ

Στην ενότητα αυτή θα παρουσιαστεί η διαδικασία της ανάλυσης μιας υπολογιστικής σχέσης από έναν parser. Στο παράδειγμα που θα παρατεθεί, η γλώσσα προγραμματισμού υπό το πλαίσιο της οποίας τίθεται σε λειτουργία ο parser έχει 2 επίπεδα γραμματικής, το λεξιλογικό και το συντακτικό.

Στο πρώτο στάδιο, το οποίο όπως αναφέρθηκε είναι η λεξιλογική ανάλυση, το εισαγόμενο string χαρακτήρων διασπάται σε σύμβολα τα οποία έχουν κάποιο νόημα. Το νόημα αυτό διευκρινίζεται από μια γραμματική που ορίζεται από γνωστές – «κανονικές» εκφράσεις – σύμβολα. Για παράδειγμα, σε έναν parser που υλοποιεί υπολογισμό μαθηματικών πράξεων [14], εάν δοθεί ως input το string: *“10*(2+4)^2”,* θα χωριστεί στις ακόλουθες εκφράσεις ή αλλιώς tokens όπως λέγονται στην ορολογία των parsers: 10 , * , (, 2 , + , 4 ,) , ^ , 2. Η κάθε έκφραση που προέκυψε έχει προφανώς νόημα σε μαθηματικό επίπεδο, αφού όπως φαίνεται υπάρχουν 4 νούμερα, 2 παρενθέσεις και 3 σύμβολα μαθηματικών πράξεων. Εδώ πρέπει να σημειωθεί ότι εκφράσεις όπως πχ 10* ή (2 δεν προκύπτουν, διότι τα σύμβολα * , + , ^ , (,) σηματοδοτούν την έναρξη νέας έκφρασης στη γλώσσα του parser.

Στο δεύτερο στάδιο (συντακτική ανάλυση) ελέγχεται εάν οι εκφράσεις που προέκυψαν μπορούν να διαμορφώσουν μία αποδεκτή συνολική έκφραση. Αυτό γίνεται με τη βοήθεια μιας ανεξάρτητης από τα συμφραζόμενα γραμματικής. Παρ’όλα αυτά η γραμματική που χρησιμοποιεί ο εκάστοτε parser δεν μπορεί να προβλέψει πάντα όλους τους πιθανούς κανόνες που πρέπει να τηρεί η σχέση που διαμορφώνουν οι εκφράσεις, απαιτείται λοιπόν λεπτομερής ανάγνωση του documentation του parser που χρησιμοποιείται ή αναζήτηση κάποιας custom γραμματικής που καλύπτει τις εκάστοτε απαιτήσεις.

Στο τελευταίο στάδιο (Semantic Parsing), όπου λαμβάνονται οι απαραίτητες ενέργειες, στην περίπτωση του παραδείγματος έχουμε calculator parser. Συνεπώς στο στάδιο αυτό γίνεται ο υπολογισμός της μαθηματικής σχέσης (evaluation), όπως αυτή έχει διαμορφωθεί από τα 2 προηγούμενα στάδια και βεβαίως πρέπει κ εδώ να τονιστεί ότι τα μαθηματικά σύμβολα και οι συγκεκριμένες εκφράσεις που αναγνωρίζονται και υλοποιούνται στο στάδιο αυτό εξαρτώνται από τον εκάστοτε parser και τον τρόπο που ο προγραμματιστής τον έχει κατασκευάσει. Γι’αυτό απαιτείται η ανάγνωση του documentation, ώστε να εξασφαλισθεί τι έκφραση είναι επιτρεπτή, τι μαθηματική πράξη σηματοδοτείται και με τι σύμβολο κτλ.



Σχήμα 2.7. Η λειτουργία του parser σε βήματα [14]

2.7 ΠΡΟΣΑΡΜΟΣΤΙΚΑ ΠΕΡΙΒΑΛΛΟΝΤΑ ΥΛΟΠΟΙΗΣΗΣ ΥΠΟΛΟΓΙΣΜΩΝ ΚΑΙ ΒΕΛΤΙΣΤΟΠΟΙΗΣΗΣ ΣΕ ΠΡΑΓΜΑΤΙΚΟ ΧΡΟΝΟ

Έχουν γίνει πολλές προσπάθειες δημιουργίας περιβαλλόντων υπολογισμών και βελτιστοποίησης. Μάλιστα η Mathworks παρέχει το Optimization Toolbox καθώς και το Optimization App, τα οποία είναι εφαρμογές που τρέχουν στο Matlab και μπορούν να υλοποιήσουν βελτιστοποίηση με διάφορες μεθόδους. Το documentation των εφαρμογών αυτών είναι πολύ λεπτομερές και βοηθητικό.

Όσον αφορά τη βελτιστοποίηση σε πραγματικό χρόνο πολύ ενδιαφέρον παρουσιάζει το paper των Park, Pan και Manocha (Park, Pan, Manocha 2013) [15], το οποίο περιγράφει τη χρήση ενός περιβάλλοντος βελτιστοποίησης σε πραγματικό χρόνο με σκοπό την αποφυγή εμποδίων και το motion planning ρομπότ σε δυναμικά περιβάλλοντα χρησιμοποιώντας GPUs. Με τη βελτιστοποίηση σε πραγματικό χρόνο ασχολήθηκε και ο Christos G. Cassandras (Cassandras 2011, Boston University) [16], εστιάζοντας σε στοχαστικά συστήματα και μειώνοντας το χρόνο που απαιτείται για τους υπολογισμούς. Με αφορμή μια άλλη μηχανολογική εφαρμογή, οι Elixmann, Puschke και άλλοι συγγραφείς (Elixmann, Puschke et al. 2014) [17] δημιούργησαν ένα προγραμματιστικό περιβάλλον που υλοποιεί Non-linear Model Predictive Control και dynamic real-time optimization χημικών και ενεργειακών διεργασιών. Το περιβάλλον αυτό λέγεται OptoEcon-Toolbox και δίνει πολλές επιλογές και δυνατότητες χάρη στο παραμετρικό του σχεδιασμό. Τέλος, πρέπει να αναφερθεί ο NEOS (Network – Enabled Optimization System) Server for Optimization [18], μία εφαρμογή που παρέχει πρόσβαση σε μια βιβλιοθήκη από optimization solvers και που είναι ιδιαίτερα user-friendly, καθώς απαιτεί από το χρήστη την περιγραφή του προβλήματος που θέλει να λύσει και την επιλογή της μεθόδου, μέσω της επιλογής του optimization solver που αυτός επιθυμεί.

2.8 ΣΥΝΟΨΗ ΒΙΒΛΙΟΓΡΑΦΙΚΗΣ ΑΝΑΛΥΣΗΣ

Στην ενότητα αυτή έγινε αναφορά σε βασικά στοιχεία σχεδιασμού, βελτιστοποίησης και parsing. Παρατέθηκαν πληροφορίες για τα σχεδιαστικά μοντέλα που βασίζονται στη χρήση του Η/Υ και περιεγράφηκε συντόμως η μέθοδος βελτιστοποίησης με τη χρήση γενετικών αλγορίθμων, με την οποία ασχολείται η παρούσα εργασία. Επίσης έγινε αναφορά στους parsers, σε προγράμματα – εργαλεία που λαμβάνουν ως είσοδο ένα σύνολο χαρακτήρων το οποίο και αναλύουν σε απλούστερα και κατανοητά από την εκάστοτε γλώσσα προγραμματισμού σύμβολα και δομές. Εν συνεχεία εξετάστηκαν τα ήδη υπάρχοντα περιβάλλοντα που εκτελούν βελτιστοποίηση σε πραγματικό χρόνο και αναφέρθηκαν και ενδεικτικές εφαρμογές στις οποίες τα περιβάλλοντα αυτά μπορούν να φανούν χρήσιμα και να δώσουν άμεσα λύση σε προβλήματα βελτιστοποίησης, μηχανολογικής φύσεως και όχι μόνο.

2.9 ΕΠΑΝΑΤΟΠΟΘΕΤΗΣΗ ΤΟΥ ΠΡΟΒΛΗΜΑΤΟΣ ΚΑΙ ΑΝΑΦΟΡΑ ΣΤΑ ΕΠΟΜΕΝΑ ΚΕΦΑΛΑΙΑ

Στόχος της εργασίας αυτής είναι η βελτιστοποίηση υπολογιστικών σχέσεων χρησιμοποιώντας ένα σύνολο εργαλείων το οποίο συμπεριλαμβάνει έναν parser και τη μέθοδο βελτιστοποίησης με χρήση γενετικών αλγορίθμων.

Στο επόμενο κεφάλαιο το πρόβλημα θα καταστρωθεί και θα αναλυθεί λεπτομερώς, εστιάζοντας στις σχεδιαστικές παραμέτρους. Θα εξηγηθεί η χρησιμότητα ενός parser για τέτοιου τύπου εφαρμογές και γενικά για εφαρμογές βελτιστοποίησης και θα χρησιμοποιηθεί ένας parser για την εισαγωγή και την ανάλυση της υπολογιστικής σχέσης του εν λόγω προβλήματος. Έπειτα θα καταστρωθεί η κεντρική ιδέα της εργασίας, η οποία είναι η δημιουργία ενός περιβάλλοντος Η/Υ στο οποίο θα γίνεται:

- Εισαγωγή σχεδιαστικών παραμέτρων
- Εισαγωγή και ανάλυση υπολογιστικών σχέσεων επί των ανωτέρω σχεδιαστικών παραμέτρων με χρήση parser
- Βελτιστοποίηση ως προς τις ανεξάρτητες σχεδιαστικές παραμέτρους με τη χρήση γενετικών αλγορίθμων
- Παρουσίαση των αποτελεσμάτων

Στη συνέχεια θα περιγραφεί αναλυτικά το περιβάλλον και ο τρόπος λειτουργίας του, θα εφαρμοσθεί η σχεδιασθείσα μεθοδολογία βελτιστοποίησης σε δύο εφαρμογές και θα παρατεθούν αποτελέσματα σε μορφή διαγραμμάτων. Επίσης θα παρουσιαστούν εικόνες από τη λειτουργία του προγράμματος και θα εξηγηθούν οι δυνατότητές του.

Τέλος, θα συζητηθούν τα πλεονεκτήματα χρήσης ενός τέτοιου περιβάλλοντος και θα συζητηθούν περαιτέρω προοπτικές βελτίωσης του συγκεκριμένου περιβάλλοντος. Αξίζει να σημειωθεί ότι στο παράρτημα που βρίσκεται στο τέλος της εργασίας παρατίθενται ο κώδικας δημιουργίας του περιβάλλοντος στη γλώσσα Visual Basic, καθώς και μία σύντομη εξήγηση της λειτουργίας του parser[22] που χρησιμοποιήθηκε.

ΚΕΦΑΛΑΙΟ 2 ΚΥΡΙΩΣ ΜΕΡΟΣ – ΚΕΝΤΡΙΚΗ ΙΔΕΑ

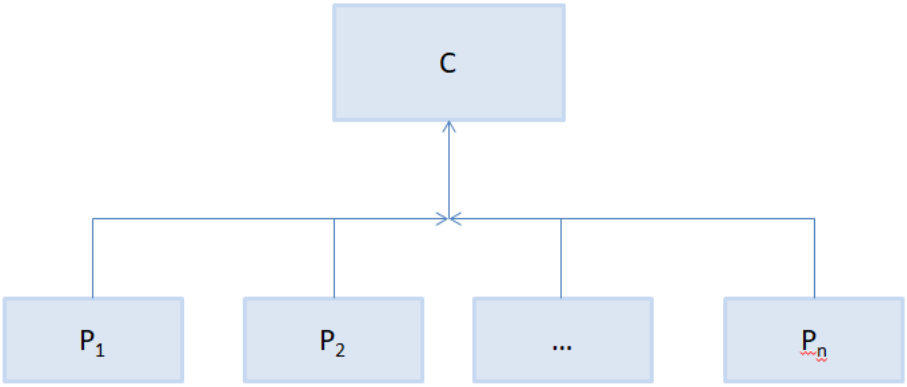
3.1 ΑΝΑΠΑΡΑΣΤΑΣΗ ΣΧΕΔΙΑΣΤΙΚΗΣ ΓΝΩΣΗΣ – ΣΧΕΔΙΑΣΤΙΚΕΣ ΠΑΡΑΜΕΤΡΟΙ

Έστω $P_1,P_2,\ldots,P_n$ οι *σχεδιαστικές παράμετροι* (Design Parameters) που υπεισέρχονται στη διαδικασία σχεδιασμού ενός προϊόντος. Ανάλογα με τους σχεδιαστικούς περιορισμούς και τις σχεδιαστικές προδιαγραφές που έχουν τεθεί από το σχεδιαστή ή τη σχεδιαστική ομάδα, αναζητούνται οι τιμές των σχεδιαστικών παραμέτρων που ικανοποιούν τα παραπάνω κριτήρια. Οι σχεδιαστικοί περιορισμοί και οι προδιαγραφές δηλαδή διαμορφώνουν τις απαιτήσεις και θέτουν ορισμένα όρια βάσει των οποίων ρυθμίζονται οι επιτρεπτές τιμές των σχεδιαστικών παραμέτρων. Το πεδίο τιμών των επιτρεπτών τιμών των σχεδιαστικών παραμέτρων περιορίζεται επίσης σύμφωνα με τον τύπο της κάθε παραμέτρου. Πιο συγκεκριμένα καθένα από τα μεγέθη $P_1,P_2,\ldots,P_n$ μπορεί να παίρνει συνεχείς τιμές, διακριτές τιμές, ή ακόμα και να είναι σταθερά. Συνεπώς το κομμάτι αυτό του σχεδιασμού μελετάει τις σχέσεις που υπάρχουν μεταξύ των σχεδιαστικών παραμέτρων για το εξεταζόμενο σχεδιαστικό πρόβλημα και μέσω της ανάλυσης των σχέσεων αυτών, οι τιμές που θα υπολογισθούν για τις παραμέτρους θα ρυθμίσουν επίσης και τα χρησιμοποιούμενα υλικά κατασκευής, τη γεωμετρία και άλλα χρήσιμα χαρακτηριστικά για την κατασκευή του υπό σχεδιασμού προϊόντος.

Οι σχεδιαστικές παράμετροι $P_1,P_2,\ldots,P_n$ σχηματίζουν μεταξύ τους μία σχέση της μορφής:

$$\boldsymbol{C = f(P_1, P_2, \ldots, P_n)}$$

Εξετάζεται λοιπόν η σχέση αυτή των παραμέτρων $P_1,P_2,\ldots,P_n$ και η υπολογιζόμενη ποσότητα C θεωρείται ένα μέτρο σύγκρισης.



Σχήμα 3.1. Διάγραμμα σχεδιαστικών παραμέτρων

Η τιμή του C είναι δηλαδή το κριτήριο με βάση το οποίο θα γίνει η αξιολόγηση των τιμών των λοιπών σχεδιαστικών παραμέτρων που θα δοκιμαστούν στη συνέχεια, για την αξιολόγηση των εναλλακτικών σχεδιασμών.

3.2 ΥΠΟΛΟΓΙΣΤΙΚΕΣ ΣΧΕΣΕΙΣ – ΣΧΕΔΙΑΣΤΙΚΕΣ ΠΑΡΑΜΕΤΡΟΙ ΒΕΛΤΙΣΤΟΠΟΙΗΣΗΣ

Η διαδικασία βελτιστοποίησης που θα γίνει, θα έχει σύμφωνα με τα προηγούμενα ως *αντικειμενική συνάρτηση* (objective function) τη μαθηματική σχέση που εκφράζει την τιμή του C, εφόσον αποφασίστηκε ότι το C θα είναι το μέτρο σύγκρισης. Η αντικειμενική συνάρτηση, ως γνωστόν είναι μια συνάρτηση της οποίας αναζητείται το μέγιστο ή το ελάχιστο κατά τη διαδικασία της βελτιστοποίησης. Για να έχει νόημα η διαδικασία αυτή πρέπει να τεθούν κάποιοι περιορισμοί επί των σχεδιαστικών παραμέτρων. Στο πρόβλημα της εργασίας αυτής οι περιορισμοί θα είναι *μοναδιαίοι* (unary constraints), θα περιλαμβάνουν δηλαδή μία μεταβλητή.

Οι μοναδιαίοι περιορισμοί έχουν να κάνουν με το εύρος των επιτρεπτών τιμών που μπορούν να πάρουν οι σχεδιαστικές παράμετροι και είναι της μορφής:

$$P_1^{min} \leq P_1 \leq P_1^{max}$$
$$P_2^{min} \leq P_2 \leq P_2^{max}$$
$$\dots$$
$$P_n^{min} \leq P_n \leq P_n^{max}$$

Όπου οι τιμές με δείκτη min παραπέμπουν σε ελάχιστη επιτρεπτή τιμή και οι τιμές με δείκτη max παραπέμπουν σε μέγιστη επιτρεπτή τιμή.

Στο πλαίσιο της βελτιστοποίησης με χρήση γενετικών αλγορίθμων, η αντικειμενική συνάρτηση ονομάζεται συνάρτηση καταλληλότητας και έχει το ίδιο νόημα. Η συνάρτηση καταλληλότητας, η οποία αποτελείται από τις προαναφερθείσες σχεδιαστικές παραμέτρους του προβλήματος, θα υποβληθεί σε μια επαναληπτική διαδικασία, η οποία μέσω των περιορισμών που θα εφαρμοστούν από τον σχεδιαστή σχετικά με το εύρος των τιμών τους, θα δώσει ως αποτέλεσμα το βέλτιστο συνδυασμό τιμών των σχεδιαστικών παραμέτρων. Βέβαια οι τιμές που θα δώσει η διαδικασία εξαρτώνται από το εάν ο σχεδιαστής επιθυμεί μεγιστοποίηση ή ελαχιστοποίηση της συνάρτησης καταλληλότητας άρα κατ'επέκταση της ποσότητας C.

Με την περαίωση της διαδικασίας βελτιστοποίησης λοιπόν, θα προκύψει ένα σετ τιμών για την ποσότητα C που ενδιαφέρει το σχεδιαστή, καθώς και για τις υπόλοιπες σχεδιαστικές παραμέτρους. Ουσιαστικά η μέθοδος βελτιστοποίησης με γενετικό αλγόριθμο θα δώσει το καλύτερο χρωμόσωμα, το οποίο βέβαια αποτελείται από 3 γονιδιώματα, τα οποία περιλαμβάνουν ακολούθως τιμές των σχεδιαστικών παραμέτρων P_1,P_2,…,P_n.

3.3 ΑΝΑΠΤΥΞΗ ΚΕΝΤΡΙΚΗΣ ΙΔΕΑΣ : ΔΗΜΙΟΥΡΓΙΑ ΕΝΟΣ ΠΕΡΙΒΑΛΛΟΝΤΟΣ Η/Υ

Η κεντρική ιδέα της εργασίας αυτής είναι ο σχεδιασμός και η δημιουργία ενός περιβάλλοντος που βασίζεται στον Η/Υ, στο οποίο αρχικά θα εισάγονται οι σχεδιαστικές παράμετροι και οι τιμές τους, καθώς και οι υπολογιστικές σχέσεις που περιγράφουν το μαθηματικό τρόπο σύνδεσης των σχεδιαστικών παραμέτρων. Στη συνέχεια οι υπολογιστικές σχέσεις θα αναλύονται με τη χρήση ενός parser, ο οποίος θα έχει λάβει εξ'αρχής πληροφορίες για τις σχεδιαστικές παραμέτρους, τα ονόματα και τις επιτρεπτές τιμές τους και τελικά σύμφωνα με όλα αυτά τα δεδομένα θα εκτελείται βελτιστοποίηση ως προς τις ανεξάρτητες σχεδιαστικές παραμέτρους – μεταβλητές επίδοσης. Η βελτιστοποίηση θα γίνει με τη χρήση γενετικών αλγορίθμων και η εφαρμογή θα ζητά από το χρήστη επίσης προαιρετικά κάποια στοιχεία για την υλοποίηση της βελτιστοποίησης. Στο τέλος της διαδικασίας τα αποτελέσματα θα παρουσιάζονται με τη μορφή γραφημάτων τα οποία θα δείχνουν την πορεία της εξέλιξης του εφαρμοζόμενου γενετικού αλγόριθμου, τη βέλτιστη τιμή δηλαδή της συνάρτησης καταλληλότητας συναρτήσει του αριθμού των επαναλήψεων σε κάθε σημείο.

Στις επόμενες σελίδες ακολουθεί η περιγραφή κάθε τμήματος που θα απαρτίζει το περιβάλλον προς σχεδιασμό, ώστε να διαμορφωθεί μία γενική ιδέα για τη δομή του interface και τις λειτουργίες που θα υλοποιεί.

Εισαγωγή σχεδιαστικών παραμέτρων

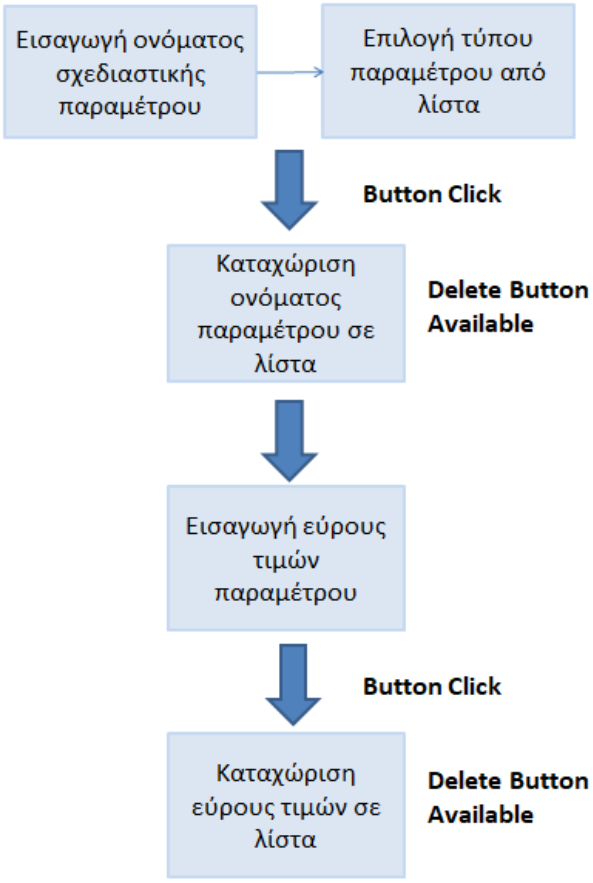
Το πρώτο και κυριότερο κομμάτι του περιβάλλοντος πρέπει να είναι η δήλωση των σχεδιαστικών παραμέτρων του προβλήματος που επιθυμεί να λύσει ο χρήστης. Η εισαγωγή των σχεδιαστικών παραμέτρων (μεταβλητών) θα γίνεται σε στάδια.

Αρχικά θα εισάγεται το όνομα της παραμέτρου, επιβεβαιώνεται από το πρόγραμμα εάν το δοθέν όνομα είναι αποδεκτό και εάν δεν είναι ο χρήστης θα καλείται να εισάγει ένα νέο αποδεκτό όνομα. Μη αποδεκτά θεωρούνται τα ονόματα που έχουν ήδη δηλωθεί, ή τα ονόματα που ξεκινούν με αριθμό – σύμβολο. Όταν δοθεί αποδεκτό όνομα, το πρόγραμμα θα το αποθηκεύει και εν συνεχεία θα ζητάει τον τύπο της παραμέτρου που δηλώθηκε. Η επιλογή του

τύπου θα γίνεται από μία ήδη διαμορφωμένη λίστα η οποία περιέχει μερικά προτεινόμενα είδη μεταβλητών, πχ Real Discrete ή Real Continuous κτλ.

Εφόσον ο χρήστης επιλέξει τον τύπο της μεταβλητής, το πρόγραμμα θα ζητάει το εύρος τιμών της δηλωθείσας παραμέτρου. Εδώ λοιπόν εφαρμόζεται από το χρήστη ο σχεδιαστικός περιορισμός που αφορά την εν λόγω σχεδιαστική παράμετρο. Ο χρήστης θα δηλώνει το εύρος τιμών, εισάγοντας μια ελάχιστη και μια μέγιστη τιμή. Οι τιμές αυτές πρέπει να είναι αριθμοί και αν περιέχουν άλλους χαρακτήρες το πρόγραμμα θα ζητάει από το χρήστη να εισάγει νέες αποδεκτές τιμές.

Προς διευκόλυνση του χρήστη, μπορούν να διαμορφωθούν λίστες την ώρα που τρέχει το πρόγραμμα, με τα ονόματα των δηλωμένων σχεδιαστικών παραμέτρων καθώς και με το εύρος τιμών τους. Στην περίπτωση που ο χρήστης θέλει να αλλάξει κάποιο δεδομένο, θα μπορεί κάτω από κάθε λίστα να μπει ένα κουμπί Delete, ώστε ο χρήστης να μπορεί να σβήσει ένα όνομα ή μια τιμή και να εισάγει νέα δεδομένα.



Σχήμα 3.2. Σχεδιάγραμμα διαδικασίας εισαγωγής σχεδιαστικών παραμέτρων στο περιβάλλον

Εισαγωγή και ανάλυση υπολογιστικών σχέσεων επί των σχεδιαστικών παραμέτρων με χρήση parser

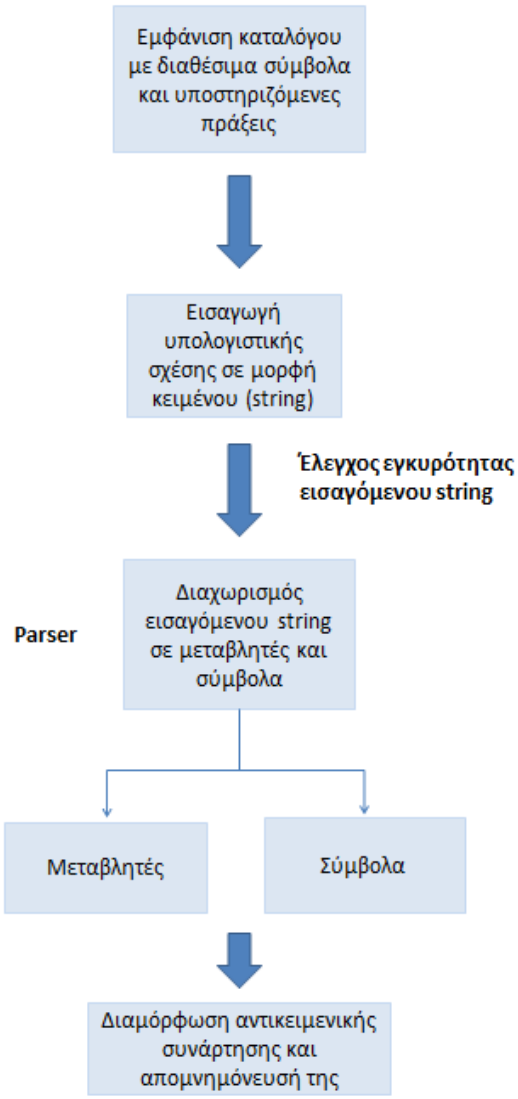
Εάν η δήλωση των σχεδιαστικών παραμέτρων και των τιμών τους έχει ολοκληρωθεί με επιτυχία, τότε ο χρήστης θα καλείται να δηλώσει την υπολογιστική σχέση που διέπει το πρόβλημα που επιθυμεί να λύσει. Εδώ εισάγεται η αντικειμενική συνάρτηση του προβλήματος προς βελτιστοποίηση, ή αλλιώς η συνάρτηση καταλληλότητας σε μορφή κειμένου (string). Ο χρήστης θα πρέπει να εξασφαλίσει στο σημείο αυτό ότι η σχέση που εισάγει περιέχει σχεδιαστικές παραμέτρους που δήλωσε στο προηγούμενο κομμάτι. Εάν εισάγει μια σχέση που περιέχει και μεταβλητές μη δηλωμένες προηγουμένως, τότε το πρόγραμμα θα εμφανίσει ένα μήνυμα ενημερώνοντας τον χρήστη ότι κάποια σχεδιαστική παράμετρος δεν έχει δηλωθεί. Στην περίπτωση αυτή, ο χρήστης θα μπορεί να εισάγει μια νέα σχέση που να περιέχει παραμέτρους ήδη δηλωμένες, ή να προσθέσει κάποια νέα σχεδιαστική παράμετρο που ίσως παρέλειψε στο προηγούμενο βήμα καθώς και το εύρος τιμών της. Εν συνεχεία μπορεί να εισάγει εκ νέου την υπολογιστική σχέση, η οποία αυτή τη φορά θα καταχωρηθεί με επιτυχία, εφόσον η άγνωστη πριν παράμετρος έγινε πια γνωστή.

Στο σημείο αυτό, τον έλεγχο της υπολογιστικής σχέσης που εισήγαγε ο χρήστης αναλαμβάνει ο parser που χρησιμοποιείται στο περιβάλλον. Λαμβάνοντας υπόψιν τα ονόματα των δηλωμένων σχεδιαστικών παραμέτρων, ο parser χωρίζει τη δοθείσα σχέση σε μεταβλητές και σύμβολα. Επειδή ο κάθε parser έχει τη δική του τεκμηρίωση και έχει επίσης και τη δική του γραμματική – σύνταξη όσον αφορά τις μαθηματικές σχέσεις που παίρνει ως είσοδο. Τα σύμβολα που υλοποιούν τις διάφορες μαθηματικές πράξεις είναι συγκεκριμένα και εάν χρησιμοποιηθεί κάποιο άλλο τότε το πρόγραμμα θα εμφανίσει ένα μήνυμα ζητώντας να χρησιμοποιηθούν τα γνωστά σε αυτό μαθηματικά σύμβολα. Στο σημείο αυτό μπορεί επίσης να εμφανίζεται ένα μήνυμα στο χρήστη με τον κατάλογο των επιτρεπτών συμβόλων για την εισαγωγή των μαθηματικών σχέσεων. Έτσι μπορεί επίσης ο χρήστης να δει τι σύμβολα υποστηρίζει το πρόγραμμα και τι πράξεις μπορεί να έχει η αντικειμενική συνάρτηση που θα εισάγει. Άλλωστε, δεν είναι βολικό να πρέπει ο χρήστης να αναζητήσει το documentation του χρησιμοποιούμενου parser για να δει τι είδους σύμβολα μπορεί να εισάγει.

Αναλύοντας την υπολογιστική σχέση για την αντικειμενική συνάρτηση που εισάγει ο χρήστης, ο parser δίνει τη δυνατότητα κάθε φορά που απαιτείται στη συνέχεια κατά τη διάρκεια

της διαδικασία της βελτιστοποίησης η χρήση της εν λόγω σχέσης, να είναι γνωστά στο πρόγραμμα τα δεδομένα για τον υπολογισμό της. Αυτό επιτρέπει τον παραμετρικό σχεδιασμό του προγράμματος και επιτρέπει την εισαγωγή οποιασδήποτε σχέσης, δεδομένου βέβαια ότι τηρεί τους κανόνες της γραμματικής και του συντακτικού του parser αλλά και ότι περιέχει ήδη δηλωμένες σχεδιαστικές παραμέτρους.

Στην περίπτωση που εξετάζει η παρούσα εργασία ο parser καταχωρεί τις μεταβλητές – σχεδιαστικές παραμέτρους $P_1,P_2,\ldots,P_n$ και «μαθαίνει» στο πρόγραμμα ότι η αντικειμενική συνάρτηση, ή αλλιώς συνάρτηση καταλληλότητας για τη μετέπειτα βελτιστοποίηση είναι η συγκεκριμένη μαθηματική σχέση και ισούται βέβαια με τη ζητούμενη ποσότητα C.



Σχήμα 3.3. Σχεδιάγραμμα διαδικασίας εισαγωγής υπολογιστικής σχέσης και ανάλυσής της από parser

Βελτιστοποίηση ως προς τις ανεξάρτητες σχεδιαστικές παραμέτρους

Στο σημείο αυτό και εφόσον έχει γίνει επιτυχώς η εισαγωγή των σχεδιαστικών παραμέτρων και της υπολογιστικής σχέσης της αντικειμενικής συνάρτησης, το πρόγραμμα οφείλει να ζητήσει από το χρήστη ορισμένα στοιχεία σχετικά με τον τρόπο που θα διεξαχθεί η βελτιστοποίηση με τη χρήση γενετικών αλγορίθμων.

Αρχικά ο χρήστης θα καλείται να επιλέξει το είδος της βελτιστοποίησης, δηλαδή εάν επιθυμεί μεγιστοποίηση ή ελαχιστοποίηση της αντικειμενικής συνάρτησης. Για το πρόβλημα της δοκού γίνεται η επιλογή για μεγιστοποίηση ή ελαχιστοποίηση της σταθεράς ελατηρίου k. Αυτό μπορεί να γίνει απλά με την προσθήκη 2 checkbox στο περιβάλλον. Ο χρήστης θα πρέπει να πατήσει σε ένα από τα 2 checkbox και προφανώς δίπλα από το ένα θα αναγράφεται η ένδειξη Ελαχιστοποίηση ενώ δίπλα από το $2^{ο}$ η ένδειξη Μεγιστοποίηση. Η επιλογή αυτή βέβαια θα σηματοδοτεί αναλόγως στον κώδικα του προγράμματος την αναζήτηση ελάχιστης ή μέγιστης τιμής της συνάρτησης καταλληλότητας.

Στη συνέχεια ο χρήστης θα πρέπει να εισάγει το μέγεθος του πληθυσμού που θα δημιουργήσει ο γενετικός αλγόριθμος στο αρχικό του στάδιο (Population Generation). Αρνητικές τιμές δε θα γίνονται αποδεκτές και ο χρήστης θα καλείται να εισάγει νέα αποδεκτή τιμή. Εδώ το πρόγραμμα δημιουργεί έναν πρώτο πληθυσμό δημιουργώντας χρωμοσώματα των οποίων οι τυχαίες τιμές καθορίζονται από το επιτρεπόμενο εύρος που έχει εισάγει ο χρήστης στη δήλωση των σχεδιαστικών παραμέτρων. Το πρόγραμμα εδώ θα πρέπει να υπολογίσει τη συνάρτηση καταλληλότητας για κάθε χρωμόσωμα και να ταξινομήσει με κριτήριο την τιμή αυτή τα χρωμοσώματα ξεκινώντας από αυτό με τη μεγαλύτερη τιμή καταλήγοντας σε εκείνο με τη μικρότερη. Ο υπολογισμός της συνάρτησης καταλληλότητας γίνεται με τη χρήση του parser,ο οποίος έχει αποθηκεύσει τη δομή της αντικειμενικής συνάρτησης και κάθε φορά παίρνει τιμές τυχαίες αλλά μέσα στα δοθέντα επιτρεπτά όρια που έχουν δοθεί στην αρχή από το χρήστη για τις σχεδιαστικές παραμέτρους

Το επόμενο δεδομένο που ζητάει το πρόγραμμα από το χρήστη είναι το ποσοστό των χρωμοσωμάτων που θα επιλεγεί να επιβιώσει ανέπαφο στην επόμενη γενιά (Selection). Αυτό μπορεί να γίνει απλά, με το χρήστη να καλείται να εισάγει ένα θετικό ακέραιο στο διάστημα (0,100). Μη αποδεκτές τιμές θα πρέπει να απορρίπτονται και σε αυτήν την περίπτωση. Το

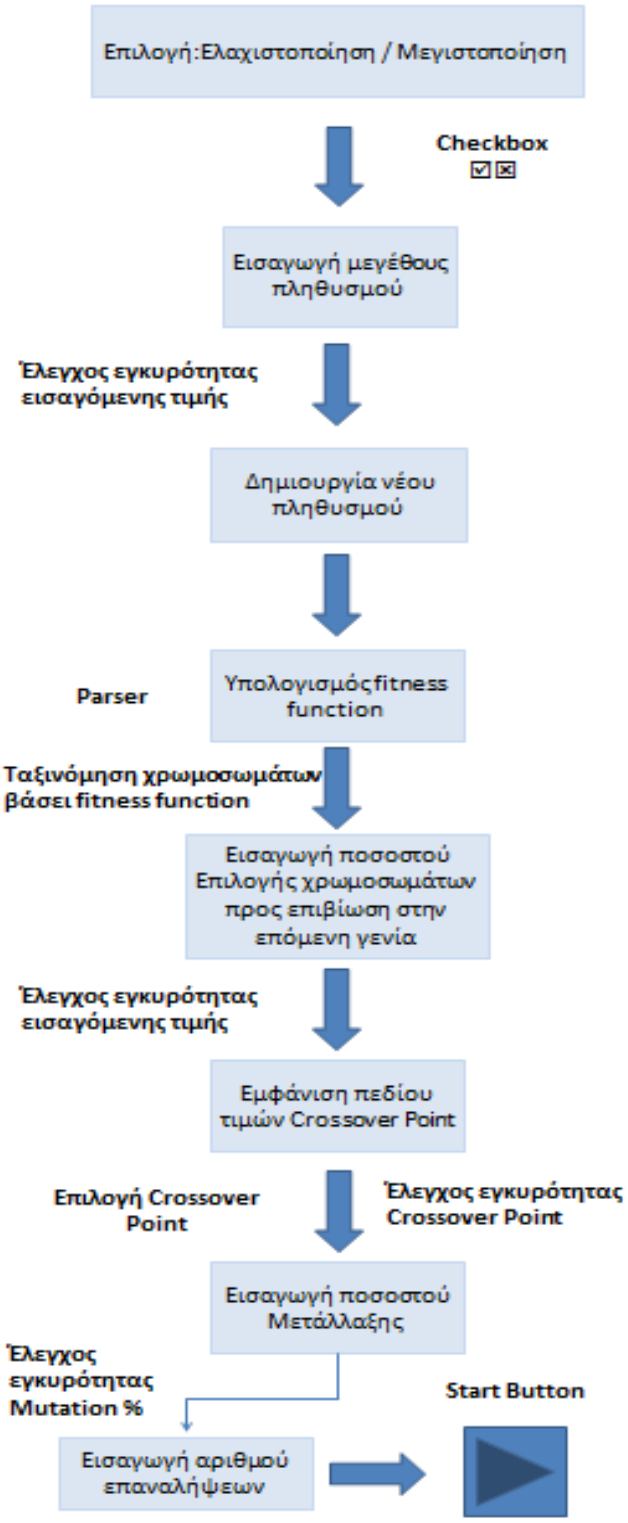
πρόγραμμα θα πρέπει σε αυτό το σημείο να καταλάβει ότι τα υπόλοιπα χρωμοσώματα θα διασταυρωθούν μεταξύ τους (Crossover).

Όσον αφορά τη διαδικασία της διασταύρωσης, ο χρήστης θα πρέπει να εισάγει ένα επιθυμητό σημείο διασταύρωσης, το οποίο βέβαια θα παίρνει τιμές εξαρτώμενες από το μήκος των χρωμοσωμάτων που έχουν διαμορφωθεί στο επίπεδο της δημιουργίας του νέου πληθυσμού. Λόγω της παραμετρικής σχεδίασης του περιβάλλοντος, το μέγεθος του χρωμοσώματος δεν είναι εξ'αρχής γνωστό. Γι'αυτό θα ήταν βολικό για το χρήστη, μετά από την εισαγωγή του επιθυμητού μεγέθους του πληθυσμού στο προηγούμενο βήμα, να εμφανίζεται ένα μήνυμα με το επιτρεπτό εύρος του σημείου διασταύρωσης που μπορεί να δηλώσει. Σε κάθε περίπτωση, τιμή που δηλώνεται και βρίσκεται εκτός του μεγέθους του χρωμοσώματος θα πρέπει να απορρίπτεται.

Για τη διαδικασία της μετάλλαξης, θα μπορεί να εισάγει ο χρήστης κι εδώ ένα ποσοστό επί των χρωμοσωμάτων προς μετάλλαξη. Ωστόσο επειδή το ποσοστό αυτό είναι σε κάθε περίπτωση πολύ μικρός αριθμός, θα ήταν βολικό να είναι ένας προκαθορισμένος μικρός αριθμός και έτσι ο χρήστης να μην ασχοληθεί με την εισαγωγή του.

Ενδεικτικά, μπορεί να προστεθεί και ένα πεδίο όπου ο χρήστης θα μπορεί να εισάγει τον επιθυμητό αριθμό επαναλήψεων που θα τρέξει ο αλγόριθμος. Παρ'όλα αυτά, θα πρέπει το πρόγραμμα να λαμβάνει υπόψη του ορισμένα κριτήρια τερματισμού του γενετικού αλγορίθμου, παραδείγματος χάρη τον τερματισμό όταν για αριθμό επαναλήψεων παραπάνω από έναν συγκεκριμένο φυσικό αριθμό, ο αλγόριθμος δε δίνει καλύτερη τιμή συνάρτησης καταλληλότητας.

Στο τέλος του περιβάλλοντος, θα πρέπει να μπει ένα κουμπί με το πάτημα του οποίου ξεκινάει η διαδικασία της βελτιστοποίησης. Στο παρακάτω σχήμα εξηγείται η γενική ιδέα της δομής του interface στο κομμάτι που αφορά τη βελτιστοποίηση και εξηγούνται μερικές λειτουργίες που θα εκτελούνται όταν ο χρήστης εισάγει τα ζητούμενα στοιχεία.



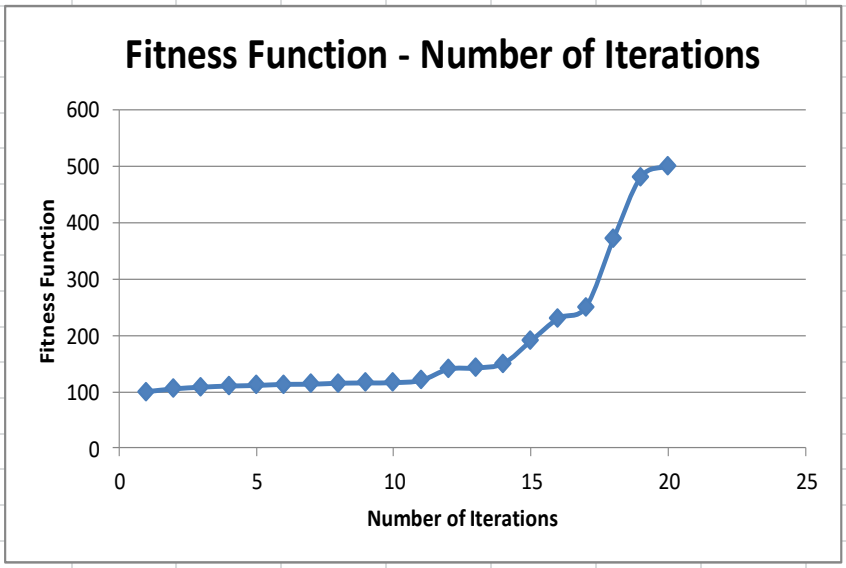
Σχήμα 3.4. Σχεδιάγραμμα διαδικασίας βελτιστοποίησης

Παρουσίαση Αποτελεσμάτων

Πατώντας το κουμπί για την εκκίνηση της διαδικασίας της βελτιστοποίησης, το πρόγραμμα θα πρέπει να εμφανίζει ένα μήνυμα για το εάν η διαδικασία τερματίστηκε επιτυχώς. Στην περίπτωση που ο αλγόριθμος τρέξει επιτυχώς, το περιβάλλον θα πρέπει να παρουσιάζει τα αποτελέσματα του αλγορίθμου στο χρήστη με κάποιο τρόπο. Είναι πολύ απλό και βολικό η παρουσίαση να γίνει με τη δημιουργία από το πρόγραμμα ενός αρχείου κειμένου, στο οποίο θα έχουν αποθηκευτεί με τη σωστή σημειολογία οι μέγιστες τιμές συνάρτησης καταλληλότητας ανά επανάληψη και δίπλα η σειριακή αρίθμηση των επαναλήψεων. Το αρχείο αυτό κειμένου μπορεί να δημιουργείται στην επιφάνεια εργασίας του χρήστη, και όταν αυτός το ανοίγει με το Microsoft Office Excel, να εμφανίζονται το διάγραμμα που περιεγράφηκε, ώστε να δει την πορεία του αλγορίθμου και εν τέλει την τιμή στην οποία συγκλίνει αυτός, η οποία είναι και η ζητούμενη λύση του προβλήματος βελτιστοποίησης.

Παρακάτω φαίνεται ένα παράδειγμα απεικόνισης αποτελεσμάτων με τυχαίες τιμές συνάρτησης καταλληλότητας και για 20 επαναλήψεις.

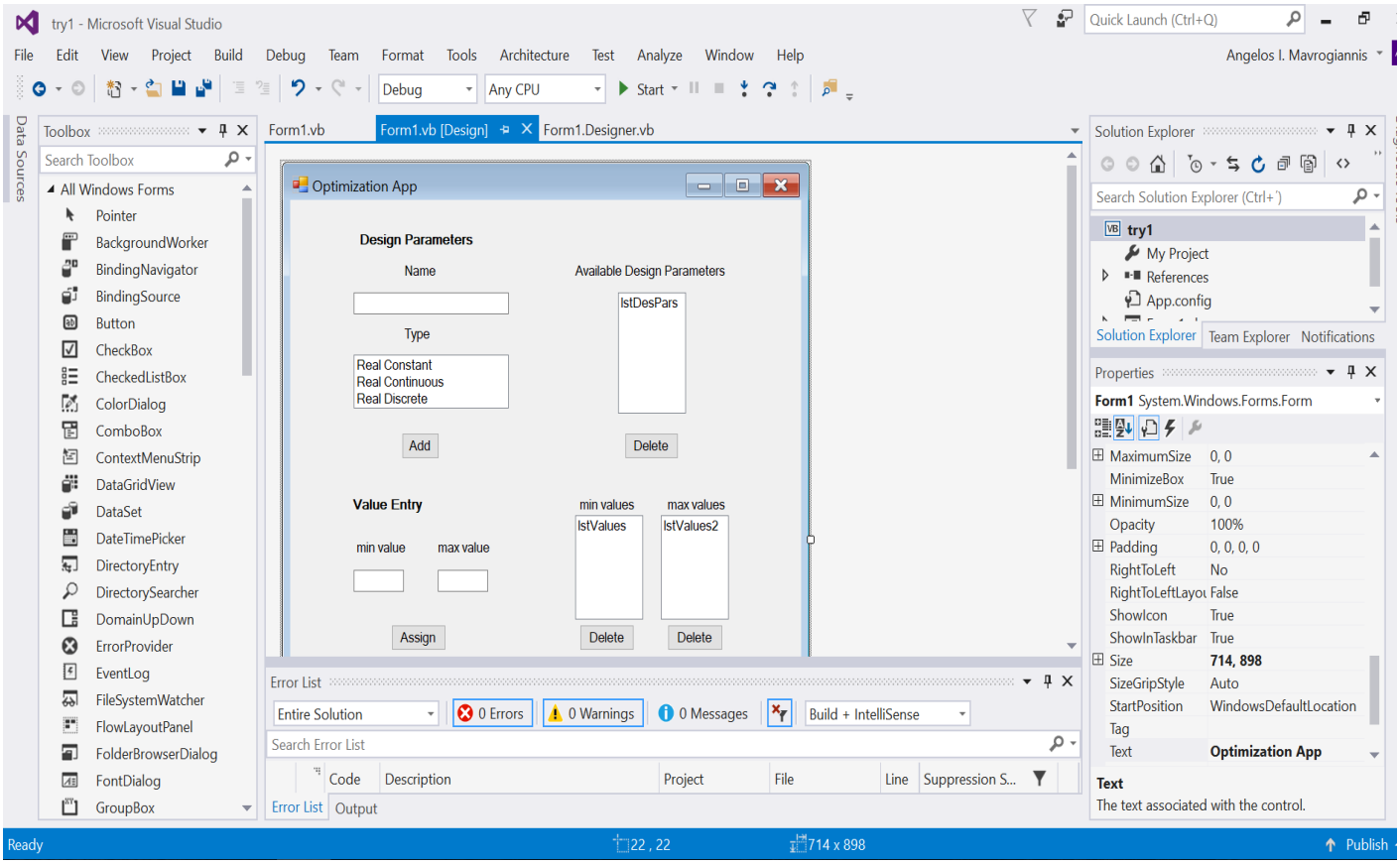
Number of iterations	Maximum Fitness Function
1	100
2	105
3	108
4	110
5	111
6	113
7	113.2
8	115
9	115.5
10	116
11	120
12	140
13	142
14	150
15	190
16	230
17	250
18	370
19	480
20	500



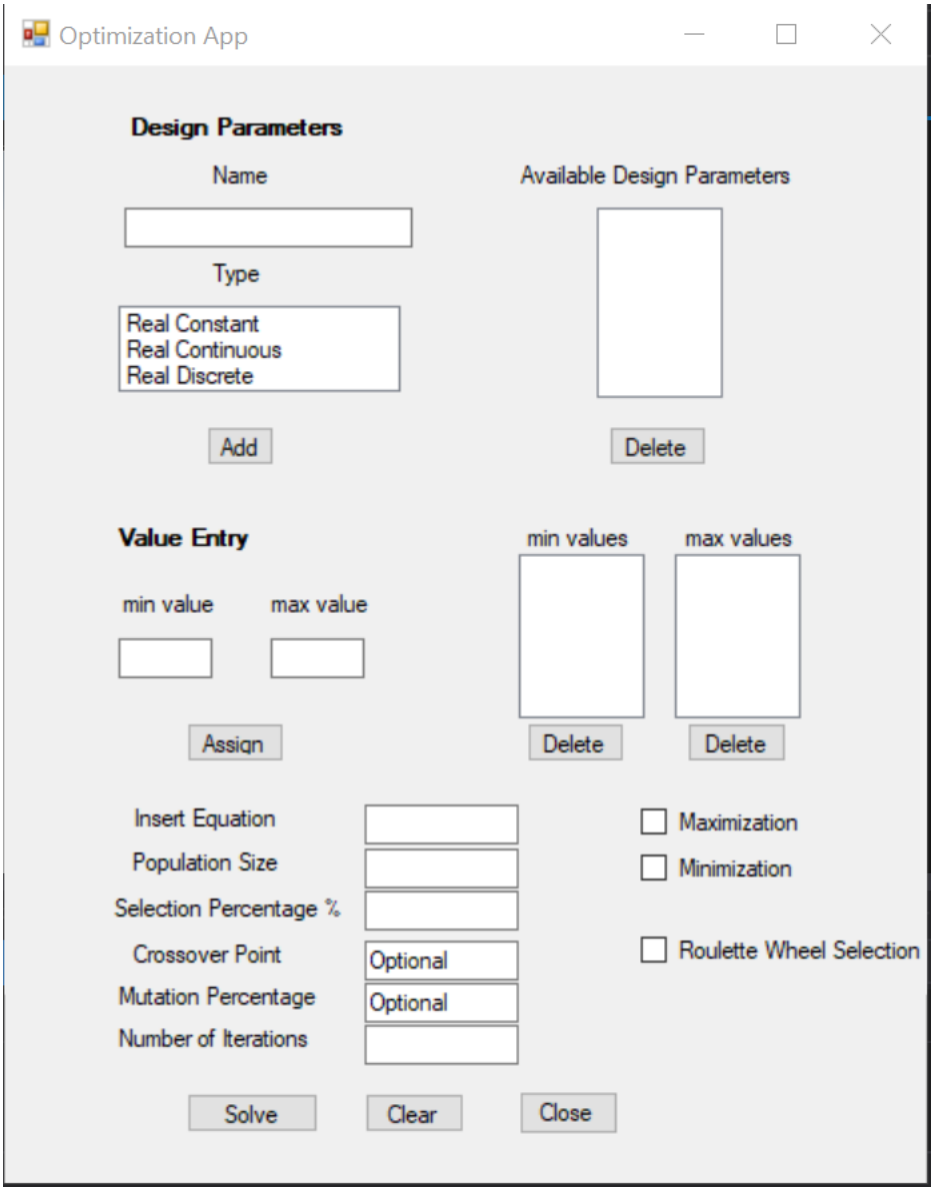
Σχήμα 3.5. Ενδεικτικός τρόπος απεικόνισης αποτελεσμάτων με χρήση του Microsoft Office Excel

ΚΕΦΑΛΑΙΟ 3 ΠΕΡΙΓΡΑΦΗ ΤΟΥ ΠΕΡΙΒΑΛΛΟΝΤΟΣ

Η ανάπτυξη του περιβάλλοντος έγινε στο Microsoft Visual Studio Enterprise και ο parser που χρησιμοποιήθηκε για την εισαγωγή και ανάλυση των υπολογιστικών σχέσεων είναι ο Mathparser.NET [22]. Παρατίθεται εδώ το γραφικό περιβάλλον εργασίας στο Visual Studio καθώς και η γενική εικόνα του περιβάλλοντος που αναπτύχθηκε.



Σχήμα 4.1. Το γραφικό περιβάλλον εργασίας του Visual Studio Enterprise



Σχήμα 4.2. Το δημιουργηθέν περιβάλλον ως σύνολο

Στις επόμενες ενότητες αυτού του κεφαλαίου θα εξηγηθεί η λειτουργία του περιβάλλοντος και θα εφαρμοστούν παραδείγματα χρήσης του σε μηχανολογικές εφαρμογές.

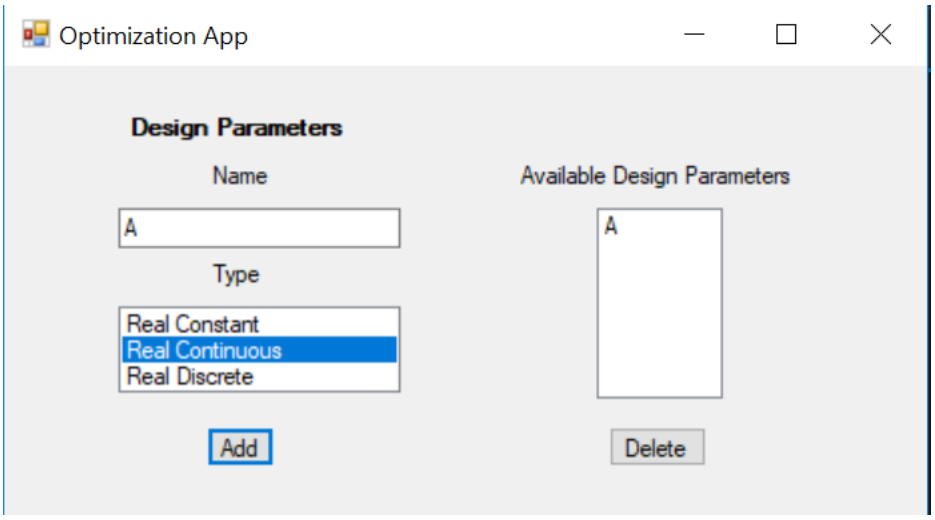
4.1 ΔΗΛΩΣΗ ΣΧΕΔΙΑΣΤΙΚΩΝ ΠΑΡΑΜΕΤΡΩΝ ΚΑΙ ΤΙΜΩΝ

Αρχικά ο χρήστης καλείται να δηλώσει τις σχεδιαστικές παραμέτρους. Πρώτα πρέπει να δηλωθεί το όνομα της σχεδιαστικής παραμέτρου στο σχετικό πεδίο και ύστερα να επιλεχθεί ο τύπος της παραμέτρου από μία λίστα με διαθέσιμους τους εξής τύπους:

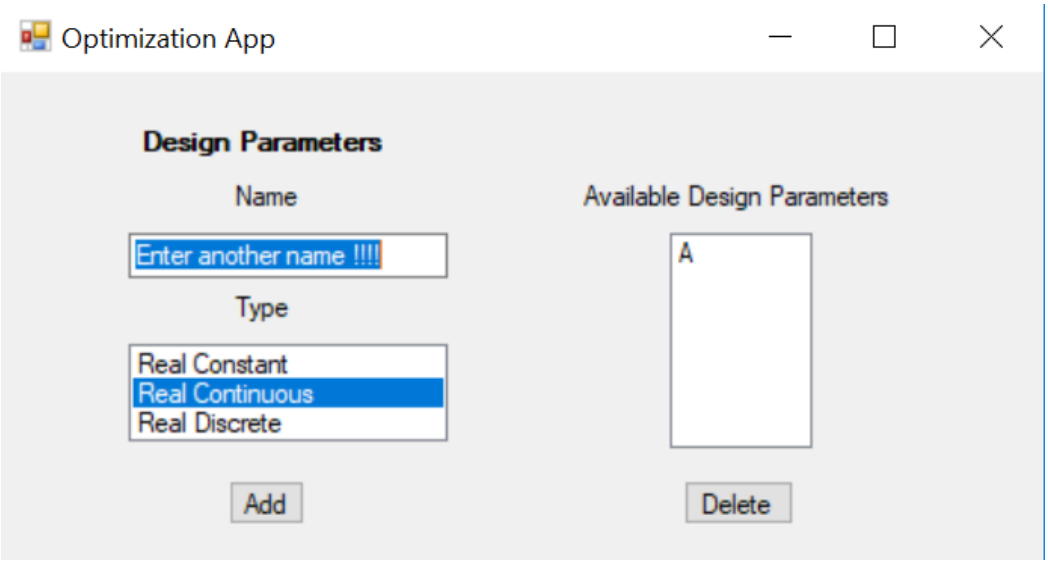
- Πραγματική Σταθερά (Real Constant)
- Πραγματική Συνεχής (Real Continuous)
- Πραγματική Διακριτή (Real Discrete)

Εφόσον το όνομα που εισήχθη είναι αποδεκτό και δεν έχει δηλωθεί ξανά προηγουμένως, πατώντας το κουμπί “Add” η παράμετρος και το είδος της καταχωρούνται στο πρόγραμμα και στον parser. Προς διευκόλυνση του χρήστη, κάθε επιτυχώς δηλωμένη παράμετρος εμφανίζεται σε μία λίστα στο επάνω δεξιά κομμάτι του περιβάλλοντος, με ονομασία “Available Design Parameters”. Εάν ο χρήστης επιθυμεί να σβήσει κάποια ήδη υπάρχουσα σχεδιαστική παράμετρο, τότε την επιλέγει από τη λίστα αυτή πατώντας επάνω στο όνομα που επιθυμεί και ύστερα πατάει το κουμπί “Delete” που βρίσκεται κάτω από τη λίστα.

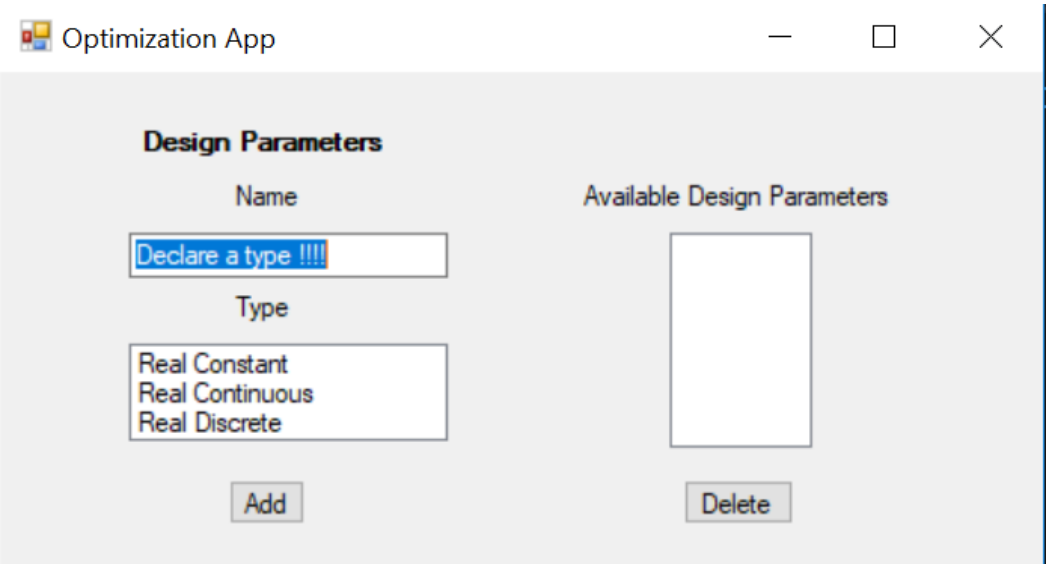
Ακολουθούν εικόνες από δοκιμές εισαγωγής παραμέτρων και παραδείγματα από λανθασμένη εισαγωγή. Όταν γίνεται λανθασμένη εισαγωγή το πρόγραμμα εμφανίζει μηνύματα που βοηθούν το χρήστη να καταλάβει τι πρέπει να διορθώσει ώστε να γίνει σωστά η δήλωση των παραμέτρων.



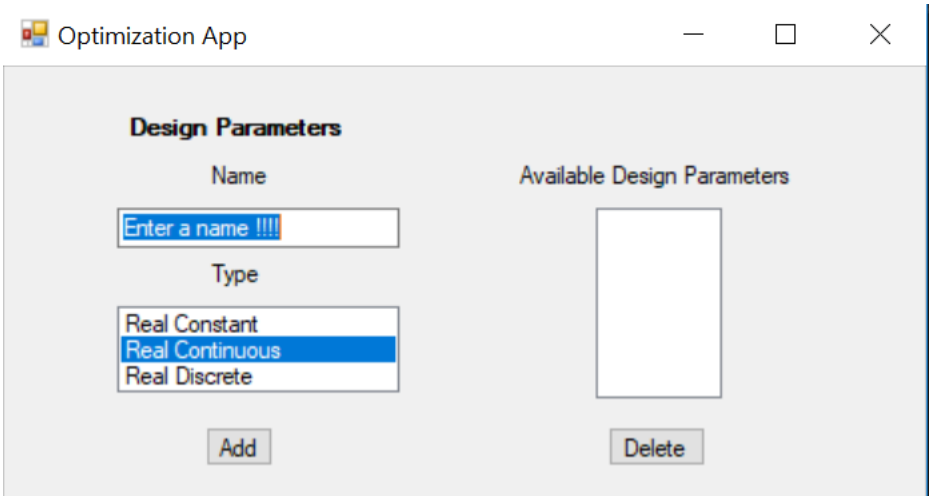
Σχήμα 4.3. Επιτυχής δήλωση παραμέτρου με όνομα A



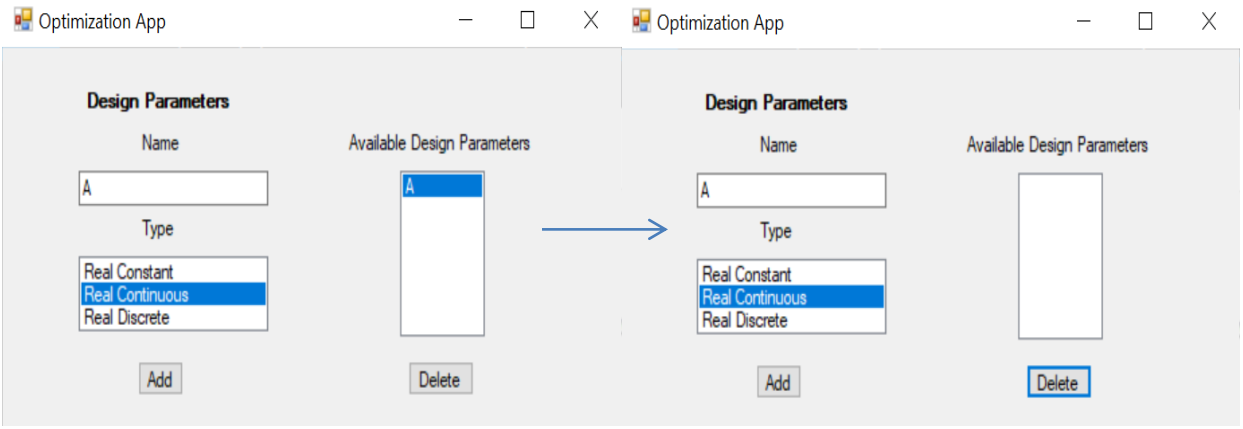
Σχήμα 4.4. Ανεπιτυχής δήλωση παραμέτρου με όνομα ήδη δηλωθέν προηγουμένως



Σχήμα 4.5. Προσπάθεια δήλωσης παραμέτρου χωρίς να έχει επιλεχτεί ο τύπος της

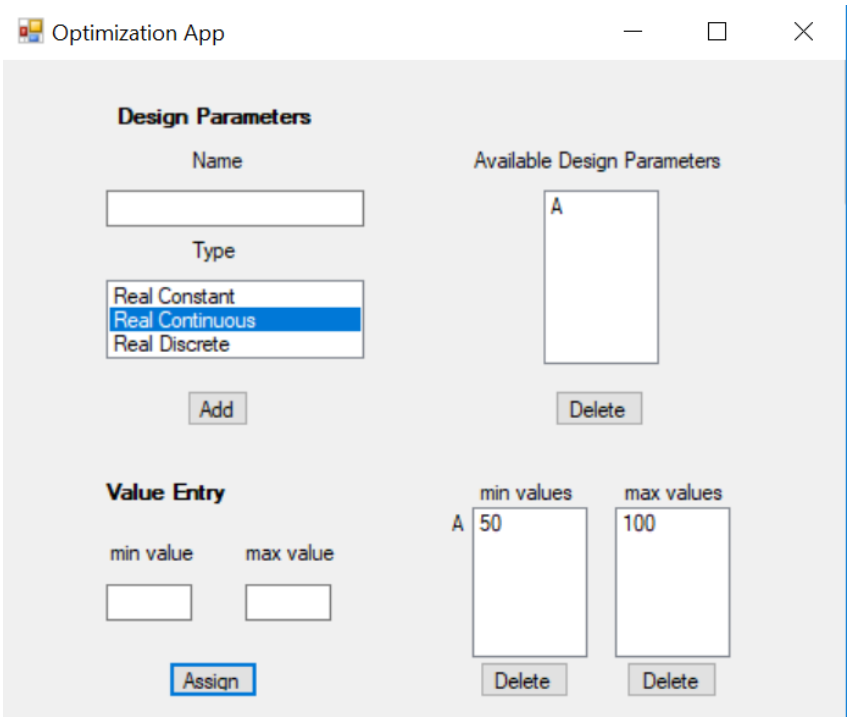


Σχήμα 4.6. Προσπάθεια δήλωσης παραμέτρου χωρίς να έχει οριστεί το όνομά της

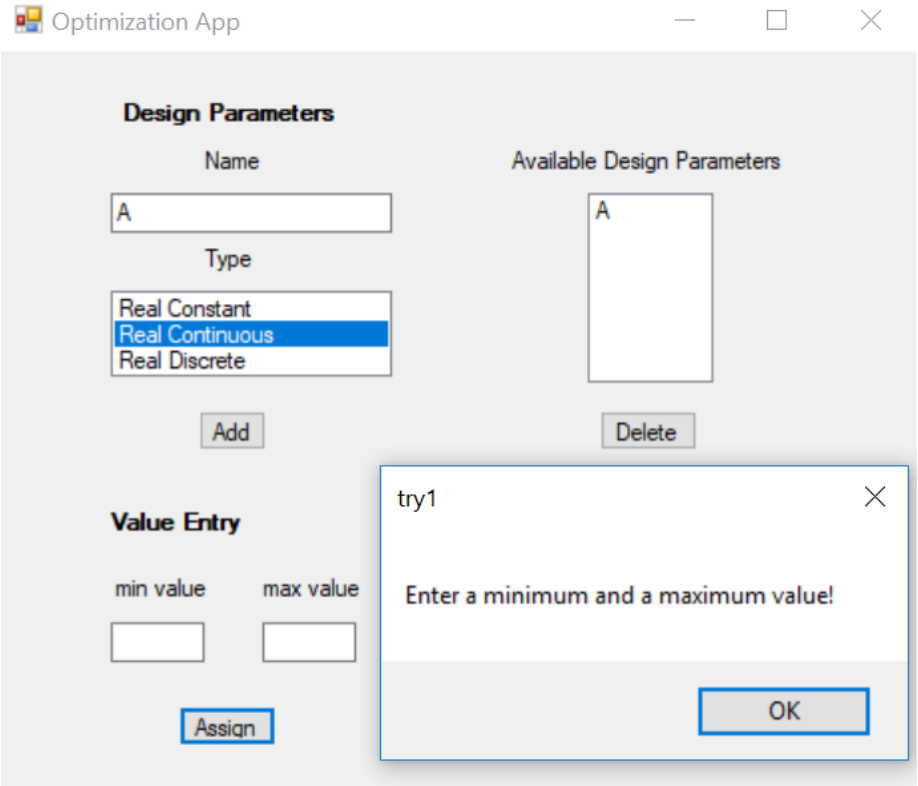


Σχήμα 4.7. Διαγραφή της δηλωμένης μεταβήτής A πατώντας το πλήκτρο Delete

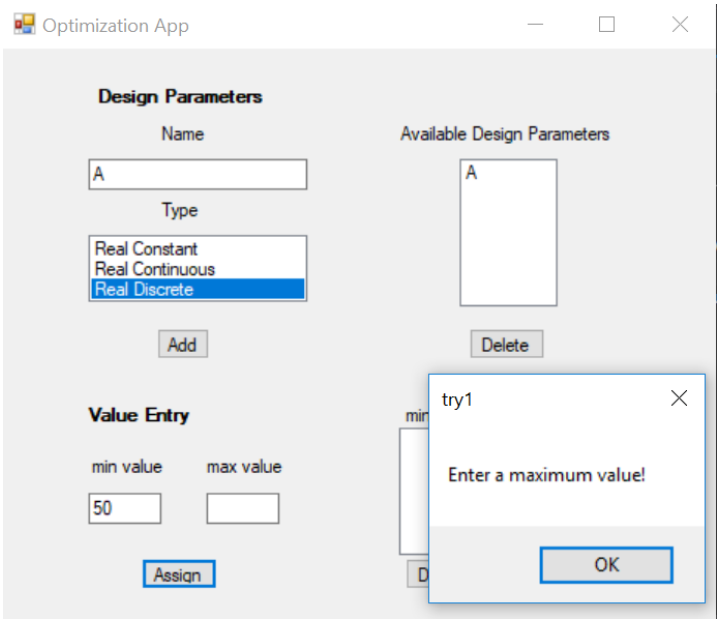
Μετά από την επιτυχημένη δήλωση του ονόματος μίας παραμέτρου, ο χρήστης θα πρέπει να εισάγει το εύρος τιμών της παραμέτρου στο πεδίο κάτω από το κουμπί “Add” με όνομα “Value Entry”. Στο πεδίο με την ένδειξη “min value” ο χρήστης πρέπει να εισάγει την ελάχιστη τιμή της παραμέτρου, ενώ στο πεδίο με την ένδειξη “max value” τη μέγιστη. Πατώντας το κουμπί “Assign”, οι δηλωμένες τιμές καταχωρούνται στο πρόγραμμα και εμφανίζονται στις λίστες με όνομα “min values” και “max values” αντίστοιχα, ενώ δίπλα στις τιμές εμφανίζεται και το όνομα της παραμέτρου στην οποία αυτές αντιστοιχούν.



Σχήμα 4.8. Επιτυχής δήλωση εύρους τιμών μεταβλητής Α



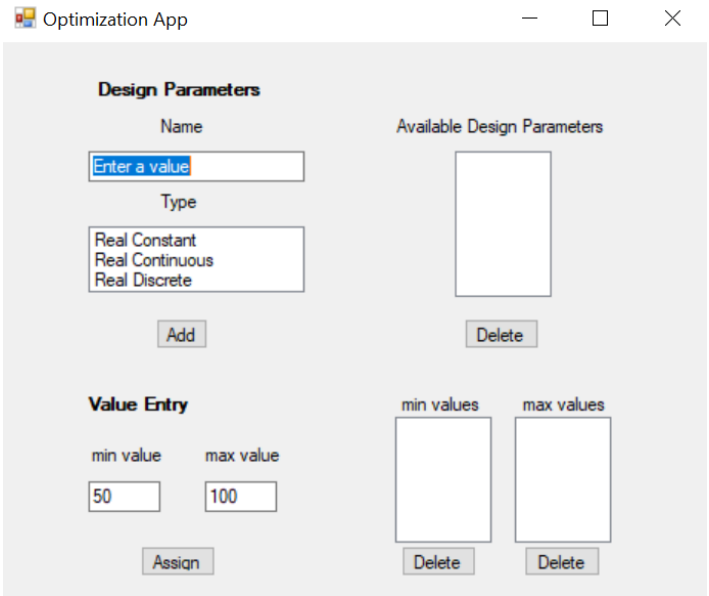
Σχήμα 4.9. Πάτημα κουμπιού Assign χωρίς να έχουν εισαχθεί min και max τιμές



Σχήμα 4.10. Πάτημα κουμπιού Assign χωρίς να έχει εισαχθεί η max τιμή

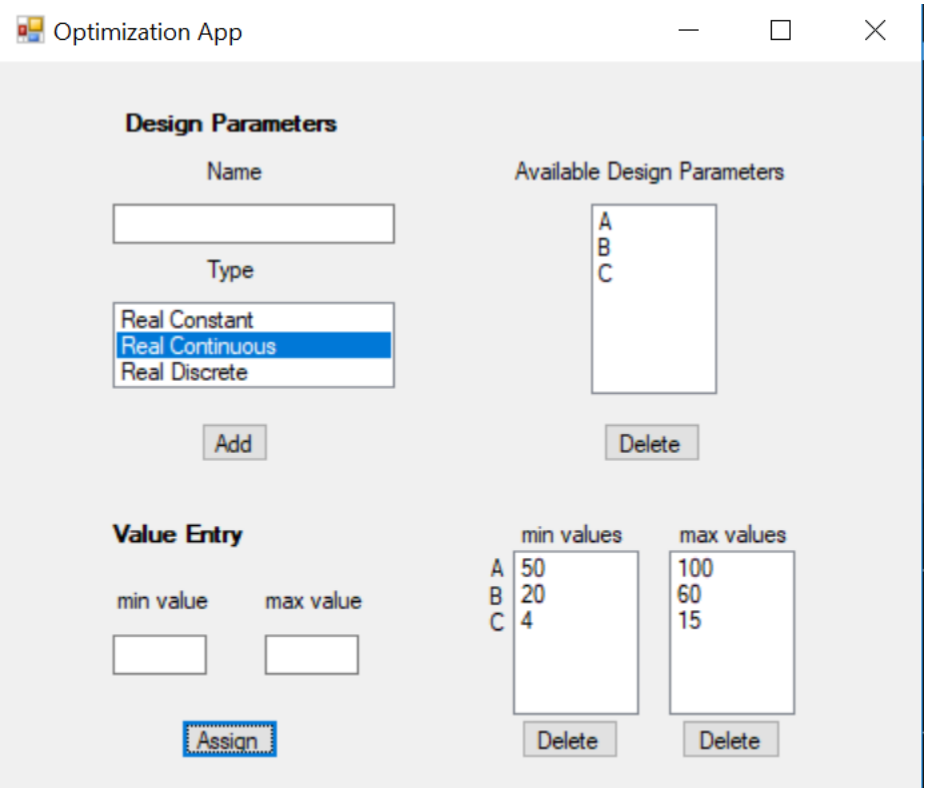
Εάν έχει γίνει επιτυχώς η δήλωση των τιμών των παραμέτρων και ο χρήστης θέλει να σβήσει κάποια μη επιθυμητή τιμή μπορεί να το κάνει πατώντας το πλήκτρο “Delete” κάτω από την αντίστοιχη λίστα τιμών.

Επίσης, εάν ο χρήστης προσπαθήσει να δηλώσει τιμές μη έχοντας πρώτα δηλώσει όνομα παραμέτρου, εμφανίζεται μήνυμα που του ζητάει να εισάγει όνομα παραμέτρου.



Σχήμα 4.11. Πάτημα κουμπιού Assign χωρίς να έχει δηλωθεί κάποια παράμετρος

Ο χρήστης μπορεί να επαληθεύσει εάν έχει δηλώσει τα ονόματα και τα εύρη τιμών των σχεδιαστικών παραμέτρων που επιθυμεί ελέγχοντας τις λίστες “Available Design Parameters” και “min values”, “max values”, οι οποίες παρουσιάζουν τις καταχωρημένες παραμέτρους και τις τιμές τους.



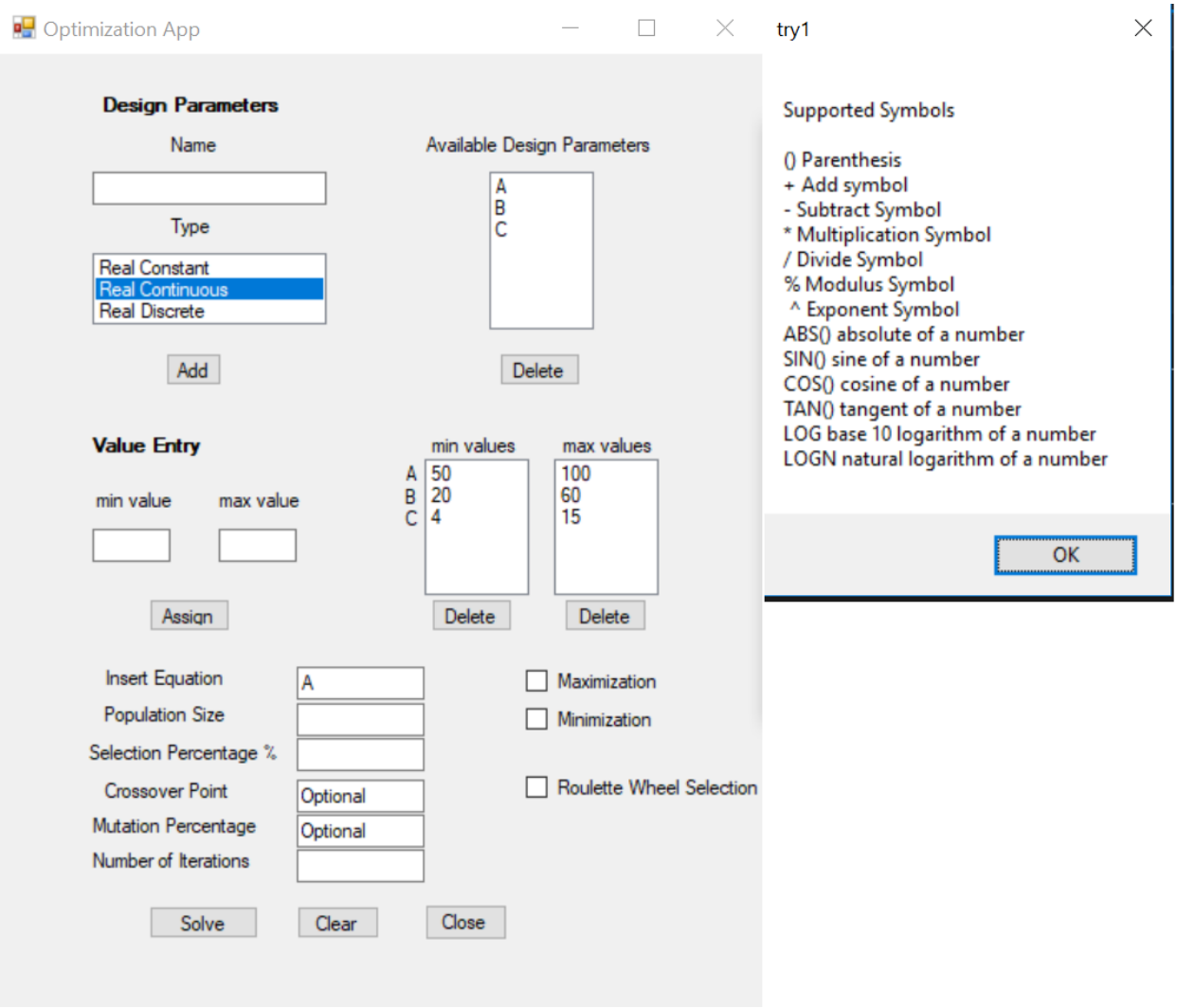
Σχήμα 4.12. Παράδειγμα επιτυχημένης δήλωσης τριών σχεδιαστικών παραμέτρων και των τιμών τους

4.2 ΕΙΣΑΓΩΓΗ ΚΑΙ ΑΝΑΛΥΣΗ ΥΠΟΛΟΓΙΣΤΙΚΩΝ ΣΧΕΣΕΩΝ

Εφόσον ο χρήστης έχει δηλώσει με επιτυχία τα ονόματα και τα εύρη τιμών των σχεδιαστικών παραμέτρων που επιθυμεί, καλείται να εισάγει την εξίσωση – αντικειμενική συνάρτηση – συνάρτηση καταλληλότητας που περιγράφει το πρόβλημα που επιθυμεί να λύσει. Η εξίσωση αυτή θα πρέπει να αποτελείται από τις δηλωμένες σχεδιαστικές παραμέτρους και από σύμβολα

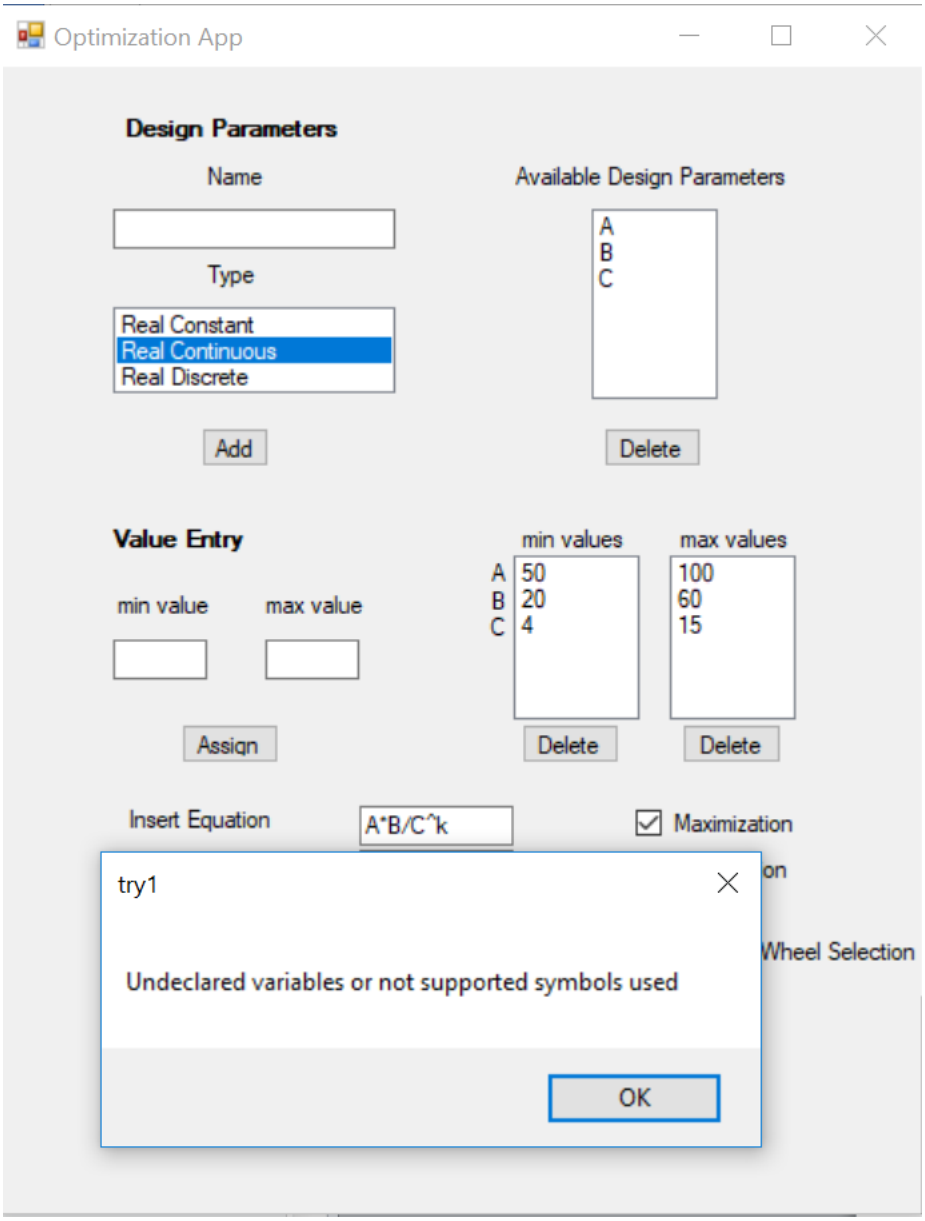
μαθηματικών πράξεων που υποστηρίζει ο χρησιμοποιούμενος parser, αφού οι απαραίτητες πράξεις και διαδικασίες που θα γίνουν στη διαδικασία βελτιστοποίησης θα υλοποιηθούν με τη χρήση του parser.

Επειδή ο κάθε parser υποστηρίζει τις δικές του πράξεις και τα δικά του σύμβολα προς διεκπεραίωση αυτών, μόλις ο χρήστης πληκτρολογήσει κάτι στο πεδίο εισαγωγής της εξίσωσης “Insert Equation”, εμφανίζεται στην οθόνη ένας κατάλογος με τα υποστηριζόμενα μαθηματικά σύμβολα.



Σχήμα 4.13. Εμφάνιση καταλόγου υποστηριζόμενων συμβόλων και πράξεων μόλις ο χρήστης πληκτρολογήσει στο πεδίο της εισαγωγής της εξίσωσης

Εάν ο χρήστης εισάγει εξίσωση που περιέχει κάποια μεταβλητή που δεν έχει δηλωθεί παραπάνω ή κάποιο μη υποστηριζόμενο μαθηματικό σύμβολο, τότε εμφανίζεται σχετικό μήνυμα και η διαδικασία σταματάει.



Σχήμα 4.14. Εισαγωγή μη δηλωμένης παραμέτρου στην εξίσωση και εμφάνιση ανάλογου μηνύματος

Ο parser χρειάζεται διότι ο χρήστης πληκτρολογεί και εισάγει την εξίσωση σε μορφή χαρακτήρων (string), δηλαδή για παράδειγμα:

" A*B/C"

Παρουσιάζεται λοιπόν η ανάγκη αποκωδικοποίησης της συμβολοσειράς που εισάγει ο χρήστης, με την έννοια του να ξεχωρίσει το πρόγραμμα ποιες είναι οι μεταβλητές και ποια είναι τα σύμβολα. Για το λόγο αυτό, στο προηγούμενο βήμα όπου δηλώνονται οι μεταβλητές έχει γίνει καταχώριση από τον parser του ονόματος της κάθε μεταβλητής, ώστε να μπορεί να γίνει η αποκωδικοποίηση στην εισαγόμενη συμβολοσειρά. Η λειτουργία λοιπόν και η χρησιμότητα του parser έγκειται στη «μετάφραση» της αλφαριθμητικής σχέσης που εισάγει ο χρήστης σε μαθηματική εξίσωση με γνωστές μεταβλητές και τιμές στο πρόγραμμα.

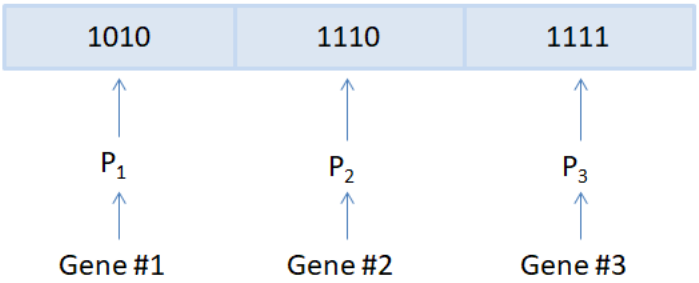
4.3 ΔΙΑΔΙΚΑΣΙΑ ΒΕΛΤΙΣΤΟΠΟΙΗΣΗΣ

Το επόμενο κομμάτι της εφαρμογής σχετίζεται καθαρά με το γενετικό αλγόριθμο βελτιστοποίησης και έχει να κάνει με τη *δημιουργία ενός αρχικού πληθυσμού* (Initial Population Generation), την επιλογή των καλύτερων μελών του πληθυσμού αυτού, τη διασταύρωση των υπόλοιπων μελών μεταξύ τους, τη μετάλλαξη (Mutation) κάποιων μελών εκ του ολικού πληθυσμού και εν τέλει την επανάληψη της διαδικασίας αυτής μέχρις ότου ο αλγόριθμος δώσει κάποια επιθυμητή τιμή ή για συγκεκριμένο αριθμό επαναλήψεων.

Πιο συγκεκριμένα, ο χρήστης καλείται να εισάγει το μέγεθος του πληθυσμού που θα δημιουργηθεί στο σχετικό πεδίο δίπλα στην ένδειξη Population Size. Η τιμή αυτή πρέπει να είναι ένας ακέραιος αριθμός. Το πρόγραμμα ακολούθως, χρησιμοποιώντας γεννήτρια τυχαίων αριθμών, δημιουργεί τριάδες τιμών των μεταβλητών που έχουν δηλωθεί παραπάνω. Οι τιμές τους είναι προφανώς μέσα στο πεδίο τιμών που ορίζουν οι μέγιστες και ελάχιστες τιμές που έχουν επίσης δηλωθεί. Οι τριάδες αυτές, ο αριθμός των οποίων είναι ο ακέραιος που εισάγει ο χρήστης ως μέγεθος πληθυσμού, αναπαρίστανται στο πρόγραμμα με δυαδική μορφή και παίρνουν έτσι τη μορφή ενός χρωμοσώματος. Το χρωμόσωμα συνίσταται από 3 *«γονιδιώματα»* (genes), τα οποία είναι οι τιμές των δηλωμένων μεταβλητών, αναπαριστώμενες στο δυαδικό

σύστημα και βέβαια με σεβασμό στην ακολουθιακή τους σειρά. Δηλαδή πχ. για P_1, P_2, P_3 σχεδιαστικές παραμέτρους έχει αποφασιστεί εξ'αρχής το κάθε χρωμόσωμα να αποτελείται από:

- Gene #1 : τιμή της μεταβλητής P_1
- Gene#2 : τιμή της μεταβλητής P_2
- Gene#3 : τιμή της μεταβλητής P_3



Σχήμα 4.15. Παράδειγμα χρωμοσώματος και γονιδιωμάτων

Να σημειωθεί εδώ ότι οι τιμές που αναγράφονται στο σχήμα χρησιμοποιούνται ως παράδειγμα και δεν έχουν εξαχθεί από το πρόγραμμα. Εξάλλου το μέγεθος του κάθε γονιδιώματος, του bitstring δηλαδη που το αναπαριστά, εξαρτάται από τις μέγιστες τιμές που δηλώνει ο χρήστης για κάθε μεταβλητή, καθώς και από την επιθυμητή ακρίβεια σε δεκαδικά ψηφία.

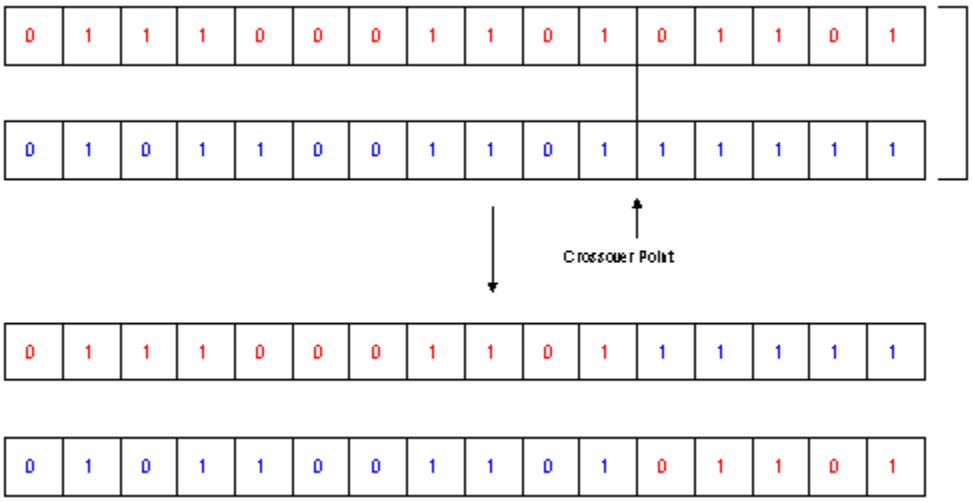
Αφού λοιπόν έχει γίνει η δημιουργία του αρχικού πληθυσμού, το πρόγραμμα υπολογίζει τη συνάρτηση καταλληλότητας για κάθε τριάδα τιμών των δηλωμένων σχεδιαστικών παραμέτρων, και καταχωρεί τα αποτελέσματα ταξινομημένα από ελάχιστη σε μέγιστη τιμή. Για την ακρίβεια η ταξινόμηση αυτή έγινε με την απλή μέθοδο γνωστή ως «bubble sort».

Εν συνεχεία, ο χρήστης καλείται να εισάγει το ποσοστό του πληθυσμού που θα περάσει στην επόμενη «γενιά» λύσεων. Ο αριθμός αυτός εισάγεται στο πεδίο Selection Percentage % και

αναμένεται ακέραιος αριθμός, τον οποίο το πρόγραμμα μετατρέπει σε ακριβή αριθμό των μελών του πληθυσμού και αναλόγως υπολογίζει τον αριθμό των λύσεων – χρωμοσωμάτων τα οποία θα μεταβούν ανέπαφα στην επόμενη γενιά του αλγορίθμου. Η διαδικασία της επιλογής των λύσεων μπορεί να γίνει είτε επιλέγοντας ένα ποσοστό των χρωμοσωμάτων με τις μέγιστες τιμές συνάρτησης καταλληλότητας ή με τη μέθοδο Επιλογής Ρουλέτας. Ο χρήστης μπορεί να πατήσει πάνω στο checkbox με την ένδειξη “Roulette Wheel Selection” ώστε να επιλέξει τη μέθοδο αυτή. Διαφορετικά η επιλογή γίνεται με την απλή μέθοδο που προαναφέρθηκε. Σύμφωνα με την Επιλογή Ρουλέτας, οι ταξινομημένες από ελάχιστη σε μέγιστη τιμές της συνάρτησης καταλληλότητας των λύσεων που παρέχει ο πληθυσμός κανονικοποιούνται. Αυτό σημαίνει ότι η κάθε υπολογισθείσα τιμή της συνάρτησης καταλληλότητας διαιρείται με το σύνολο – άθροισμα των τιμών της συνάρτησης καταλληλότητας όλων των λύσεων, έτσι ώστε το άθροισμα όλων των κανονικοποιημένων τιμών να ισούται με τη μονάδα. Παράλληλα ταξινομούνται οι κανονικοποιημένες αυτές τιμές από το μεγαλύτερο στο μικρότερο. Στο σημείο αυτό υπολογίζονται οι «συσσωρευμένες» τιμές της συνάρτησης καταλληλότητας. Συσσωρευμένη τιμή είναι η τιμή της κανονικοποιημένης συνάρτησης καταλληλότητας μιας λύσης, προστιθέμενη στο άθροισμα όλων των προηγουμένων κανονικοποιημένων τιμών. Η διαδικασία αυτή ελέγχεται ως προς την εγκυρότητά της παρατηρώντας τη συσσωρευμένη τιμή της τελευταίας και μέγιστης τιμής, η οποία πρέπει να ισούται με τη μονάδα. Εφόσον η ανωτέρω διαδικασία έχει γίνει σωστά, επιλέγεται ένας τυχαίος αριθμός R από το πρόγραμμα, με πεδίο τιμών το [0,1]. Η επιλεχθείσα λύση προκύπτει από μία επαναληπτική διαδικασία. Ξεκινώντας από τη μεγαλύτερη συσσωρευμένη τιμή και κατεβαίνοντας προς τις μικρότερες, η λύση της οποίας η συσσωρευμένη τιμή είναι η πρώτη μικρότερη αλγεβρικά τιμή από τον τυχαίο αριθμό R είναι και η επιλεχθείσα λύση. Η επαναληπτική αυτή διαδικασία γίνεται τόσες φορές όσες ορίζει ο χρήστης, εφόσον έχει δηλώσει ποσοστό επιλογής (Selection Percentage %), κ έτσι επιλέγεται συγκεκριμένος αριθμός λύσεων. Πρέπει να σημειωθεί επίσης εδώ, ότι σε κάθε επανάληψη ο τυχαίος αριθμός R είναι διαφορετικός, όπως ευκόλως εννοείται.

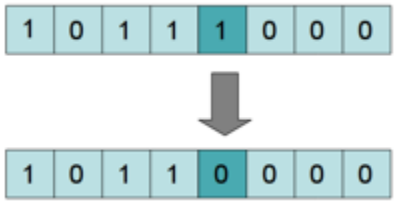
Σειρά έχει η διαδικασία της διασταύρωσης, για την οποία ο χρήστης μπορεί να εισάγει ένα συγκεκριμένο σημείο διασταύρωσης, αν και είναι προτιμότερο το σημείο διασταύρωσης να είναι κάθε φορά τυχαίο και έτσι να μην επιλέγεται από το χρήστη. Γι’αυτό, στο πεδίο εισαγωγής του σημείου διασταύρωσης αναγράφεται η λέξη “Optional”, καθιστώντας την συμπλήρωση αυτού του πεδίου προαιρετική. Στη συγκεκριμένη εφαρμογή γίνεται διασταύρωση με ένα σημείο

(Single – Point Crossover). Για τις εναπομείναντες λύσεις, δηλαδή για τις μη επιλεχθείσες, επιλέγεται ένα σημείο μεταξύ του 1 και του μεγέθους του bitstring που περιγράφει το χρωμόσωμα, και πέραν αυτού του σημείου, οι λύσεις ανά 2 ανταλλάσσουν τα 0 και 1 τους, δημιουργώντας έτσι «απογόνους», νέα χρωμοσώματα διαφορετικά από πριν.



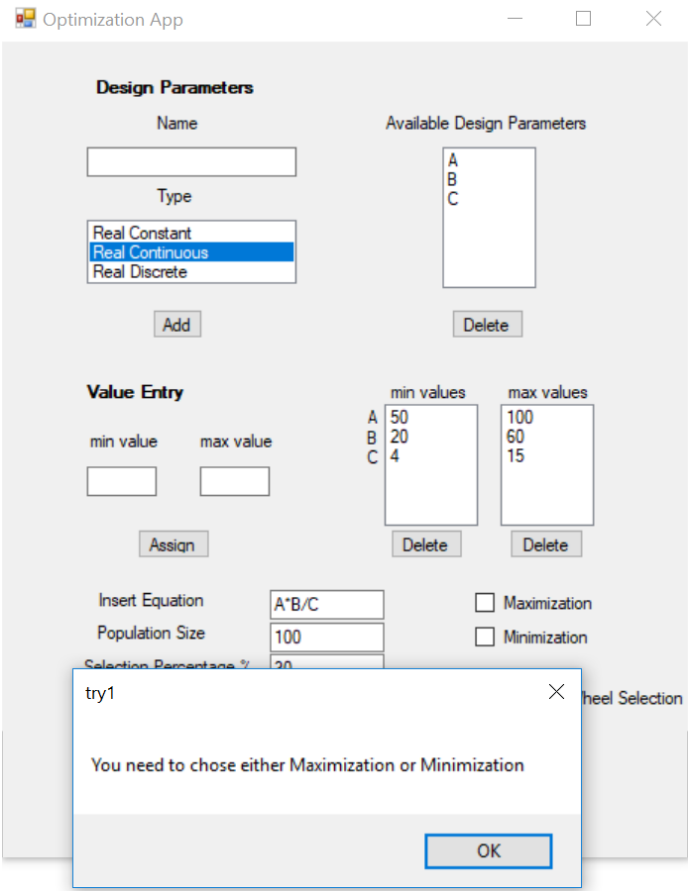
Σχήμα 4.16. Σχηματική αναπαράσταση της διαδικασίας της Διασταύρωσης (Crossover)

Στη συνέχεια εφαρμόζεται η διαδικασία της μετάλλαξης σε ένα πολύ μικρό ποσοστό του πληθυσμού. Η μετάλλαξη γίνεται με την επιλογή ενός σημείου μετάλλαξης (Single Point Bitstring Mutation). Ο χρήστης μπορεί να επιλέξει το ποσοστό μετάλλαξης προαιρετικά, διαφορετικά το πρόγραμμα επιλέγει αυτόνομα ένα πολύ μικρό ποσοστό. Στο πεδίο εισαγωγής του ποσοστού μετάλλαξης “Mutation Percentage” αναγράφεται η ένδειξη “Optional”. Το πρόγραμμα επιλέγει τυχαία ένα μικρό αριθμό χρωμοσωμάτων από το συνολικό πληθυσμό και με τυχαία διαδικασία πάλι επιλέγει ένα ψηφίο από το επιλεγμένο χρωμόσωμα. Το συγκεκριμένο ψηφίο μετατρέπεται από 0 σε 1 ή από 1 σε 0, και έτσι διατηρείται η διαφορετικότητα από τη μία γενιά στην επόμενη.



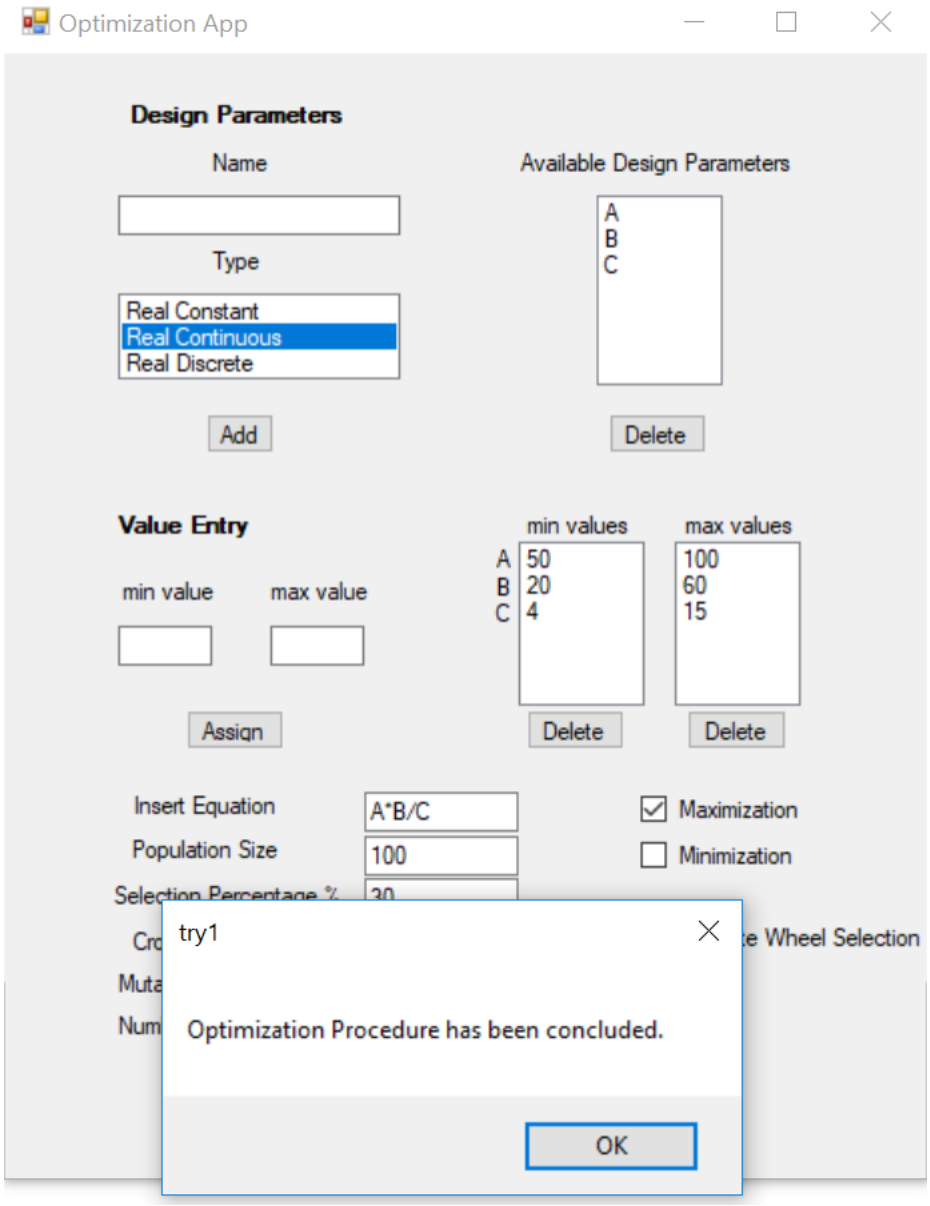
Σχήμα 4.17. Σχηματική αναπαράσταση της διαδικασίας της Μετάλλαξης (Mutation) [19]

Το τελευταίο πράγμα που πρέπει να κάνει ο χρήστης είναι να επιλέξει εάν επιθυμεί μεγιστοποίηση ή ελαχιστοποίηση της εισαγόμενης σχέσης. Η επιλογή αυτή γίνεται πατώντας πάνω στο checkbox με την ένδειξη “Maximization” ή στο checkbox με την ένδειξη “Minimization”. Προφανώς ο χρήστης πρέπει να επιλέξει είτε μεγιστοποίηση είτε ελαχιστοποίηση, γι’αυτό εάν πατήσει και στα 2 checkbox ή σε κανένα από τα 2 εμφανίζεται σχετικό μήνυμα.



Σχήμα 4.18. Εμφάνιση σχετικού μηνύματος ύστερα από παράλειψη επιλογής μεγιστοποίησης / ελαχιστοποίησης

Έπειτα από τη διαδικασία της δημιουργίας του αρχικού πληθυσμού, τα επόμενα βήματα (Επιλογή, Διασταύρωση, Μετάλλαξη) επαναλαμβάνονται μέχρις ότου είτε πάρουμε κάποια επιθυμητή τιμή της συνάρτησης καταλληλότητας που ψάχνουμε, είτε για συγκεκριμένο αριθμό επαναλήψεων, ο οποίος μπορεί να δηλωθεί από το χρήστη στο πεδίο “Number of Iterations”.
Πατώντας το κουμπί “Solve”, ξεκινάει η διαδικασία της βελτιστοποίησης και μετά από λίγα δευτερόλεπτα εμφανίζεται ένα μήνυμα που ενημερώνει το χρήστη ότι η διαδικασία τελείωσε επιτυχώς.

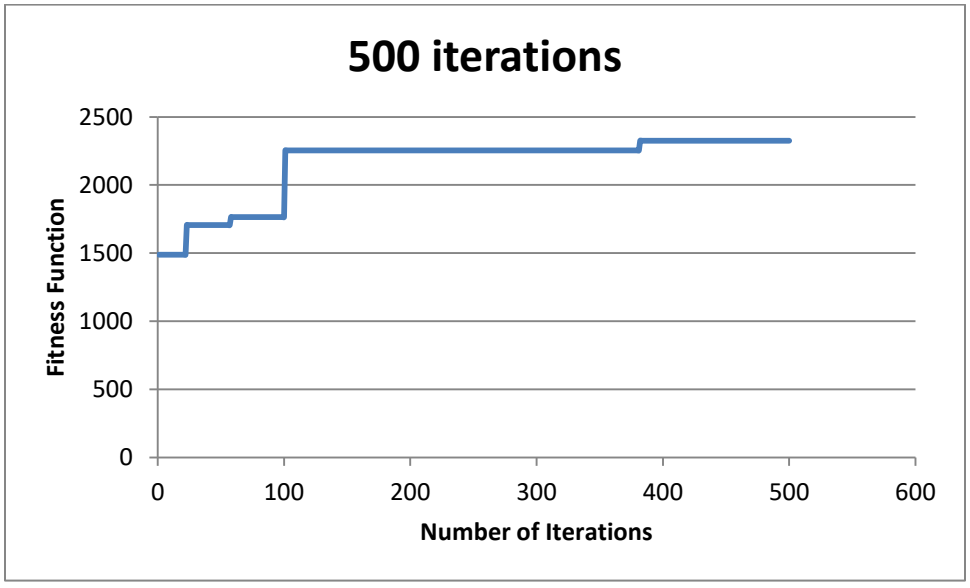


Σχήμα 4.19. Επιτυχής ολοκήρωση της διαδικασίας βελτιστοποίησης

Το περιβάλλον διαθέτει και ένα κουμπί “Clear”. Πατώντας το κουμπί αυτό, μηδενίζονται οι τιμές των μεταβλητών και όλα τα πεδία αδειάζουν, ώστε ο χρήστης να μπορεί να εισάγει νέα δεδομένα.

4.4 ΠΑΡΟΥΣΙΑΣΗ ΑΠΟΤΕΛΕΣΜΑΤΩΝ

Με τη λήξη της διαδικασίας, δημιουργείται σε επιλεγμένο φάκελο ένα αρχείο κειμένου, το οποίο περιέχει τον αύξοντα αριθμό των επαναλήψεων που έτρεξε ο αλγόριθμος και τη μέγιστη τιμή συνάρτησης καταλληλότητας ανά επανάληψη. Ανοίγοντας το αρχείο αυτό με το Microsoft Office Excel, παίρνουμε την καμπύλη μέγιστης τιμής συνάρτησης καταλληλότητας και αριθμού επαναλήψεων. Η καμπύλη αυτή θα αποτελέσει τον τρόπο παρουσίασης των αποτελεσμάτων της πορείας του αλγορίθμου και έχει την ακόλουθη μορφή.



Σχήμα 4.20. Παράδειγμα καμπύλης Συνάρτησης Καταλληλότητας – Αριθμού Επαναλήψεων

4.5 ΕΦΑΡΜΟΓΗ Ι ΠΡΟΒΟΛΗ ΔΟΚΟΣ

Στο παράδειγμα αυτό εξετάζεται η ακαμψία (σταθερά ελατηρίου) k μιας πρόβολης δοκού, η οποία περιγράφεται από την εξίσωση

$$k = \frac{F}{\delta} \left(\frac{N}{m}\right)$$

Όμως ως γνωστόν:

$$F = \sigma \cdot A \, (N)$$

$$\sigma = E \cdot \varepsilon \, (Pa)$$

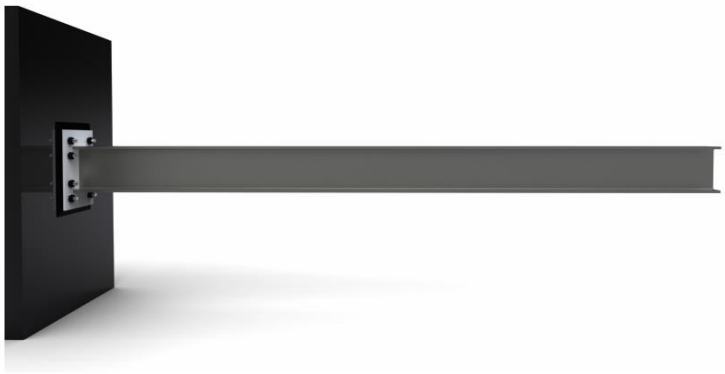
$$\varepsilon = \frac{\delta}{l}$$

Άρα:

$$k = \frac{\sigma \cdot A}{\delta} = \frac{E \cdot \varepsilon \cdot A}{\delta} = \frac{E \cdot \delta \cdot A}{l \cdot \delta} = \frac{E \cdot A}{l} \left(\frac{N}{m}\right)$$

Όπου:

- k η ακαμψία ή αλλιώς σταθερά ελατηρίου της δοκού (N/m)
- F η δύναμη που ασκείται στη δοκό (N)
- δ η μεταβολή του μήκους της δοκού λόγω παραμόρφωσης (m)
- σ η τάση στην οποία υπόκειται η δοκός λόγω της εφαρμογής της δύναμης F
- ε η παραμόρφωση της δοκού
- E το μέτρο ελαστικότητας ή αλλιώς μέτρο Young της δοκού (Pa)
- A η διατομή της δοκού (m^2)
- l το μήκος της δοκού (m)



Σχήμα 4.21. Μια τυπική δοκός [20]

Η γνωστή μαθηματική σχέση που εξήχθη από την ανωτέρω ανάλυση θα αποτελέσει και την αντικειμενική συνάρτηση του προβλήματος βελτιστοποίησης που θα επιλυθεί, ή αλλιώς τη γνωστή συνάρτηση καταλληλότητας, μιλώντας σε ορολογία γενετικών αλγορίθμων.
Τα εύρη τιμών των σχεδιαστικών παραμέτρων που θα χρησιμοποιηθούν για την εφαρμογή της δοκού παρατίθενται στον παρακάτω πίνακα.

Μεταβλητή	Ελάχιστη Τιμή	Μέγιστη Τιμή	Μονάδα Μέτρησης
E	140	180	Gpa
A	3	8	m^2
l	5	10	m

Πίνακας 1. Εύρος τιμών των σχεδιαστικών παραμέτρων

Στο πρόβλημα αυτό θα γίνει μεγιστοποίηση της ποσότητας k.

$$max(k) = max\left(\frac{E \cdot A}{l}\right)$$

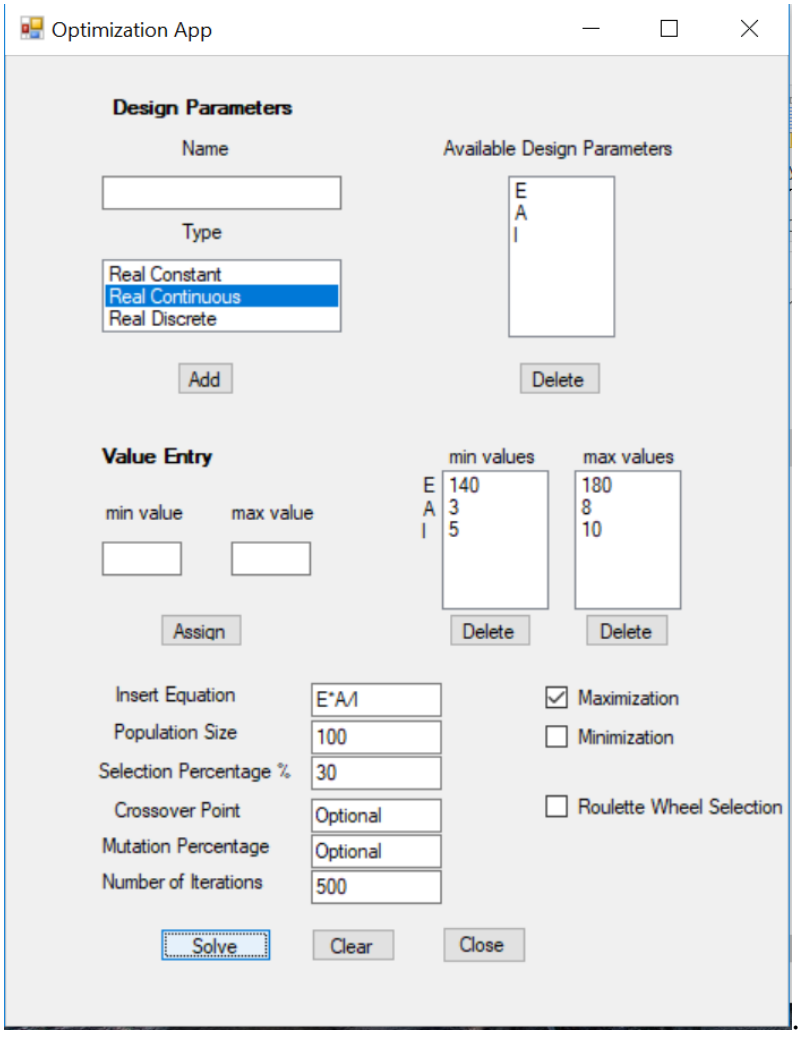
Υπό τους περιορισμούς:

$$140GPa \leq E \leq 180GPa$$

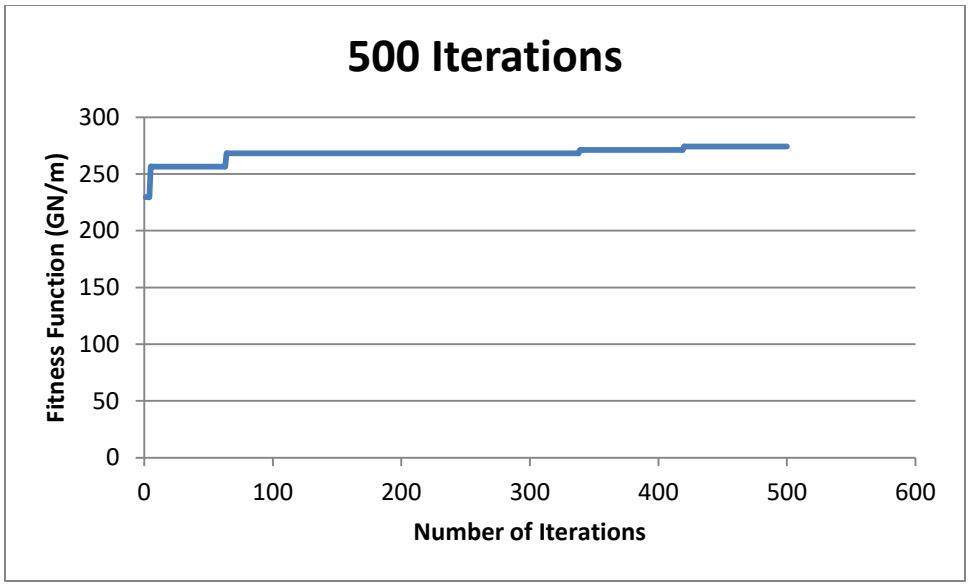
$$3m^2 \leq A \leq 8m^2$$

$$5m \leq l \leq 10m$$

Θα γίνουν δοκιμές για 500, 1000, 3000 και 5000 επαναλήψεις και τα αποτελέσματα θα παρουσιαστούν σε καμπύλες με τη μέγιστη ανά επανάληψη τιμή της συνάρτησης καταλληλότητας συναρτήσει του αριθμού των επαναλήψεων. Ο γενετικός αλγόριθμος ξεκινά δημιουργώντας έναν αρχικό πληθυσμό 100 μελών και το ποσοστό των λύσεων που επιλέγονται ως καλύτερες και επιβιώνουν σε επόμενη γενιά είναι 30%. Δεν εισάγονται συγκεκριμένα σημεία διασταύρωσης και ποσοστό μετάλλαξης, ώστε το πρόγραμμα να τα επιλέγει τυχαία ανά επανάληψη.



Σχήμα 4.22. Εισαγωγή των δεδομένων στο πρόγραμμα



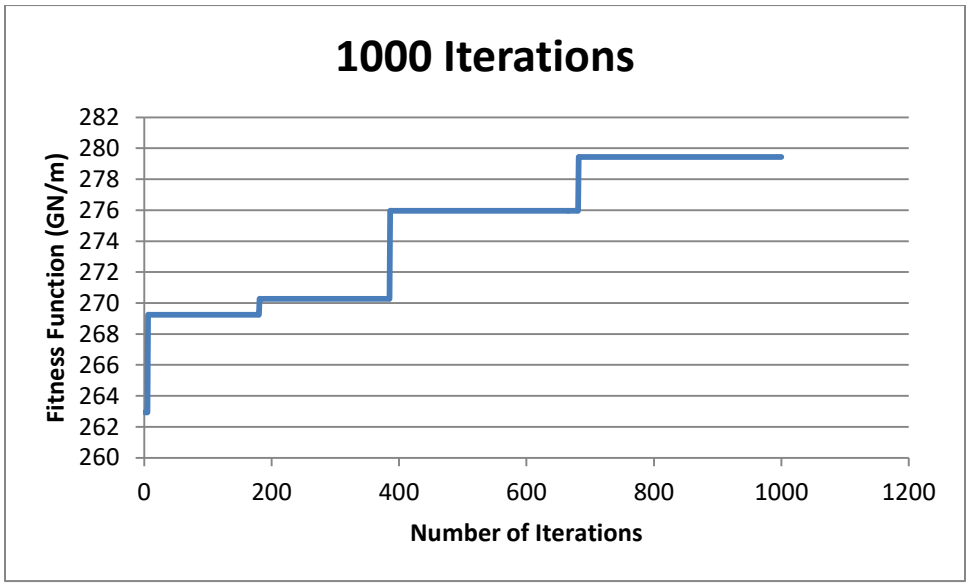
Σχήμα 4.23. Η πορεία του αλγορίθμου για 500 επαναλήψεις

Επιμέρους Αποτελέσματα

$E = 179$

$A = 7.817$

$l = 5.103$



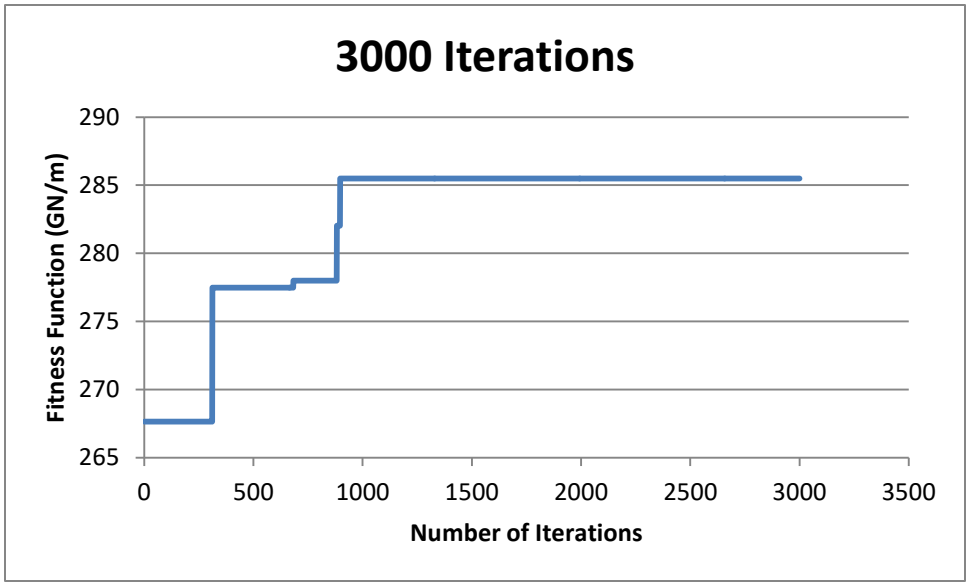
Σχήμα 4.24. Η πορεία του αλγορίθμου για 1000 επαναλήψεις

Επιμέρους Αποτελέσματα

$E = 178$

$A = 7.909$

$l = 5.038$



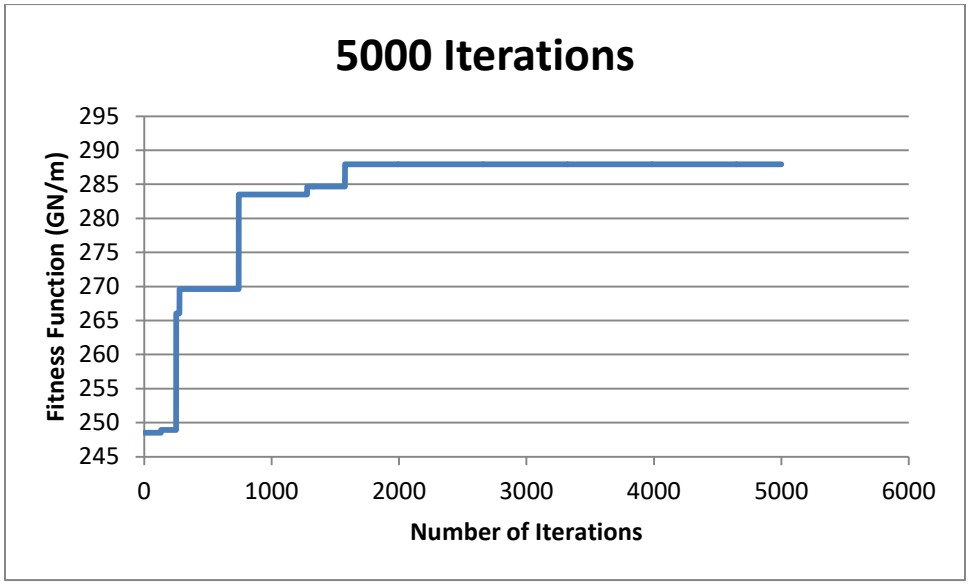
Σχήμα 4.25. Η πορεία του αλγορίθμου για 3000 επαναλήψεις

Επιμέρους Αποτελέσματα

$E = 179$

$A = 7.981$

$l = 5.004$



Σχήμα 4.26. Η πορεία του αλγορίθμου για 5000 επαναλήψεις

Επιμέρους Αποτελέσματα

$E = 180$

$A = 8$

$l = 5.001$

Όπως φαίνεται και από τα παραπάνω διαγράμματα ο αλγόριθμος συγκλίνει προς τη μέγιστη τιμή του k υπό τους διατυπωμένους περιορισμούς και μάλιστα όσο αυξάνεται ο αριθμός των επαναλήψεων, αυξάνεται και η ακρίβεια της προσέγγισης.

Πιο συγκεκριμένα, ο ακόλουθος πίνακας δείχνει τις βέλτιστες προσεγγίσεις και το σφάλμα για τους αριθμούς των επαναλήψεων που εφαρμόστηκε ο αλγόριθμος γνωρίζοντας ότι η βέλτιστη τιμή για τις τιμές των σχεδιαστικών παραμέτρων που εισήχθησαν είναι

$$k = 288GN/m$$

Number of Iterations	Calculated Value	Real Value	Error %
500	274.2000784	288	4.7916
1000	279.4366812	288	2.9734
3000	285.4914069	288	0.871
5000	287.9424115	288	0.02

Πίνακας 2. Υπολογισμός του σφάλματος για τις βέλτιστες τιμές

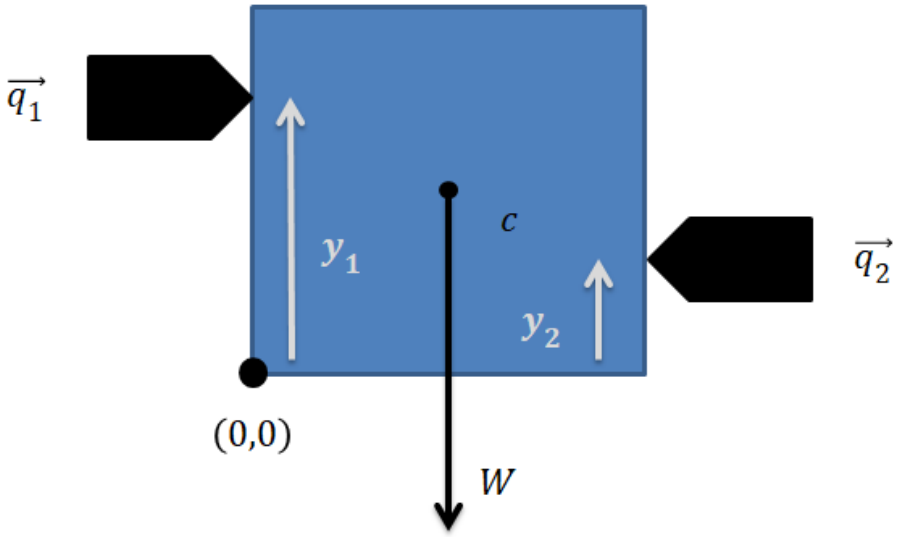
4.6 ΕΦΑΡΜΟΓΗ ΙΙ ΡΟΜΠΟΤΙΚΟ ΧΕΡΙ

Έστω ένα ρομποτικό χέρι, το οποίο χρησιμοποιώντας δύο δάχτυλα ασκεί δυνάμεις σε αντικείμενα και μπορεί έτσι να τα πιάσει προσομοιώνοντας ένα αληθινό ανθρώπινο χέρι και στη συνέχεια να τα μετακινήσει. Το ρομπότ μπορεί να ασκήσει πολύ μικρές δυνάμεις στο αντικείμενο μέσω των ακροδαχτύλων του. Αυτό οφείλεται στα μοτέρ τα οποία μπαίνουν στο

χέρι και λόγω της ειδικής γεωμετρίας και του μικρού μεγέθους του χεριού πρέπει να είναι μικρά και συνεπώς μικρής ισχύος. Εκτός αυτού, είναι επιθυμητό το ρομπότ να πιάνει με ασφάλεια τα αντικείμενα, μπορεί να θεωρηθεί δηλαδή ότι απαιτείται ένα επίπεδο ευαισθησίας – ενδοτικότητας στην επαφή των δαχτύλων και του αντικειμένου, το οποίο μπορεί επίσης να είναι και εύθραυστο.

Σχήμα 4.27. Ένα ρομποτικό χέρι πιάνει με δύο δάχτυλα ένα εύθραυστο αντικείμενο [21]

Αν θεωρηθεί ότι το αντικείμενο που πιάνει το ρομπότ είναι ορθογωνικό και γίνει απλούστευση στις δύο διαστάσεις, τότε παίρνουμε το ακόλουθο σχήμα.



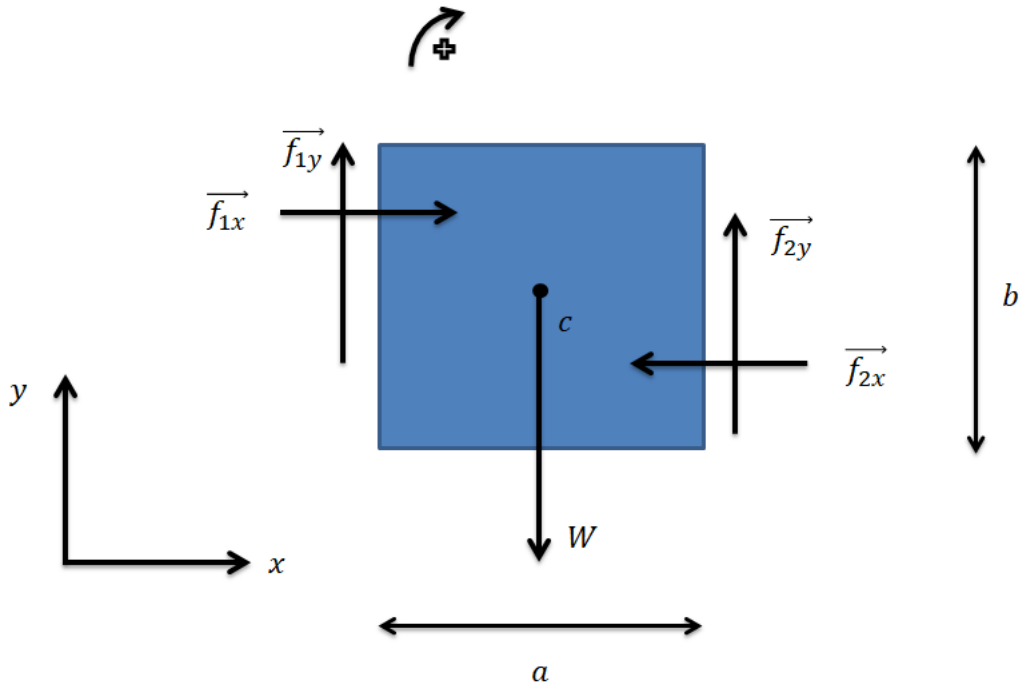
Σχήμα 4.28. Δάχτυλα τοποθετημένα στις δύο πλαϊνές πλευρές του αντικειμένου

Το c παριστάνει το κέντρο μάζας, υποθέτοντας ομοιόμορφη κατανομή μάζας στο αντικείμενο, και άρα τοποθετείται στο γεωμετρικό κέντρο του ορθογωνίου. Η επαφή του κάθε δαχτύλου με το αντικείμενο θεωρείται σημειακή, ώστε να μη γίνει πολύ περίπλοκο το πρόβλημα. Τα $\overrightarrow{q_1}$ και $\overrightarrow{q_2}$ είναι διανύσματα θέσης ξεκινώντας από την αρχή των συντεταγμένων καταλήγοντας στο σημείο επαφής του κάθε δαχτύλου αντίστοιχα με το αντικείμενο. Αναζητείται ο βέλτιστος συνδυασμός των κατακόρυφων αποστάσεων y_1 και y_2, δηλαδή των κατακόρυφων αποστάσεων των θέσεων τοποθέτησης των δαχτύλων, προκειμένου να υπάρχει:

- ισορροπία στον οριζόντιο άξονα
- ισορροπία στον κατακόρυφο άξονα και
- ισορροπία στις ροπές

Επιπλέον θα πρέπει το άθροισμα των κάθετων δυνάμεων f_{1x} και f_{2x} να είναι όσο δυνατόν μικρότερο. Γίνεται έλεγχος μόνο στις κάθετες δυνάμεις επάνω στο αντικείμενο. Οι υπόλοιπες εφαπτομενικές δυνάμεις προκύπτουν λόγω τριβής και μάλιστα υποτίθεται ότι υφίσταται οριακή τριβή. Ακολουθεί το διάγραμμα ελευθέρου σώματος του αντικειμένου, το οποίο παρουσιάζει τις

δυνάμεις που αυτό δέχεται, καθώς και μερικές απαραίτητες αποστάσεις που απαιτούνται για τις πράξεις στη συνέχεια.



Σχήμα 4.29. Διάγραμμα Ελευθέρου Σώματος του Αντικειμένου

$$\overrightarrow{q_1} = [x_1, y_1]^T$$
$$\overrightarrow{q_2} = [x_2, y_2]^T$$
$$\overrightarrow{r_c} = [x_c, y_c]^T$$
$$\overrightarrow{f_1} = [f_{1x}, f_{1y}]^T$$
$$\overrightarrow{f_2} = [f_{2x}, f_{2y}]^T$$
$$W = m \cdot g$$
$$\mu$$

Ισορροπία:

$$\overrightarrow{\Sigma F} = \vec{0} \Rightarrow \begin{cases} f_{1x} - f_{2x} = 0 \\ f_{1y} + f_{2y} - W = 0 \end{cases} \quad (1)$$

Όμως ισχύει:

$$\begin{cases} f_{1y} = \mu \cdot f_{1x} \\ f_{2y} = \mu \cdot f_{2x} \end{cases}$$

Άρα:

$$(1) \Rightarrow \begin{cases} f_{1x} - f_{2x} = 0 & (\Sigma F_x) \\ \mu \cdot (f_{1x} + f_{2x}) - W = 0 & (\Sigma F_{y)} \end{cases}$$

Είναι:

$$\overrightarrow{\Sigma M} = \vec{0} \Rightarrow \overrightarrow{r_1^c} \times \overrightarrow{f_1} + \overrightarrow{r_2^c} \times \overrightarrow{f_2} = \vec{0} \Rightarrow$$
$$(\overrightarrow{q_1} - \overrightarrow{r_c}) \times \overrightarrow{f_1} + (\overrightarrow{q_2} - \overrightarrow{r_c}) \times \overrightarrow{f_2} = \vec{0}$$

Η υλοποίηση στο περιβάλλον δεν υποστηρίζει ισοτικούς περιορισμούς σε αυτό το παράδειγμα, όμως οι επιθυμητές τιμές είναι οι εξής:

$$\Sigma F_x = 0$$
$$\Sigma F_y = 0$$
$$\Sigma M = 0$$
$$f_{1x} + f_{2x} \ll$$

Γι'αυτό θα γίνει ενσωμάτωση των ισοτικών περιορισμών – συνθηκών ισορροπίας στην αντικειμενική συνάρτηση. Υψώνοντας στο τετράγωνο τους όρους ΣF_x, ΣF_y και ΣM και αθροίζοντάς τους εξασφαλίζεται η ισορροπία, η οποία συνεπάγεται και τη σύγκλισή τους προς το 0. Επίσης προσθέτοντας τις δύο κάθετες δυνάμεις f_{1x} και f_{2x} και απαιτώντας ελαχιστοποίηση του συνολικού αθροίσματος εξασφαλίζεται η ζητούμενη ελαχιστοποίηση των δύο αυτών δυνάμεων.

Βάσει της παραπάνω ανάλυσης, η αντικειμενική συνάρτηση – συνάρτηση καταλληλότητας διαμορφώνεται ως εξής:

$$C = Cf_x + Cf_y + C_M + C_f$$

Όπου:

$$Cf_x = (\Sigma F_x)^2 = (f_{1x} - f_{2x})^2$$
$$Cf_y = \left(\Sigma F_y\right)^2 = [\mu \cdot (f_{1x} + f_{2x}) - W]^2$$
$$C_M = (\Sigma M)^2 = [f_{1x} \cdot (\mu \cdot x_c + y_1 - y_c) - f_{2x} \cdot (\mu \cdot x_c + y_c - y_2)]^2$$
$$C_f = f_{1x} + f_{2z}$$

Άρα τίθεται το πρόβλημα ελαχιστοποίησης:

$$argmin C$$
$$y_1, y_2 \epsilon [y_{min}, y_{max}] = [0, b]$$
$$f_{1x}, f_{2x} \in [f_{min}, f_{max}]$$

Το βάρος, οι διαστάσεις του αντικειμένου και ο συντελεστής τριβής είναι σταθερές τιμές και στο παράδειγμα που θα εφαρμοστεί στο περιβάλλον είναι οι ακόλουθες:

$$W = m \cdot g = 2N$$
$$a = 0.4m$$
$$b = 0.5m$$
$$\mu = 0.7$$

Έχοντας ορίσει τις διαστάσεις του αντικειμένου και υποθέτοντας ομοιόμορφα κατανεμημένη τη μάζα του, εύκολα συνεπάγεται για τις συντεταγμένες του κέντρου μάζας ότι:

$$x_c = \frac{a}{2} = 0.2m$$
$$y_c = \frac{b}{2} = 0.25m$$

Βάσει των τιμών για το συγκεκριμένο παράδειγμα και θεωρώντας μέγιστη δύναμη 5N, οι περιορισμοί διαμορφώνονται ως εξής:

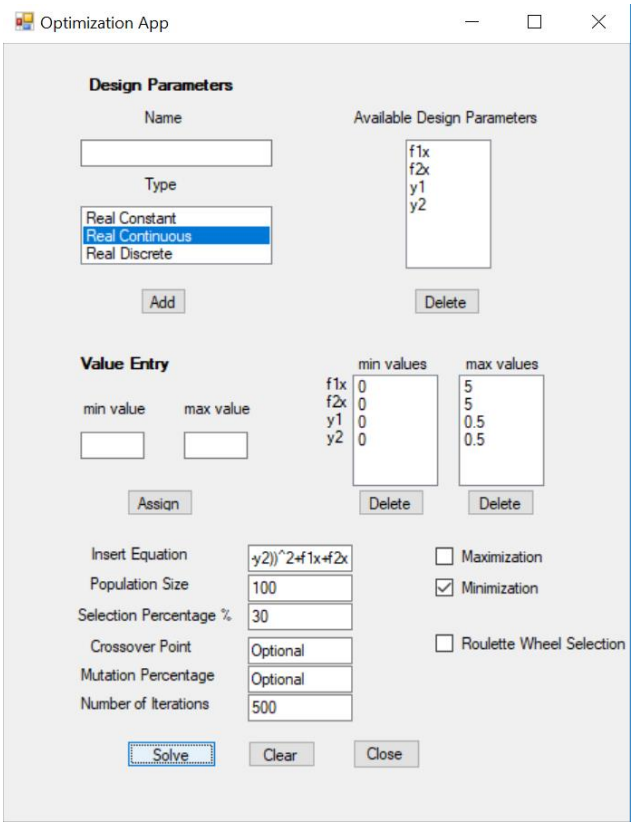
$$0 \leq y_1 \leq 0.5m$$

$$0 \leq y_2 \leq 0.5m$$

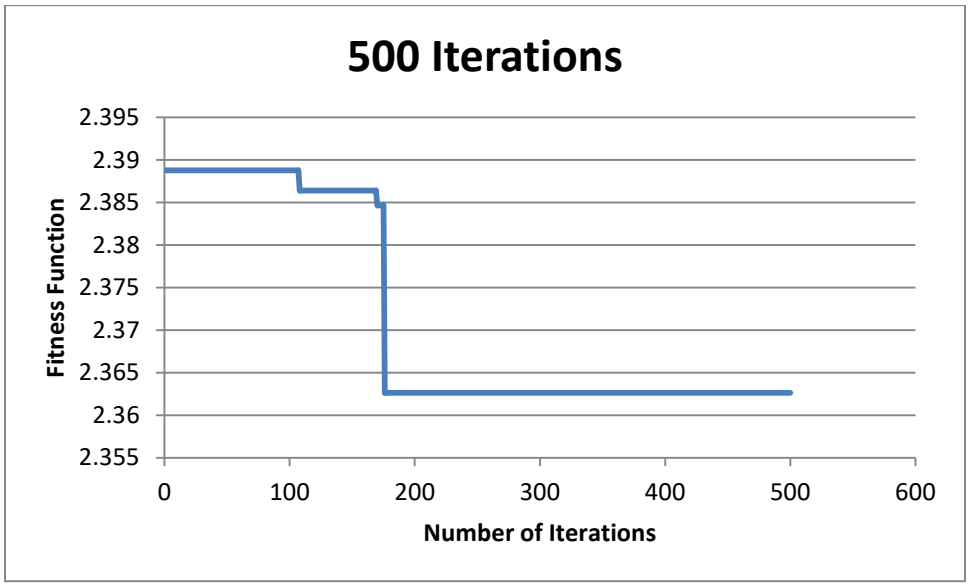
$$0 \leq f_{1x} \leq 5N$$

$$0 \leq f_{2x} \leq 5N$$

Η εφαρμογή θα υλοποιηθεί στο περιβάλλον για αρχικό πληθυσμό 100 μελών, με επιλογή των ή των 30% των καλύτερων από κάθε γενιά και ο αλγόριθμος θα τρέξει για 500, 1000, 3000 και 5000 επαναλήψεις – γενιές. Έπειτα από κάθε διάγραμμα θα παρουσιαστούν και οι βέλτιστες υπολογισθείσες τιμές για τα y_1, y_2, f_{1x}, f_{2x} , ώστε να εξασφαλισθεί ότι βρίσκονται εντός ορίων και ο μετασχηματισμός της συνάρτησης καταλληλότητας λειτουργεί σωστά για να καλύψει τους ισοτικούς περιορισμούς του προβλήματος.



Σχήμα 4.30. Εισαγωγή στοιχείων για την εφαρμογή του ρομποτικού χεριού



Σχήμα 4.31. Η πορεία του αλγορίθμου για 500 επαναλήψεις

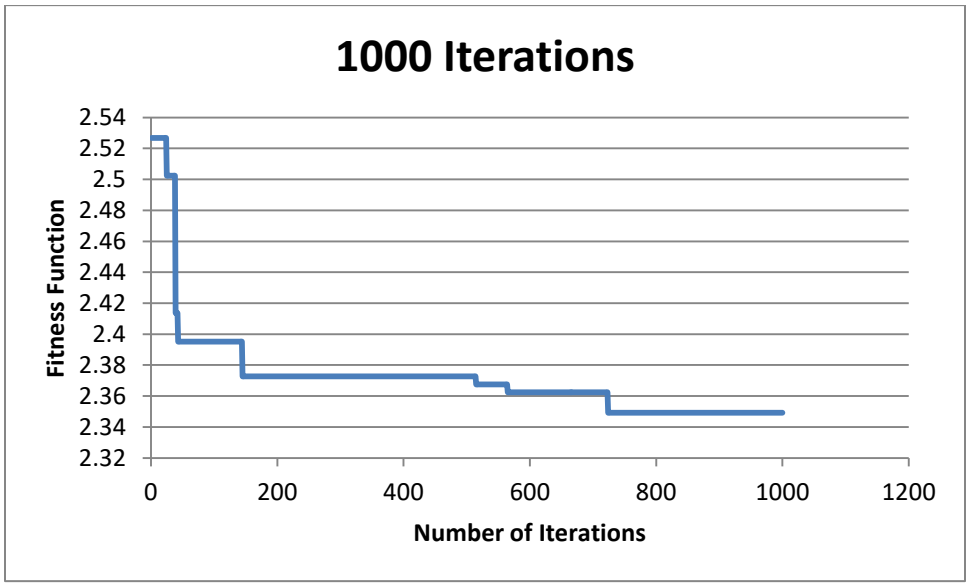
Επιμέρους Αποτελέσματα

$$f_{1x} = 0.811N$$

$$f_{2x} = 0.899N$$

$$y_1 = 0.275$$

$$y_2 = 0.232m$$



Σχήμα 4.32. Η πορεία του αλγορίθμου για 1000 επαναλήψεις

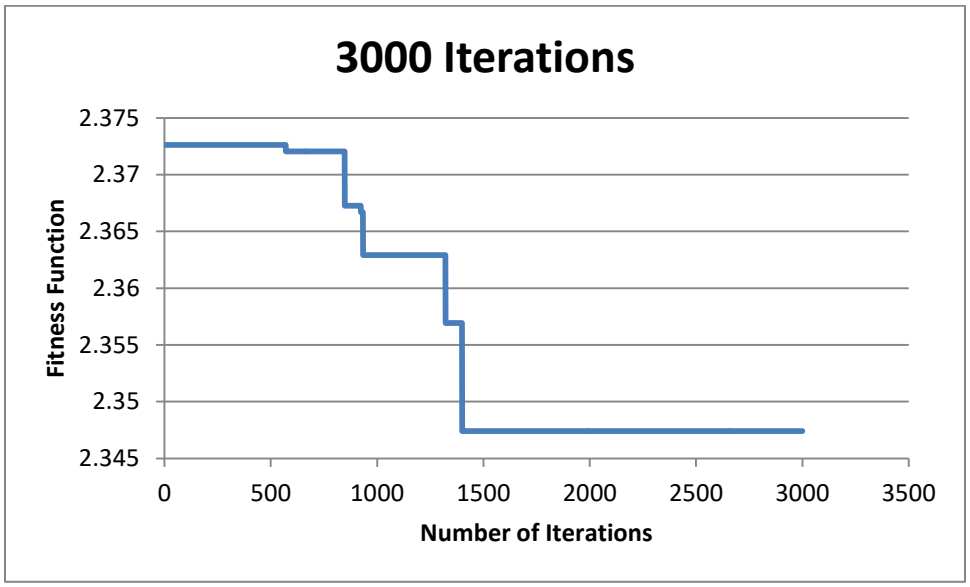
Επιμέρους Αποτελέσματα

$f_{1x} = 0.897N$

$f_{2x} = 0.878N$

$y_1 = 0.290m$

$y_2 = 0.199m$



Σχήμα 4.33. Η πορεία του αλγορίθμου για 3000 επαναλήψεις

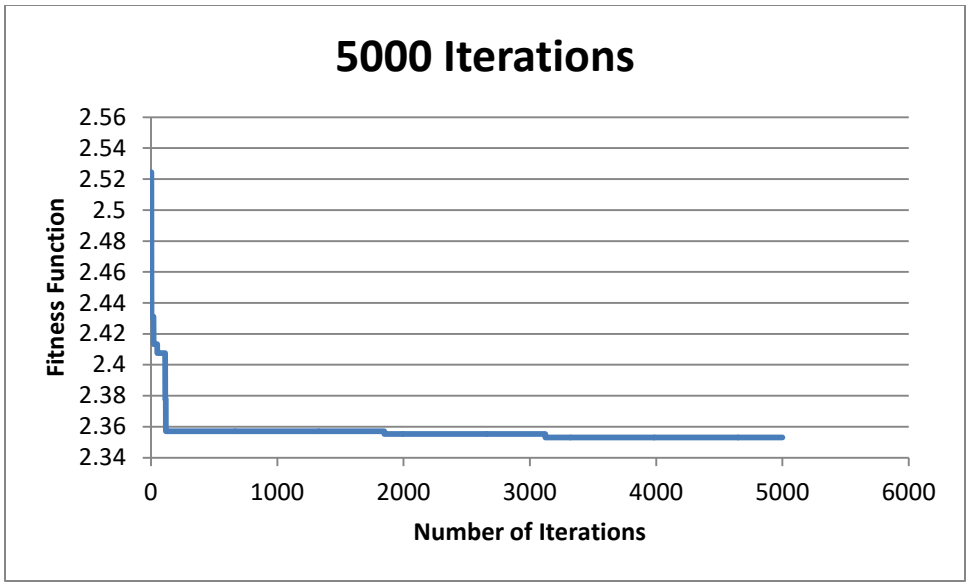
Επιμέρους Αποτελέσματα

$f_{1x} = 0.905N$

$f_{2x} = 0.902N$

$y_1 = 0.213m$

$y_2 = 0.292m$



Σχήμα 4.34. Η πορεία του αλγορίθμου για 5000 επαναλήψεις

Επιμέρους Αποτελέσματα

$$f_{1x} = 0.905N$$

$$f_{2x} = 0.902N$$

$$y_1 = 0.213m$$

$$y_2 = 0.292m$$

Οι τιμές που δίνουν οι εκτελέσεις του αλγορίθμου βρίσκονται όλες εντός των επιτρεπτών ορίων που διατυπώθηκαν προηγουμένως και ο αλγόριθμος συγκλίνει στο 0, το οποίο είναι και η βέλτιστη επιθυμητή τιμή ώστε το σώμα να μη δέχεται δυνάμεις. Ωστόσο από τα διαγράμματα φαίνεται ότι με την αύξηση του αριθμού των επαναλήψεων δεν επιτυγχάνεται καλύτερη σύγκλιση.

Η ακρίβεια αυτή οφείλεται στη διαμόρφωση που έγινε στην αντικειμενική συνάρτηση, η οποία, ως άθροισμα μπορεί να μην ικανοποιεί όλες τις επιμέρους συνθήκες, καθώς επίσης και στην τυχαιότητα που ως γνωστόν υπεισέρχεται στη διαδικασία των γενετικών αλγορίθμων.

4.7 ΠΛΕΟΝΕΚΤΗΜΑΤΑ ΧΡΗΣΗΣ ΤΟΥ ΑΝΩΤΕΡΩ ΠΕΡΙΒΑΛΛΟΝΤΟΣ

Τα κυριότερα πλεονεκτήματα της χρήσης του περιβάλλοντος που περιεγράφηκε στην προηγούμενη ενότητα αναμένεται να είναι τα εξής:

- Δυνατότητα πραγματοποίησης βελτιστοποίησης σε πραγματικό χρόνο. Οι σχεδιαστικές παράμετροι και η αντικειμενική συνάρτηση δηλώνονται τη στιγμή που τρέχει το πρόγραμμα και δεν απαιτούνται αλλαγές στον κώδικα για κάθε διαφορετικό πρόβλημα.
- Δυνατότητα συγκριτικής ανάλυσης και αξιολόγησης εναλλακτικών σχεδιασμών (Concepts), δηλώνοντας διάφορους συνδυασμούς τιμών σχεδιαστικών παραμέτρων.
- Αντιμετώπιση πολλών ειδών προβλημάτων και αντικειμενικών συναρτήσεων χάρη στην παραμετρική σχεδίαση που επιτρέπει η χρήση του parser.
- Δυνατότητα επεξεργασίας των ήδη δηλωμένων σχεδιαστικών παραμέτρων και των τιμών τους μέσα από ειδικά προγραμματιζόμενα κουμπιά.
- Απεικόνιση των δηλωμένων σχεδιαστικών σχεδιαστικών παραμέτρων καθώς και των τιμών τους προς διευκόλυνση του χρήστη. Ο χρήστης έχει έτσι μια εποπτική ματιά των δεδομένων του προβλήματος.
- Δυνατότητα εισαγωγής πραγματικών διακριτών αλλά και συνεχών σχεδιαστικών παραμέτρων.
- Μετατροπή αριθμητικών εκφράσεων υπό μορφή κειμένου σε αναλυτικές υπολογιστικές σχέσεις μέσω χρήσης parser.
- Υλοποίηση βελτιστοποίησης με χρήση γενετικών αλγορίθμων, μια από τις καλύτερες σύγχρονες μεθόδους βελτιστοποίησης, της οποίας τα πλεονεκτήματα έναντι άλλων μεθόδων είναι πάρα πολλά και αναφέρθηκαν στο προηγούμενο κεφάλαιο.
- Δυνατότητα Ελαχιστοποίησης αλλά και Μεγιστοποίησης της αντικειμενικής συνάρτησης. Η επιλογή γίνεται πατώντας σε ένα από τα 2 διαθέσιμα checkbox.
- Δυνατότητα επιλογής των τιμών των χαρακτηριστικών μεγεθών που υπεισέρχονται στη διαδικασία της βελτιστοποίησης με γενετικό αλγόριθμο, όπως επιλογή Μεγέθους Πληθυσμού, Ποσοστού Επιλογής χρωμοσωμάτων προς επιβίωση, σημείου διασταύρωσης, ποσοστού Μετάλλαξης, αριθμού επαναλήψεων του αλγορίθμου.

- Δυνατότητα επεξεργασίας πολλών ειδών αντικειμενικών συναρτήσεων, με διάφορες πράξεις, και εμφάνιση καταλόγου με διαθέσιμες υποστηριζόμενες πράξεις και σύμβολα προς διευκόλυνση του χρήστη.
- Εμφάνιση πεδίου τιμών του χρωμοσώματος που έχει δημιουργηθεί, ώστε να ξέρει ο χρήστης τι τιμή μπορεί να εισάγει ως σημείο διασταύρωσης.
- Συνεχής έλεγχος εγκυρότητας εισαγόμενων ποσοτήτων, τόσο στο πεδίο δήλωσης των σχεδιαστικών παραμέτρων και των τιμών τους, όσο και στα επόμενα πεδία που αφορούν τη διαδικασία βελτιστοποίησης αυτή καθαυτή.
- Απλό και ευνόητο interface, με λίγα κουμπιά και κατανοητά πεδία εισαγωγής τιμών.
- Συνοπτική και παραστατική απεικόνιση αποτελεσμάτων μέσω του γνωστού προγράμματος Microsoft Office Excel. Μάλιστα το πρόγραμμα θα δημιουργεί ένα αρχείο με αποτελέσματα, αναγνώσιμο από Excel, το οποίο θα μπορεί ο χρήστης να κρατήσει ως αρχείο.

4.8 ΣΥΜΠΕΡΑΣΜΑΤΑ – ΠΕΡΑΙΤΕΡΩ ΠΡΟΟΠΤΙΚΕΣ

Το περιβάλλον που περιεγράφηκε στις προηγούμενες ενότητες μπορεί να υλοποιηθεί σε μια απλή αντικειμενοστραφή γλώσσα προγραμματισμού, επιλέγοντας έναν parser από τους πολλούς που υπάρχουν online. Διαφορετικά, υπάρχει και η δυνατότητα δημιουργίας ενός custom parser, ώστε να επιλεχθεί από το σχεδιαστή ο κατάλογος των υποστηριζόμενων πράξεων και συμβόλων, ώστε το περιβάλλον βελτιστοποίησης να μπορεί να επεξεργαστεί ακόμα περισσότερα και πιο πολύπλοκα είδη αντικειμενικών συναρτήσεων.

Σε κάθε περίπτωση, η δημιουργία ενός τέτοιου περιβάλλοντος θα ήταν πολύ χρήσιμη για ένα σχεδιαστή μηχανικό, καθώς θα μπορεί να λύσει «επί τόπου» το πρόβλημά του (Right On Time). Θα μπορεί να εισάγει τις σχεδιαστικές παραμέτρους του προβλήματός του, τις τιμές τους, καθώς και τα κριτήρια που θέλει να γίνει η βελτιστοποίηση και μέσα σε λίγα δευτερόλεπτα να μπορεί να δει την πορεία του αλγορίθμου και την προτεινόμενη υπολογισθείσα λύση μέσα από ένα παραστατικό και κατανοητό γράφημα. Το περιβάλλον βέβαια μπορεί να σχεδιαστεί ως μια εφαρμογή ανεξάρτητη, με την έννοια ότι θα μπορεί να τρέξει δίχως την απαίτηση να τρέχει την ίδια στιγμή κάποια γλώσσα προγραμματισμού, δηλαδή σαν ένα Windows Application.

Ένα απλό είδος προβλήματος βελτιστοποίησης που μπορεί να αντιμετωπιστεί από το περιβάλλον αυτό είναι το πρόβλημα της δοκού που περιεγράφηκε στην παρούσα εργασία, αποτελούμενο από πραγματικές διακριτές και συνεχείς σχεδιαστικές παραμέτρους. Προβλήματα με τέτοιου είδους απλές αντικειμενικές συναρτήσεις αντιμετωπίζονται με μεγάλη ταχύτητα και αποτελεσματικότητα από το περιβάλλον και ο σχεδιαστής μπορεί έτσι να συγκρίνει διάφορους εναλλακτικούς σχεδιασμούς που έχει διαμορφώσει βάσει των σχεδιαστικών περιορισμών και προδιαγραφών.

Παρατίθενται εδώ ορισμένες επιπλέον δυνατότητες – προοπτικές εξέλιξης της ιδέας για τη δημιουργία του περιβάλλοντος.

- Δυνατότητα αποθήκευσης του δημιουργούμενου project, συμπεριλαμβανομένου των σχεδιαστικών παραμέτρων, των τιμών τους, της συνάρτησης καταλληλότητας και των δεδομένων για τη διαδικασία της βελτιστοποίησης με το γενετικό αλγόριθμο. Έτσι, ο χρήστης θα μπορεί να ανακαλεί τα δημιουργηθέντα projects ανά πάσα στιγμή.

- Δημιουργία και χρήση ενός parser που θα υποστηρίζει τις πράξεις και τα σύμβολα που θέλει ο ίδιος ο σχεδιαστής και θα υιοθετεί τη γραμματική και το συντακτικό που αυτός επιθυμεί στο πλαίσιο του προγραμματισμού.
- Δυνατότητα αντιμετώπισης όχι μόνο μοναδιαίων περιορισμών (unary constraints) αλλά και άλλων ειδών περιορισμών προς αντιμετώπιση μεγαλύτερου εύρους μηχανολογικών και όχι μόνο προβλημάτων. (αρκετά προβλήματα επιχειρησιακής έρευνας θα μπορούσαν να επιλυθούν με αυτή τη δυνατότητα)
- Δυνατότητα περισσότερων επιλογών όσον αφορά τη διαδικασία της βελτιστοποίησης με γενετικό αλγόριθμο, παραδείγματος χάρη:
    - Δυνατότητα υλοποίησης της επιλογής χρωμοσωμάτων προς επιβίωση (Selection), με επιπλέον μεθόδους, εκτός του Roulette wheel Selection, όπως Tournament Selection, Stochastic Universal Sampling κτλ.
    - Δυνατότητα υλοποίησης της διαδικασίας της Διασταύρωσης και με άλλες τεχνικές εκτός από το Single Point Crossover, όπως Two-Point Crossover, Uniform and half uniform, Three parent Crossover.
    - Δυνατότητα υλοποίησης της διαδικασίας της Μετάλλαξης με επιπλέον τρόπους εκτός του Single-Point Mutation ,όπως Flip Bit, Boundary, Gaussian, Shrink Mutation κτλ.
    - Δυνατότητα επιλογής από το χρήστη του κριτηρίου τερματισμού του γενετικού αλγορίθμου

ΒΙΒΛΙΟΓΡΑΦΙΑ

1. Dentsoras A.J., (2014), Mechanical Systems Design, University of Patras publications, Greece
2. Goldberg D.E., 2002, The design of innovation, Massachusetts, Kluwer Academic Publishers
3. Renner, G., Ekart A., 2003, Genetic Algorithms in computer-aided design, Computer-aided Design, 35, 709-726
4. Likothanassis S., (2001), Genetic Algorithms and Applications, Hellenic Open University publications, Greece
5. Newcastle University Engineering Design Centre, Roulette Wheel Selection, (2007), retrieved from http://www.edc.ncl.ac.uk/highlight/rhjanuary2007g02.php
6. Wikipedia, Genetic Algorithm, retrieved from https://en.wikipedia.org/wiki/Genetic_algorithm
7. Unknown Author, Genetic Algorithms, retrieved from http://neo.lcc.uma.es/cEA-web.es/cEA-web
8. Schatten A., Genetic Algorithms, (2002, June 19), retrieved from http://web.cs.ucdavis.edu/~vemuri/classes/ecs271/Genetic%20Algorithms%20Short%
9. Unknown Author, Genetic Algorithms Tutorial, (2017), retrieved from https://www.tutorialspoint.com/genetic_algorithms/index.htm
10. IBM, Parsers, (2017), retrieved from https://www.ibm.com/support/knowledgecenter
11. Techopedia, Parser, retrieved from https://www.techopedia.com/definition/3854/parser
12. Frost,R.A., Hafiz, R., Callaghan, P., (2008), Parser Combinators for Ambiguous Left-Recursive Grammars, University of Durham, UK
13. Cornell University, Department of Computer Science, Object-Oriented Design and Data Structures, retrieved from https://cs.cornell.edu/courses/cs2112/2014fa/lectures/lec_parsing
14. Unknown Authors, Parsing, retrieved from https://en.wikipedia.org/wiki/Parsing
15. Park C., Pan J., Manocha D., (2013), Real-time optimization-based planning in dynamic environments using GPUs, 2013 IEEE International Conference on Robotics and Automation, CA, USA

16. Cassandras, C.G., (2012), Real-time optimization in complex stochastic environments
17. Elixmann, D., Puschke, J., (2014), A software environment for economic NMPC and dynamic real-time optimization of chemical processes
18. Winsconsin Institutes for Discovery, NEOS Server: State-of-the-Art Solvers for Numerical Optimization, retrieved from http://neos-server.org/neos/
19. KendallPark, Knapsack Approximation with Genetic Algorithms, (2014), retrieved from https://github.com/KendallPark/genetic-algorithm
20. Expedition Workshed, Cantilever Beam, (2017), retrieved from https://expeditionworkshed.org/Interactive-App/push-me-pull-me-cantilever-beam/
21. 33rd square, Teaching Robots How To Manipulate Objects By Having Them Watch Youtube Videos, (2015), retrieved from www.33rdsquare.com/2015/01/teaching-robots-how-to-manipulate.html
22. Alan Bryan, MathParser.NET, (2011), retrieved from https://www.codeproject.com/Articles/274093/Math-Parser-NET

ΠΑΡΑΡΤΗΜΑ Α

Κώδικας Υλοποίησης Υπολογιστικού Περιβάλλοντος

```
PublicClas
s Form1
                Dim varCount As Integer = 0
                Dim varType(10) As Integer
                Dim labelCount As Integer = 0
                Dim nod As Integer = 3 'accuracy / number of decimals
                Dim flagMenu As Integer = 0
                Private Sub lstDesPars_SelectedIndexChanged(sender As Object, e As
            EventArgs) Handles lstDesPars.SelectedIndexChanged
                End Sub
                Private Sub ListBox1_SelectedIndexChanged(sender As Object, e As EventArgs)
            Handles lstDesParType.SelectedIndexChanged
                End Sub
                Private Sub Button1_Click(sender As Object, e As EventArgs) Handles
            cmdAddDesPar.Click
                    'Check if type has been declared
                    If lstDesParType.SelectedItems.Count = 0 Then
                        txtDesParName.Text = "Declare a type !!!!"
                        txtDesParName.Focus()
                        Exit Sub
                    End If
                    'Check whether txtDesParName is empty
                    If txtDesParName.Text = Nothing Then
                        txtDesParName.Text = "Enter a name !!!!"
                        txtDesParName.Focus()
                        Exit Sub
                    End If
                    'Check if DP already exists
                    If lstDesPars.Items.Count >= 1 Then
                        For i = 1 To lstDesPars.Items.Count
                            If txtDesParName.Text = lstDesPars.Items(i - 1) Then
                                txtDesParName.Text = "Enter another name !!!!"
```

```
                    txtDesParName.Focus()
                    Exit Sub
                End If
            Next
        End If
        If lstDesParType.SelectedIndex = 1 Then
            varType(varCount) = 10 ^ nod 'continuous variable
        Else
            varType(varCount) = 1 'discrete variable
        End If
        lstDesPars.Items.Add(txtDesParName.Text)
        varCount = varCount + 1
    End Sub
    Private Sub Button3_Click(sender As Object, e As EventArgs) Handles
cmdDelDesPar.Click
        If lstDesPars.Items.Count = 0 Then
            txtDesParName.Text = "There is nothing here to delete"
        Else
            Dim pos As Integer = lstDesPars.SelectedIndex
            lstDesPars.Items.RemoveAt(pos)
        End If
    End Sub
    Private Sub Button4_Click(sender As Object, e As EventArgs) Handles
cmdAssignValue.Click
        'Check if user forgot to declare a parameter
        If lstDesPars.Items.Count <= lstValues.Items.Count Or
lstDesPars.Items.Count <= lstValues2.Items.Count Then
            txtDesParName.Text = "Enter a value"
            txtDesParName.Focus()
            Exit Sub
        End If
        'Check whether txtValueEntry is empty
        If txtValueEntry.Text = Nothing And txtValueEntry2.Text = Nothing Then
            MsgBox("Enter a minimum and a maximum value!")
        Else
            If txtValueEntry.Text = Nothing Then
                MsgBox("Enter a minimum value!")
                txtValueEntry.Focus()
                Exit Sub
            End If
            If txtValueEntry2.Text = Nothing Then
                MsgBox("Enter a maximum value!")
                txtValueEntry.Focus()
```

```
                Exit Sub
            End If
        End If
        lstValues.Items.Add(txtValueEntry.Text)
        lstValues2.Items.Add(txtValueEntry2.Text)
        txtDesParName.Text = ""
        txtValueEntry.Text = ""
        txtValueEntry2.Text = ""
        If labelCount = 0 Then
            Label16.Text = lstDesPars.Items(0)
        ElseIf labelCount = 1 Then
            Label17.Text = lstDesPars.Items(1)
        ElseIf labelCount = 2 Then
            Label18.Text = lstDesPars.Items(2)
        ElseIf labelCount = 3 Then
            Label19.Text = lstDesPars.Items(3)
        ElseIf labelCount = 4 Then
            Label20.Text = lstDesPars.Items(4)
        ElseIf labelCount = 5 Then
            Label21.Text = lstDesPars.Items(5)
        End If
        labelCount = labelCount + 1
    End Sub
    Private Sub cmdDelValue_Click(sender As Object, e As EventArgs) Handles
cmdDelValue.Click
        If lstValues.Items.Count = 0 Or lstValues.SelectedIndex = -1 Then
            MsgBox("There is nothing here to delete")
        Else
            Dim pos As Integer = lstValues.SelectedIndex
            lstValues.Items.RemoveAt(pos)
            If pos = 0 Then
                Label16.Text = ""
                labelCount = labelCount - 1
            ElseIf pos = 1 Then
                Label17.Text = ""
                labelCount = labelCount - 1
            ElseIf pos = 2 Then
                Label18.Text = ""
                labelCount = labelCount - 1
            ElseIf pos = 3 Then
                Label19.Text = ""
                labelCount = labelCount - 1
            ElseIf pos = 4 Then
```

```
                Label20.Text = ""
                labelCount = labelCount - 1
            ElseIf pos = 5 Then
                Label21.Text = ""
                labelCount = labelCount - 1
            End If
        End If
    End Sub
    Private Sub cmdUpdate_Click(sender As Object, e As EventArgs) Handles
cmdSolve.Click
        If CheckBox1.Checked And CheckBox2.Checked Or CheckBox1.CheckState = 0
And CheckBox2.CheckState = 0 Then
            MsgBox("You need to chose either Maximization or Minimization")
            CheckBox1.CheckState = 0
            CheckBox2.CheckState = 0
        Else
            Dim optFlag As Integer = 0
            If CheckBox2.Checked Then
                optFlag = 1
            End If
            Dim noi As Integer = TextBox4.Text 'number of iterations
            Dim x As Integer = TextBox1.Text 'number of chromosomes needed for
the solution
            Dim dlv(varCount, 2) As Integer  'dlv stands for decimal limit
values
            Dim fitRes(noi) As Double
            Dim parser As New MathParserNet.Parser
            If lstValues.Items.Count = varCount And lstValues2.Items.Count =
varCount Then
                For i = 1 To varCount
                    dlv(i, 1) = lstValues.Items(i - 1) * varType(i - 1)
                    dlv(i, 2) = lstValues2.Items(i - 1) * varType(i - 1)
                Next
                Dim nov(varCount) As Integer 'nov stands for number of values
determining the size of the chromosome
                For i = 1 To varCount
                    nov(i) = dlv(i, 2) - dlv(i, 1)
                Next
                Dim ubl(varCount) As String 'ubl stands for upper binary limit
                For i = 1 To varCount
                    ubl(i) = Convert.ToString(nov(i), 2)
                Next
                Dim b(varCount) As Integer 'b stands for bandwith
```

```
                For i = 1 To varCount
                    b(i) = Len(ubl(i))
                Next
                Dim bandwith As Integer = b(1)
                For i = 2 To varCount
                    bandwith = bandwith + b(i)
                Next
                Dim values(x, varCount) As Integer 'it's later required for
easier calculation of the fitness function
                Dim bvalues(x, varCount) As String
                For i = 1 To x 'initialization of the matrix values so that
check for identical chromosome parts is legit
                    For j = 1 To varCount
                        bvalues(i, j) = "0"
                    Next
                Next
                Dim binval As String = "0"
                Dim val As Integer = 0
                For i = 1 To x
                    For k = 1 To varCount
                        Randomize()
                        val = CInt(Int(((nov(k) - 0 + 1) * Rnd()) + 0))
                        binval = Convert.ToString(val, 2)
                        While Len(binval) <> b(k)
                            binval = binval.Insert(0, "0")
                        End While
                        values(i, k) = val
                        bvalues(i, k) = binval
                    Next
                Next
                Dim chr(x) As String 'creates an array which consists of the
created chromosomes
                Dim fitness(x) As Double
                For g = 1 To noi
                    For i = 1 To x
                        If g = 1 Then
                            chr(i) = bvalues(i, 1)
                            For j = 2 To varCount
                                chr(i) = chr(i) & bvalues(i, j)
                            Next
                            parser.RemoveAllVariables()
                            For j = 1 To varCount
                                Dim realVal As Double = values(i, j) / varType(j
```

```
- 1) + lstValues.Items(j - 1)
                                parser.AddVariable(lstDesPars.Items(j - 1),
realVal)
                            Next
                            Try
                                fitness(i) = parser.SimplifyDouble(txtEq.Text)
                            Catch ex As Exception
                                MsgBox("Undeclared variables or not supported
symbols used ")
                                Exit Sub
                            End Try
                        Else
                            parser.RemoveAllVariables()
                            Dim c As Integer = b(1)
                            Dim realVal1 As Double = Convert.ToInt32(Mid(chr(i),
1, c), 2) / varType(0) + lstValues.Items(0)
                            parser.AddVariable(lstDesPars.Items(0), realVal1)
                            If g = noi And i = x Then
                                MsgBox(realVal1)
                            End If
                            For j = 2 To varCount
                                Dim realVal2 As Double =
Convert.ToInt32(Mid(chr(i), c + 1, b(j)), 2) / varType(j - 1) +
lstValues.Items(j - 1)
                                parser.AddVariable(lstDesPars.Items(j - 1),
realVal2)
                                If g = noi And i = x Then
                                    MsgBox(realVal2)
                                End If
                                c = c + b(j)
                            Next
                            Try
                                fitness(i) = parser.SimplifyDouble(txtEq.Text)
                            Catch ex As Exception
                                MsgBox("Undeclared variables or not supported
symbols used ")
                                Exit Sub
                            End Try
                        End If
                    Next
                    Dim n As Integer = x
                    While n <> 0 'bubble sort algorithm implementation
                        Dim newn As Integer = 0
```

```
                    If optFlag = 0 Then
                        For i = 2 To n
                            If fitness(i - 1) > fitness(i) Then
                                Dim temp2 As Double = fitness(i - 1)
                                fitness(i - 1) = fitness(i)
                                fitness(i) = temp2
                                Dim temp3 As String = chr(i - 1)
                                chr(i - 1) = chr(i)
                                chr(i) = temp3
                                newn = i
                            End If
                        Next
                    Else
                        For i = 2 To n
                            If fitness(i - 1) < fitness(i) Then
                                Dim temp2 As Double = fitness(i - 1)
                                fitness(i - 1) = fitness(i)
                                fitness(i) = temp2
                                Dim temp3 As String = chr(i - 1)
                                chr(i - 1) = chr(i)
                                chr(i) = temp3
                                newn = i
                            End If
                        Next
                    End If
                    n = newn
                End While
                fitRes(g) = fitness(x) 'fittness results is gonna be
presented on an excel file
                Dim selno As Integer = Int(TextBox2.Text / 100 * x) 'selno
stands for selected number
                If (x - selno) Mod 2 <> 0 Then 'making sure the number of
the non-selected chromosomes is an even number
                    selno = selno + 1
                End If
                Dim selchr(selno) As String 'selected chromosomes matrix
                Dim u As Integer = 1
                For i = x - selno + 1 To x
                    selchr(u) = chr(i)
                    u = u + 1
                Next
                Dim cross(x - selno) As String
                For i = 1 To x - selno
```

```
                    cross(i) = chr(i)
                Next
                For i = 1 To x - selno - 1 Step 2 'breeding is being done
                    Dim flag1 As Integer = 0
                    While flag1 = 0
                        Randomize()
                        Dim crp As Integer = CInt(Math.Floor((bandwith - 1)
* Rnd())) + 1
                        Dim tempstr As String = cross(i)
                        Dim c1 As String = cross(i)
                        Dim c2 As String = cross(i + 1)
                        c1 = Mid(cross(i), 1, crp) & Mid(cross(i + 1), crp +
1)
                        c2 = Mid(cross(i + 1), 1, crp) & Mid(tempstr, crp +
1)
                        Dim c As String = b(1)
                        Dim tv(varCount) As Integer 'tv stands for test
values
                        tv(1) = Convert.ToInt32(Mid(c1, 1, c), 2)
                        For j = 2 To varCount
                            tv(j) = Convert.ToInt32(Mid(c1, c + 1, b(j)), 2)
                            c = c + b(j)
                        Next
                        c = b(1)
                        Dim tv2(varCount) As Integer
                        tv2(1) = Convert.ToInt32(Mid(c2, 1, c), 2)
                        For j = 2 To varCount
                            tv2(j) = Convert.ToInt32(Mid(c2, c + 1, b(j)),
2)
                            c = c + b(j)
                        Next
                        For j = 1 To varCount
                            If tv(j) >= 0 And tv(j) <= nov(j) And tv2(j) >=
0 And tv2(j) <= nov(j) Then
                                flag1 = 1
                                If j = varCount Then
                                    cross(i) = c1
                                    cross(i + 1) = c2
                                End If
                            Else
                                flag1 = 0
                                Exit For
                            End If
```

```
                        Next
                    End While
                Next
                For i = 1 To x - selno
                    chr(i) = cross(i)
                Next
                For i = 1 To 2
                    Dim flag2 As Integer = 0
                    While flag2 = 0
                        Randomize()
                        Dim mutChr As Integer = CInt(Math.Floor((x - 1) *
Rnd())) + 1
                        Dim mutBit As Integer = CInt(Math.Floor((bandwith -
1) * Rnd())) + 1
                        Dim str As String = chr(mutChr)
                        Dim cr As String = chr(mutChr)
                        If str(mutBit) = "0" Then
                            cr = str.Remove(mutBit, 1).Insert(mutBit, "1")
                        Else
                            cr = str.Remove(mutBit, 1).Insert(mutBit, "0")
                        End If
                        Dim c As String = b(1)
                        Dim tv(varCount) As Integer 'tv stands for test
values
                        tv(1) = Convert.ToInt32(Mid(cr, 1, c), 2)
                        For j = 2 To varCount
                            tv(j) = Convert.ToInt32(Mid(cr, c + 1, b(j)), 2)
                            c = c + b(j)
                        Next
                        For j = 1 To varCount
                            If tv(j) >= 0 And tv(j) <= nov(j) Then
                                flag2 = 1
                                If j = varCount Then
                                    chr(mutChr) = cr
                                End If
                            Else
                                flag2 = 0
                                Exit For
                            End If
                        Next
                    End While
                Next
            Next
```

```
                Dim file As System.IO.StreamWriter
                file =
My.Computer.FileSystem.OpenTextFileWriter("c:\Users\Aggelos\Desktop\results.txt"
, True)
                For i = 1 To noi
                    file.WriteLine(fitRes(i))
                Next
                file.Close()
                MsgBox("Optimization Procedure has been concluded.")
            Else
                MsgBox("You need to declare all parameters")
                Exit Sub
            End If
        End If
    End Sub
    Private Sub Button1_Click_1(sender As Object, e As EventArgs) Handles
cmdClose.Click
        End
    End Sub
    Private Sub Button2_Click(sender As Object, e As EventArgs)
    End Sub
    Private Sub Button2_Click_1(sender As Object, e As EventArgs) Handles
cmdDelValue2.Click
        If lstValues2.Items.Count = 0 Or lstValues2.SelectedIndex = -1 Then
            MsgBox("There is nothing here to delete")
        Else
            Dim pos As Integer = lstValues2.SelectedIndex
            lstValues2.Items.RemoveAt(pos)
            If pos = 0 Then
                Label16.Text = ""
            ElseIf pos = 1 Then
                Label17.Text = ""
            ElseIf pos = 2 Then
                Label18.Text = ""
            ElseIf pos = 3 Then
                Label19.Text = ""
            ElseIf pos = 4 Then
                Label20.Text = ""
            ElseIf pos = 5 Then
                Label21.Text = ""
            End If
        End If
    End Sub
```

```
    Private Sub Form1_Load(sender As Object, e As EventArgs) Handles MyBase.Load
    End Sub
    Private Sub Button3_Click_1(sender As Object, e As EventArgs)
    End Sub
    Private Sub Button4_Click_1(sender As Object, e As EventArgs) Handles
cmdClear.Click
        txtDesParName.Text = ""
        txtEq.Text = ""
        txtValueEntry.Text = ""
        txtValueEntry2.Text = ""
        TextBox1.Text = ""
        TextBox2.Text = ""
        TextBox4.Text = ""
        lstDesPars.Items.Clear()
        lstValues.Items.Clear()
        lstValues2.Items.Clear()
        Label16.Text = ""
        Label17.Text = ""
        Label18.Text = ""
        Label19.Text = ""
        Label20.Text = ""
        Label21.Text = ""
        CheckBox1.CheckState = 0
        CheckBox2.CheckState = 0
        CheckBox3.CheckState = 0
        varCount = 0
        labelCount = 0
    End Sub
    Private Sub CheckedListBox1_SelectedIndexChanged(sender As Object, e As
EventArgs)
    End Sub
    Private Sub txtEq_TextChanged(sender As Object, e As EventArgs) Handles
txtEq.TextChanged
        If txtEq.Focused = True And flagMenu = 0 Then
            MsgBox("Supported Symbols" & vbCrLf & "" & vbCrLf & "() Parenthesis"
& vbCrLf &
                   "+ Add symbol" & vbCrLf &
            "- Subtract Symbol" & vbCrLf & "* Multiplication Symbol" & vbCrLf &
"/ Divide Symbol" _
             & vbCrLf & "% Modulus Symbol" & vbCrLf & " ^ Exponent Symbol" &
vbCrLf &
             "ABS() absolute of a number" & vbCrLf & "SIN() sine of a number" &
vbCrLf &
```

```
            "COS() cosine of a number" & vbCrLf & "TAN() tangent of a number" &
vbCrLf &
            "LOG base 10 logarithm of a number" & vbCrLf & "LOGN natural
logarithm of a number")
            flagMenu = 1
        End If
    End Sub
    Private Sub TextBox3_TextChanged(sender As Object, e As EventArgs) Handles
TextBox3.TextChanged
    End Sub
    Private Sub TextBox5_TextChanged(sender As Object, e As EventArgs) Handles
TextBox5.TextChanged
    End Sub
End Class
```

ΠΑΡΑΡΤΗΜΑ Β

Τρόπος Λειτουργίας Χρησιμοποιούμενου Parser

Για να χρησιμοποιηθεί ο parser [22] και οι built-in συναρτήσεις του πρέπει ο χρήστης να τον εντάξει με κάποιο τρόπο στο προγραμματιστικό περιβάλλον που χρησιμοποιεί. Στο Visual Studio, ο χρήστης πρέπει να πατήσει δεξί κλίκ στο πεδίο “References” το οποίο είναι τμήμα του ευρύτερου πεδίου “Solution Explorer”, ύστερα να πατήσει πάνω στην επιλογή “Add Reference” και να εισάγει το path που καταλήγει στο .dll αρχείο του parser. Στην περίπτωση του χρησιμοποιούμενου parser το αρχείο αυτό ονομάζεται “MathParserNet.dll” . Επιλέγοντάς το και πατώντας “Ok” το Visual Studio” αναγνωρίζει τον parser και ο χρήστης μπορεί να τον χρησιμοποιήσει γράφοντας σχετικό κώδικα σύμφωνα με το documentation του.

Ο συγκεκριμένος parser μπορεί να υπολογίσει εκφράσεις με κάποιες συναρτήσεις του όπως οι “SimplifyInt()” και “SimplifyDouble()” οι οποίες δίνουν αντίστοιχα ακέραιο και δεκαδικό αριθμό ως αποτέλεσμα. Μέσα στις παρανθέσεις μπαίνουν οι μαθηματικές εκφράσεις που ο χρήστης θέλει να υπολογίσει, και μπορεί να μπει κείμενο με αριθμούς ή να δηλωθούν αρχικά κάποιες μεταβλητές ώστε να αναγνωρίζονται από τον parser στο εισαγόμενο κείμενο προς υπολογισμό. Η δήλωση μεταβλητών στον parser γίνεται με τη συνάρτηση “AddVariable(variable name,value)” και εφόσον γίνει, για να αλλάξουν οι τιμές των μεταβλητών πρέπει να γίνει εκ νέου δήλωση, αφού πρώτα με τη συνάρτηση “RemoveAllVariables()” αφαιρεθούν τα ονόματα και οι τιμές των μεταβλητών. Ο parser υποστηρίζει και άλλες λειτουργίες, όπως εισαγωγή custom συνάρτησης και άλλα τα οποία όμως δε χρησιμοποιούνται στην εν λόγω εργασία. Παραθέτουμε εδώ τον κατάλογο με τα υποστηριζόμενα από τον parser μαθηματικά σύμβολα και τη σημασία τους

- () -- Parenthesis
- + -- Add symbol (3 + 2)
- - -- Subtract symbol (3 - 2)
- * -- Multiplication symbol (3 * 2)
- / -- Divide symbol (3 / 2)
- % -- Modulus symbol (3 % 2)(divides the two numbers, but returns the remainder)
- ^ -- Exponent symbol (3 ^ 2)(squares 3)
- ABS -- Function returns the absolute of a number (ABS(-3))
- SIN -- Returns the sine of a number (SIN(3.14))
- COS -- Returns the cosine of a number (COS(3.14))
- TAN -- Returns the tangent of a number (TAN(3.14))
- LOG -- Returns the base 10 logarithm of a number
- LOGN -- Returns the natural logarithm of a number

(Τελευταία σελίδα της εργασίας)