ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ
ΤΜΗΜΑ ΜΗΧΑΝΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ, ΤΟΜΕΑΣ ΡΕΥΣΤΩΝ

ΕΡΓΑΣΤΗΡΙΟ ΘΕΡΜΙΚΩΝ ΣΤΡΟΒΙΛΟΜΗΧΑΝΩΝ
ΜΟΝΑΔΑ ΠΑΡΑΛΛΗΛΗΣ ΥΠΟΛΟΓΙΣΤΙΚΗΣ
ΡΕΥΣΤΟΔΥΝΑΜΙΚΗΣ ΚΑΙ ΒΕΛΤΙΣΤΟΠΟΙΗΣΗΣ

Αναθαθμισμένη Παραλλαγή της Τεχνικής Σμήνους Σωματιδίων στη Βελτιστοποίηση

Διπλωματική Εργασία

Δημήτρης Χ. Σωφρονίου

Επίβλεψη: Καθηγητής Ε.Μ.Π. **Κ. Χ. Γιαννάκογλου**

Αθήνα, Μάρτιος 2011

NATIONAL TECHNICAL UNIVERSITY OF ATHENS
SCHOOL OF MECHANICAL ENGINEERING

LAB OF THERMAL TURBOMACHINES
PARALLEL CFD & OPTIMIZATION UNIT

Enhanced Variant of the Particle Swarm Method in Optimization

Diploma Thesis

Dimitris C. Sofroniou

Advisor: **K.C. Giannakoglou**, Professor NTUA

Athens, March 2011

Ευχαριστίες…

Οφείλω κατ' αρχήν να εκφράσω την ειλικρινή μου ευγνωμοσύνη προς τον Καθηγητή κ. Κ.Χ. Γιαννάκογλου για την υποστήριξη και την υπομονή, ιδιαίτερα κατά την εκπόνηση του παρόντος, και την εμπιστοσύνη και προσωπική μέριμνα προς το πρόσωπό μου τα τελευταία τρία χρόνια. Η συνολική εμπειρία των σπουδών μου θα ήταν τελείως διαφορετική αν απουσίαζε ο ίδιος και η δουλειά του. Εκ της ομάδας του Εργαστηρίου Θερμικών Στροβιλομηχανών, νιώθω υπόχρεος προς τη διδάκτορα Βαρβάρα Ασούτη και την υποψήφια διδάκτορα Ευγενία Κοντολέοντος για το χρόνο που μου διέθεσαν, παρέχοντάς μου συνεχή τεχνική υποστήριξη και καθοδήγηση. Τις ευχαριστώ για την καλοσύνη και την προθυμία να βοηθήσουν, πέρα από κάθε τυπική υποχρέωση, όπως και όλους τους άλλους φίλους από την ερευνητική ομάδα.

Καθότι η ολοκλήρωση της διπλωματικής μου εργασίας συμπίπτει με την ολοκλήρωση των σπουδών μου στο τμήμα Μηχανολόγων Μηχανικών του Εθνικού Μετσοβίου Πολυτεχνείου, θα ήθελα να ευχαριστήσω και όσους, φίλους και συναδέλφους, με τον ένα ή τον άλλο τρόπο, βοήθησαν στην περάτωση των υποχρεώσεών μου ως σπουδαστή.

Τέλος, ευχαριστώ τους δικούς μου ανθρώπους για την αγάπη και τη στήριξή τους. Ιδιαίτερα δε, τους γονείς μου, Χριστόφορο και Παναγιώτα για τις όποιες προσωπικές θυσίες μου εξασφάλισαν το προνόμιο να πραγματοποιήσω αυτές τις σπουδές.

Διπλωματική εργασία Δημήτρη Χ. Σωφρονίου
υπό την επίβλεψη του Καθηγητού Ε.Μ.Π. Κ. Χ. Γιαννάκογλου

Εργαστήριο Θερμικών Στροβιλομηχανών
Μονάδα Παράλληλης Υπολογιστικής Ρευστοδυναμικής &
Βελτιστοποίησης

Αναβαθμισμένη Παραλλαγή της Τεχνικής Σμήνους Σωματιδίων στη Βελτιστοποίηση

Στόχο της παρούσης εργασίας αποτέλεσε η ανάπτυξη ενός εναλλακτικού αλγορίθμου Στοχαστικής Βελτιστοποίησης βασιζομένου στην εδραιωμένη μέθοδο Σμήνους Σωματιδίων (Particle Swarm Optimization – PSO). Η προϋπάρχουσα κεντρική ιδέα πλαισιώνεται από δοκιμασμένες στο χώρο της βελτιστοποίησης λύσεις και από ορισμένες προσθήκες του γράφοντος, με στόχο το τελικό αποτέλεσμα να αποτελέσει μία λειτουργική και ανταγωνιστική εναλλακτική λύση, ειδικά όσον αφορά προβλήματα βελτιστοποίησης ως προς περισσότερα του ενός κριτηρίων. Παρουσιάζεται διεξοδικά η πρόοδος του αλγορίθμου με κάθε προσθήκη, ενώ εκτίθενται παράλληλα και διάφορα αντιπροσωπευτικά της αιχμής του δόρατος της Στοχαστικής Βελτιστοποίησης παραδείγματα, για λόγους σύγκρισης αλλά και πληρότητας. Ιδιαίτερη μνεία γίνεται στους δημοφιλείς Εξελικτικούς Αλγορίθμους (ΕΑ), το αντίπαλο δέος, ουσιαστικά, της τεχνικής Σμήνους Σωματιδίων, επί των οποίων το Εργαστήριο Θερμικών Στροβιλομηχανών (ΕΘΣ) έχει να επιδείξει σπουδαία δραστηριότητα και τεχνογνωσία και ο γράφων μία σχετική εμπειρία. Επιχειρείται μία απευθείας αντιπαραβολή, τόσο φιλοσοφίας όσο και πρακτικής, των δύο ιδεών, ενώ είναι σαφής καθ' όλη την έκταση αυτού του εκπονήματος η πρόθεση αντιστοίχισης, τμηματικά, της μίας με την άλλη, ώστε να τονιστεί ο ενιαίος χαρακτήρας του χώρου της Στοχαστικής Βελτιστοποίησης και να ταυτοποιηθούν τα πάγια χαρακτηριστικά των μεθόδων αυτών.

Ο ολοκληρωμένος αλγόριθμος δοκιμάζεται, κατόπιν, σε επιλεγμένες εφαρμογές, ακαδημαϊκού και βιομηχανικού ενδιαφέροντος, όλες δύο στόχων: Οι περιπτώσεις των μαθηματικών συναρτήσεων ZDT-1 και ZDT-3 είναι αντιπροσωπευτικά δείγματα προβλημάτων που έχουν αναπτυχθεί από ακαδημαϊκό φορέα ειδικά ως μέσο δοκιμής και σύγκρισης τέτοιων μεθόδων και αποτελούν στην ουσία μοντελοποίηση των δυσχερειών που αναμένεται να συναντήσει ένας αλγόριθμος βελτιστοποίησης σε βιομηχανικές εφαρμογές. Τέλος, δοκιμάζεται και έναντι της, υπό περιορισμούς, αεροδυναμικής βελτιστοποίησης πτερυγίου αεροσυμπιεστή. Τα αποτελέσματα που εκτίθενται προσφέρονται για σύγκριση της παρούσας πρότασης με έναν ενδεικτικό ΕΑ, επιβεβαιώνουν τα όσα είναι γνωστά για τις διαφορές στη συμπεριφορά των δύο τεχνικών, ενώ πιστοποιούν την ανταγωνιστικότητα του παρουσιαζόμενου λογισμικού.

Πίνακας περιεχομένων

1. Πρόλογος

(Το κυρίως σώμα της διπλωματικής αυτής εργασίας είναι γραμμένο στην Αγγλική γλώσσα. Το κείμενο που ακολουθεί αποτελεί μια ιδιαιτέρως εκτενή περίληψη αυτής, όπου γίνεται μάλιστα επίκληση σχημάτων και μαθηματικών τύπων απ΄ το ξενόγλωσσο τμήμα, με αποτέλεσμα να δίνει μια αρκετά πλήρη εικόνα του περιεχομένου της διπλωματικής εργασίας. Η δομή του ελληνικού κειμένου ακολουθεί αυστηρά τη δομή σε κεφάλαια του λεπτομερούς ξενόγλωσσου κειμένου.)

Η παρούσα διπλωματική εργασία στοχεύει στην δημιουργία και δοκιμή μίας αναβαθμισμένης παραλλαγής της τεχνικής βελτιστοποίησης που βασίζεται στη μέθοδο Σμήνους Σωματιδίων (ΒΣΣ), (Particle Swarm Optimization - PSO) [1]. Ο προτεινόμενος αλγόριθμος (ΠΑ) δανείζεται την προϋπάρχουσα βασική ιδέα της ΒΣΣ, επί της οποίας γίνονται στη συνέχεια προσθήκες και βελτιώσεις, με σκοπό το τελικό αποτέλεσμα να αποτελεί ένα, κατά το δυνατόν στο πλαίσιο διπλωματικής εργασίας, ανταγωνιστικό και πλήρες εργαλείο βελτιστοποίησης, για χρήση σε κάθε είδους εφαρμογές.

Το θεωρητικό υπόβαθρο του παρόντος εκτείνεται πέρα από τα στενά όρια της τεχνικής ΒΣΣ: παρουσιάζεται εκτενώς ο ευρύτερος χώρος της Στοχαστικής Βελτιστοποίησης (ΣΒ), υποπεριοχή του οποίου αποτελεί, άλλωστε, η συγκεκριμένη τεχνική και οι διάφορες παραλλαγές της. Ιδιαιτέρως εμμένουμε στους λεγόμενους Εξελικτικούς Αλγορίθμους (Evolutionary Algorithms - ΕΑ) [45, 46], καθότι αποτελούν την πιο ευρέως εφαρμοσμένη και δημοφιλή μέθοδο ΣΒ ως σήμερα και είναι ενδεικτικοί της φιλοσοφίας και της πρακτικής που διέπουν τη ΣΒ. Η έντονη δραστηριότητα του Εργαστηρίου Θερμικών Στροβιλομηχανών (ΕΘΣ) στη χρήση και κυρίως στην ανάπτυξη λογισμικού βασισμένου στους ΕΑ απετέλεσε το έναυσμα για την εκπόνηση της εργασιας αυτής, ώστε να καταστεί δυνατή η αντιπαραβολή των δύο μεθόδων. Το πλήθος σχετικών δημοσιεύσεων προδίδει το ενδιαφέρον από ακαδημαϊκούς και βιομηχανικούς κύκλους να επενδύσουν στη ΒΣΣ, όπως ήδη κάνουν με τους ΕΑ. Παρουσιάζει, συνεπώς, έντονο ενδιαφέρον και η παράλληλη εξέταση των δύο τεχνικών, για να διαπιστωθούν τυχόν ομοιότητες ή διαφορές στη δομή και τη συμπεριφορά και να εντοπισθούν τα ειδοποιά στοιχεία της καθεμιάς, τα οποία, μάλιστα, όπως θα δειχθεί, παρουσιάζουν ευθεία αντιστοιχία μεταξύ τους. Απώτερος στόχος αυτής της θεωρητικής διερεύνησης είναι, μελετώντας αυτούς τους δύο χαρακτηριστικούς εκπροσώπους τους, να καταδείξουμε τον ενιαίο χαρακτήρα όλων των Στοχαστικών Μεθόδων. Επιπλέον, προκύπτουν συμπεράσματα για το πώς η μία ή η άλλη μπορούν να αναβαθμιστούν, ανταλλάσσοντας χαρακτηριστικά.

Ο αλγόριθμος που αναπτύχθηκε προσανατολίζεται στη επίλυση προβλημάτων Πολυ-Κριτηριακής Βελτιστοποίησης (ΠΚΒ), δηλαδή προβλημάτων όπου επιδιώκεται η

βελτιστότητα του αποτελέσματος ως προς άνω του ενός κριτηρίων. Η ΠΚΒ είναι ένα ξεχωριστό κεφάλαιο της βελτιστοποίησης, με τις δικές του ιδιαιτερότητες και πρακτικές, γι αυτό και της γίνεται ιδιαίτερη μνεία σε σχετική παράγραφο της εργασίας. Η επέκταση ενός αλγορίθμου βελτιστοποίησης ώστε να ανταποκριθεί σε εφαρμογές ΠΚΒ δεν είναι απλή υπόθεση, αντίθετα είναι εξίσου απαιτητική με την ανάπτυξη του ίδιου του αλγορίθμου. Ενδεικτικό είναι, ότι το πώς θα αντιμετωπιστούν τα ζητήματα χειρισμού των υποψηφίων λύσεων σε περιπτώσεις ΠΚΒ απασχόλησε ίσως περισσότερο απ' ό,τι όλα τα υπόλοιπα μέρη του ΠΑ.

Η τεχνική ΒΣΣ ανήκει στην οικογένεια μαθηματικών μοντέλων που είναι γνωστή ως Νοημοσύνη Σμήνους (ΝΣ) [28, 29], που απαρτίζεται από πρακτικές που μιμούνται τη συλλογική συμπεριφορά ενός συνόλου εμβίων όντων, πουλιών, ψαριών, εντόμων κλπ. Γίνεται μια εκτενής επισκόπησή της ΝΣ και κατόπιν επικεντρωνόμαστε σε αυτή καθεαυτή τη ΒΣΣ: παρουσιάζεται το μαθηματικό της υπόβαθρο, όπως αυτό διαμορφώθηκε ύστερα από αρκετές προσθήκες και αλλαγές την τελευταία δεκαετία, με έμφαση στις ρυθμιστικές παραμέτρους που το διέπουν. Ακολούθως, εξετάζεται η επιρροή αυτών των παραμέτρων στη συμπεριφορά της ΒΣΣ κατά την αναζήτηση των βέλτιστων λύσεων. Χωρίς να διεξάγεται κάποια παραμετρική μελέτη ή πείραμα, με πλήθος αναφορών στη σχετική βιβλιογραφία [11, 12] και επίκληση της υπάρχουσας τεχνογνωσίας, τεκμηριώνονται οι επιλογές που έγιναν στο προτεινόμενο λογισμικό, όσον αφορά την τιμή και διακύμανση των ρυθμιστικών αυτών παραμέτρων.

Στο 4^{ο} κεφάλαιο, για πρώτη φορά γίνεται αναφορά στα νέα στοιχεία που είναι παρόντα στον ΠΑ. Ανά ενότητα παρουσιάζονται μία-μία οι κομβικότερες λειτουργίες του, με σύντομη παράθεση των επικρατέστερων στο χώρο αντίστοιχων πρακτικών, ανά περίπτωση. Σε κάποιες περιπτώσεις, εκτίθενται διάφορες εναλλακτικές που δοκιμάστηκαν και εξυπηρετούν τον ίδιο σκοπό και εξηγείται πως επιδρούν στο τελικό αποτέλεσμα. Πρέπει να τονιστεί ότι ο ΠΑ δημιουργήθηκε και προγραμματίστηκε εκ του μηδενός, με εξαίρεση φυσικά τον προϋπάρχοντα πυρήνα της ΒΣΣ. Ασφαλώς, σε κάποια σημεία, η ακολουθούμενη οδός μοιάζει με προϋπάρχουσες συνήθεις πρακτικές που κατονομάζονται, διαφέροντας στα σημεία μόνο από αυτές. Γενικά, όμως, ο ΠΑ αποκλίνει αρκετά από την πεπατημένη.

Τέλος, ο ΠΑ δοκιμάστηκε σε τρία προβλήματα, το καθένα με τις ιδιαιτερότητές του. Τα δυο πρώτα αποτελούν μαθηματικά προβλήματα ελαχιστοποίησης δύο συναρτήσεων στόχων και έχουν αναπτυχθεί ακριβώς γι αυτό το σκοπό, να αξιολογούν αλγορίθμους βελτιστοποίησης. Το πρώτο είναι σχετικά απλό και χρησίμευσε κυρίως για να πιστοποιήσει την καλή λειτουργία του ΠΑ. Το δεύτερο είναι απαιτητικότερο και πολύ διαδεδομένο στην ειδική βιβλιογραφία. Ο ΠΑ δοκιμάστηκε ακόμα σε ένα πρακτικότερης υφής πρόβλημα: χρησιμοποιήθηκε για την αναζήτηση του βέλτιστου περιγράμματος

αεροτομής του πτερυγίου στάτορα ενός αξονικού συμπιεστή, υπό αυστηρούς περιορισμούς και με κριτήρια την καλή αεροδυναμική απόδοση του μεμονωμένου πτερυγίου αλλά και την καλύτερη δυνατή λειτουργία του ως συνιστώσας του συμπιεστή.

Τα πολύ ικανοποιητικά αποτελέσματα των δοκιμών επιβεβαιώνουν την ανταγωνιστικότητα του ΠΑ και την καταλληλότητά του ως βάσης για πειραματισμούς και προσθήκες, με σκοπό τη βελτίωση των επιδόσεων και της πληρότητάς του. Τέτοιες προτάσεις για μελλοντικές επεμβάσεις γίνονται στην καταληκτική παράγραφο του κειμένου της διπλωματικής εργασίας.

2. Στοχαστική Βελτιστοποίηση - Πολυ-κριτηριακή Βελτιστοποίηση

2.1. Στοχαστικές Μέθοδοι

Οι μέθοδοι βελτιστοποίησης διακρίνονται σε Στοχαστικές (ΣΜ) και Αιτιοκρατικές [40, 45]. Η ΒΣΣ, αλλά και η οικογένεια των Εξελικτικών Μεθόδων, τις οποίες θα εξετάσουμε παρακάτω, συγκαταλέγονται στις πρώτες. Ο χαρακτηρισμός «στοχαστικές» τους αποδίδεται λόγω της τυχαιότητας που διέπει πολλές από τις λειτουργίες που συνθέτουν μια τέτοια μέθοδο: η φιλοσοφία τους βασίζεται κατά ένα βαθμό στην «περιπλάνηση» εντός του Ν-διάστατου χώρου (όπου Ν το πλήθος των μεταβλητών του προβλήματος) σε αναζήτηση των σημείων όπου το πρόβλημα βρίσκει ικανοποιητική, ει δυνατόν βέλτιστη, λύση. Πρακτικά, συμπληρώνουν αυτή την τυχαία αναζήτηση με ειδικούς μηχανισμούς που εκμεταλλεύονται την μέχρι στιγμής συγκεντρωμένη εμπειρία για τα χαρακτηριστικά του χώρου αναζήτησης, προκειμένου να επισπεύσουν την εύρεση των βελτίστων, κατευθύνοντας κατάλληλα την όλη διαδικασία και περιορίζοντας την τυχαιότητα. Αυτοί οι μηχανισμοί είναι συνήθως εμπνευσμένοι από τη φύση (εξέλιξη των ειδών - ΕΑ, συντονισμός αγέλης/σμήνους - ΒΣΣ). Μία τυπική -πληθυσμιακή- ΣΜ διαχειρίζεται ταυτόχρονα ένα πληθυσμό από υποψήφιες λύσεις που θα βρεθούν διαδοχικά σε διάφορες θέσεις στο χώρο, επιλεγμένες εντελώς τυχαία στην αρχή και υποδεικνυόμενες αργότερα.

Οι αιτιοκρατικές μέθοδοι, από την άλλη, προσεγγίζουν τις βέλτιστες λύσεις με απολύτως δομημένο και στοχευμένο τρόπο, ακολουθώντας την κλίση της συνάρτησης που εκφράζει το στόχο-κριτήριο της βελτιστοποίησης. Όσο αυτή η κλίση, στην ουσία η παράγωγος της συνάρτησης-στόχου, τείνει στο μηδέν, τόσο πιο κοντά βρισκόμαστε σε ακρότατο της συνάρτησης. Αν αυτό το ακρότατο είναι ολικό, συμπίπτει με τη βέλτιστη

λύση. Οι αιτιοκρατικές μέθοδοι μπορούν πολύ γρήγορα να οδηγηθούν σε βέλτιστη λύση, εφόσον όμως δεν «παγιδευτούν» σε τοπικό ακρότατο. Επίσης, η εφαρμογή τους είναι αδύνατη αν δε μπορούμε να υπολογίσουμε την παράγωγο της συνάρτησης στόχου παντού ή σχεδόν παντού στο πεδίο των λύσεων που θα εξεταστεί.

Μια ΣΜ δεν έχει καμία τέτοια εξάρτηση από τη δυνατότητα εύρεσης της παραγώγου, που μπορεί να είναι και εξαιρετικά δαπανηρή, ούτε κινδυνεύει τόσο από οριστική παγίδευση σε τοπικό ακρότατο. Στην πραγματικότητα, δε χρειάζεται να γνωρίζουμε το παραμικρό για το πρόβλημα, πέραν από τις μεταβλητές του, προκειμένου να την εφαρμόσουμε. Είναι λοιπόν εύκολα προσαρμόσιμη σε κάθε πρόβλημα, έστω κι αν γενικά καθυστερεί περισσότερο να βρει ικανοποιητική λύση.

2.2. Εξελικτικοί Αλγόριθμοι

Η ιδέα των ΕΑ μετρά ήδη τέσσερις δεκαετίες ζωής και είναι εμπνευσμένη από τη Δαρβινιστική αντίληψη της Εξέλιξης των ειδών και της πάλης αυτών για επιβίωση. Οι υποψήφιες λύσεις που απαρτίζουν τον πληθυσμό ενός ΕΑ, αντιπροσωπεύονται από γονιδιώματα, ενδεχομένως δυαδικά κωδικοποιημένα, που εκφράζουν τον συνδυασμό τιμών μεταβλητών που δίνει την κάθε λύση. Ο ΕΑ εφαρμόζει στα γονιδιώματα αυτά ένα σύνολο από τελεστές, που με τη σειρά τους είναι εμπνευσμένοι από τις διάφορες διαδικασίες στις οποίες συνίσταται η Εξέλιξη: Διασταύρωση/Αναπαραγωγή, Φυσική Επιλογή και Μετάλλαξη. Επειδή όμως δεν είναι απολύτως σαφής ο τρόπος που τα παραπάνω συνδυάζονται, και εν τέλει δεν επαρκούν για να μοντελοποιήσουν με απόλυτη ακρίβεια τη φυσική διεργασία που οδηγεί στη βελτίωση των χαρακτηριστικών ενός είδους, υπεισέρχεται και η τυχαιότητα.

Ένας ΕΑ προσπαθεί να βρει λύση σε ένα πρόβλημα προσαρμόζοντας την αναζήτησή της, με την ίδια λογική που ένα έμβιο είδος προσαρμόζει τα χαρακτηριστικά του στις απαιτήσεις του περιβάλλοντός του, προκειμένου να επιβιώσει. Δεν καταφέρνουν όμως όλα τα είδη να επιβιώσουν, παρά μόνο αυτά που αποδεικνύονται *ικανότερα* να προσαρμοστούν, έτσι και ο ΕΑ δεν επενδύει σε όλα τα γονιδιώματα που απαρτίζουν τον πληθυσμό, αλλά απορρίπτει κάποια και γεννά καινούρια, προϊόντα της εξελικτικής διαδικασίας. Αυτή η εναλλαγή λαμβάνει χώρα κάθε γενιά-επανάληψη του αλγορίθμου. Στόχος είναι να επιβιώνουν πάντα τα προσαρμοστικότερα είδη - οι καλύτερες λύσεις - και με συνεχή διασταύρωση και μετάλλαξη αυτών να επιτυγχάνουμε την ολοένα καλύτερη προσαρμογή - λύση. Παρεμπιπτόντως, η μετάλλαξη είναι ο κύριος εκφραστής της στοχαστικότητας στον ΕΑ: στην απλούστερή της μορφή, συνίσταται στην απλή μεταβολή του 1 σε 0 ή τούμπαλιν, ενός bit του γονιδιώματος-χρωμοσώματος, με όποια ακαθόριστη επίπτωση έχει αυτό στις ιδιότητες του γονιδιώματος.

Ένας τυπικός ΕΑ ακολουθεί την εξής ροή, όπως απεικονίζεται και στο σχήμα 2.1: αφού αρχικοποιηθούν τυχαία **λ** το πλήθος υποψήφιες λύσεις, εφαρμόζονται επαναληπτικά τα ακόλουθα βήματα, έως ότου ικανοποιηθεί κάποιο κριτήριο τερματισμού του αλγορίθμου:

- **Αξιολογούνται** οι **λ** λύσεις από το όποιο εξειδικευμένο λογισμικό υπολογισμού των συναρτήσεων-στόχων είναι διαθέσιμο, εξάγεται δηλαδή για την καθεμιά η αντίστοιχη τιμή της *αντικειμενικής* συνάρτησης.

- Οι καλύτερες εξ αυτών εισέρχονται στους **Επίλεκτους**, όπου, ανά πάσα στιγμή, συγκαταλέγονται οι **e** το πλήθος καλύτερες, μέχρι στιγμής, λύσεις.

- Επιλέγονται από τον τρέχοντα πληθυσμό οι **μ** το πλήθος **γονείς**, από **διασταύρωση** των οποίων θα προκύψει η επόμενη γενεά πλήθους **λ**, οι **απόγονοι** (ενδέχεται μ=λ). Η διαδικασία της **επιλογής** λαμβάνει, φυσικά, υπόψη την ποιότητα της κάθε λύσης, βάσει τιμής αντικειμενικής συνάρτησης, αλλά διέπεται και από κάποια τυχαιότητα.

- Πραγματοποιείται η **διασταύρωση**, μια διαδικασία που παράγει νέες υποψήφιες λύσεις συνθέτοντας τμήματα του γονιδιώματος καθενός εκ των γονέων.

- Η νέα γενιά τελικώς διαμορφώνεται με την προσθήκη προϊόντων **μετάλλαξης,** αφού αυτή εφαρμοστεί σε τυχαία επιλεγέντα μέλη του πληθυσμού, και **ελιτισμού,** ο οποίος επιβάλλει αυθαίρετα την επιβίωση και παρουσία ενός η περισσοτέρων επιλέκτων στη νέα γενεά, για να εγγυηθεί στοιχειωδώς την ποιότητα της νέας αυτής γενεάς.

2.3. Πολύ-Κριτηριακή Βελτιστοποίηση - Κατά Pareto Κυριαρχία

Όπως υπογραμμίστηκε στον πρόλογο, ο ΠΑ, παρότι είναι σε θέση να λύσει προβλήματα ενός στόχου, αναπτύχθηκε με γνώμονα την καλή απόδοση έναντι προβλημάτων Πολυ-κριτηριακής Βελτιστοποίησης (ΠΚΒ). Η προσέγγιση των τελευταίων είναι πολύ διαφορετική από αυτή των προβλημάτων ενός μοναδικού στόχου, για δύο κυρίους λόγους:

I. Ένα πρόβλημα ΠΚΒ επιδέχεται γενικά περισσότερες της μιας βέλτιστες λύσεις: Αποδεκτές μπορούν να γίνουν αυτές που έχουν την απόλυτη απόδοση ως προς ένα στόχο μόνο, ασχέτως της επίδοσής τους ως προς τους υπόλοιπους. Υπάρχουν και ενδιάμεσες λύσεις που αποδίδουν ένα αποτέλεσμα περισσότερο ισορροπημένο ως προς τους διάφορους στόχους, με ικανοποιητική συνολική ποιότητα, χωρίς να επιτυγχάνουν το απόλυτο ως προς κάποιον εξ αυτών. Σπάνια

μία και μοναδική λύση θα μπορεί να εξασφαλίσει τη βέλτιστη ως προς όλα τα κριτήρια απόδοση, καθώς αυτά τα κριτήρια είναι, γενικά, αντικρουόμενα. Αν δεν είναι, άλλωστε, το πρόβλημά μας μπορεί να αναχθεί σε ενός στόχου και να αντιμετωπιστεί αναλόγως.

II. Δεν είναι δυνατό να κατατάξουμε με απόλυτο τρόπο το σύνολο των δυνατών λύσεων ως προς την ποιότητά τους, με βάση περισσότερα του ενός κριτηρίων. Από δύο λύσεις που υπερισχύουν η μία της άλλης ως προς έναν από δυο στόχους, ποια είναι η καλύτερη? Παρ' όλα αυτά, θα χρειαστεί να επιλέξουμε μεταξύ δύο η περισσοτέρων λύσεων πολλές φορές, επενδύοντας σε κάποιες και απορρίπτοντας άλλες.

Γίνεται λοιπόν κατανοητό, ότι πρέπει να αναπτυχθούν μηχανισμοί που: **α)** θα κατατάσσουν τις λύσεις με τη μεγαλύτερη δυνατή *αξιοκρατία* (που θα αποφαίνονται για το ποια λύση θα προτιμηθεί έναντι μιας άλλης, εκ πρώτης όψεως ισοδύναμης), λαμβάνοντας υπόψη και τις ιδιότητες του προβλήματος, και **β)** θα ωθούν τον αλγόριθμο να ανακαλύψει λύσεις που θα καλύπτουν ένα ευρύ και ποικιλόμορφο φάσμα διαφορετικών συνδυασμών απόδοσης ως προς τα διάφορα κριτήρια.

Η πρώτη επιδίωξη βρίσκει εν μέρει διέξοδο στην έννοια της *κατά Pareto κυριαρχίας* [23], σύμφωνα με την οποία μία λύση κρίνεται ως *μη κυριαρχούμενη*, όταν καμία άλλη λύση δεν ξεπερνά την επίδοσή της ως προς ΟΛΟΥΣ τους τεθειμένους στόχους ταυτόχρονα. Θεωρούμε δε, ότι μία λύση *κυριαρχεί* επί μίας άλλης όταν είναι καλύτερη αυτής ως προς ένα στόχο τουλάχιστον, ενώ δεν είναι χειρότερη αυτής ως προς κάθε άλλον ξεχωριστά (εξ. 2.1). Το σύνολο των μη κυριαρχούμενων λύσεων που μπορούν να δοθούν σε ένα πρόβλημα διαμορφώνουν το λεγόμενο *Μέτωπο Pareto* και δεν τίθεται θέμα περαιτέρω σύγκρισης μεταξύ των.

Η δεύτερη επιδίωξη μπορεί να επιτευχθεί δια πολλών οδών. Πολλές επεμβάσεις μπορούν να γίνουν σε έναν αλγόριθμο βελτιστοποίησης προκειμένου να τον αποθαρρύνουν από το να επικεντρωθεί σε περιορισμένο εύρος λύσεων. Τη μεγαλύτερη συνεισφορά σε αυτό την έχουν οι ρουτίνες που αναλαμβάνουν και την κατάταξη των υποψηφίων σύμφωνα με την αρχή της κυριαρχίας, όπου μπορεί να ληφθεί επιπλέον υπόψη, δυσμενώς για την εκάστοτε υποψήφια λύση, η ύπαρξη δυσανάλογα πολλών πολύ όμοιων λύσεων.

Η λειτουργία αντιπροσωπευτικών τέτοιων ρουτινών (SPEA 2) παρουσιάζεται εκτενώς στην παράγραφο 2.3.

3. Περί της Μεθόδου Σμήνους Σωματιδίων

3.1. Νοημοσύνη Σμήνους και Τεχνική Σμήνους Σωματιδίων

Ένας καλός ορισμός της Νοημοσύνης Σμήνους (Swarm Intelligence) είναι: «η συλλογική συμπεριφορά αυταρκών μεν, αποκεντρωμένων δε, τεχνητών ή φυσικών συστημάτων». Με απλά λόγια, ΝΣ είναι αυτή η ακαθόριστη δύναμη που νοηματοδοτεί τη συμπεριφορά ενός συνόλου ατόμων κατά την επαφή τους με το περιβάλλον τους αλλά και την αλληλεπίδρασή τους: ενώ δεν υπάρχει κάποια σαφής αρχή που ορίζει πως θα συμπεριφερθεί αυτό το σύνολο, η δράση τους διέπεται από λογική και ομοιογένεια. Ένα καλό παράδειγμα είναι μία αποικία μυρμηγκιών που, χωρίς τις ιδιαίτερα εξεζητημένες διεξόδους επικοινωνίας μεταξύ των μελών της, καταφέρνει να συντονιστεί άψογα και να επιτύχει σπουδαία πράγματα. Οι μέθοδοι ΝΣ είναι μαθηματικά εργαλεία, με υπόβαθρο εμπνευσμένο από τέτοια παραδείγματα (σμήνη ψαριών, εντόμων, πουλιών κλπ.), που εκμεταλλεύονται αυτή τη νοήμονα δράση για να επιλύσουν προβλήματα.

Η τεχνική Σμήνους Σωματιδίων [1], τώρα, η οποία προέκυψε αρχικά ως εργαλείο μελέτης κοινωνικής συμπεριφοράς και μετεξελίχτηκε σε μέθοδο βελτιστοποίησης, μιμείται τη συμπεριφορά ενός σμήνους πουλιών εν πτήσει κατά την αναζήτηση τροφής. Αποκωδικοποιώντας ένα σύνολο επιμέρους δράσεων, όπως π.χ. η αποφυγή σύγκρουσης μεταξύ τους αλλά και η προσπάθεια να μην απομακρυνθούν από το σμήνος και αποδίδοντάς τις στα άτομα μιας πληθυσμιακής στοχαστικής μεθόδου καταλήγουμε στη ΒΣΣ. Όπως και τα πουλιά, τα μέλη του πληθυσμού μιας τέτοιας μεθόδου αναζητούν λύση στο πρόβλημα βασιζόμενα τόσο στην προσωπική τους αντίληψη του χώρου αναζήτησης, όσο και στη συλλογική πρόοδο του σμήνους. Αν τυχόν, δηλαδή, κάποιο άτομο δείχνει να τα πηγαίνει ιδιαίτερα καλά, έχοντας ανακαλύψει κάποια πολλά υποσχόμενη λύση ή περιοχή λύσεων, όλο το σμήνος θα συγκλίνει προς το μέρος του, διατηρώντας την ατομική του εγρήγορση.

$$\vec{X}_{i,k+1} = \vec{X}_{i,k} + \vec{V}_{i,k+1} \qquad \text{(eq. 3.1)}$$

$$\vec{V}_{i,k+1} = W \cdot \vec{V}_{i,k} + C_{cogn} \cdot R_{cogn}(\vec{P}best_i - \vec{X}_{i,k}) + C_{soc} \cdot R_{soc}(\vec{G}best_i - \vec{X}_{i,k}) \qquad \text{(eq. 3.2)}$$

Η μαθηματική διατύπωση των παραπάνω δίνεται από τις εξ. 3.1 και 3.2. Το κάθε σωματίδιο i έχει ανά πάσα στιγμή (επανάληψη k) μία ταχύτητα πτήσης $\vec{V}_{i,k}$, που καθορίζει τη θέση του στο χώρο $\vec{X}_{i,k}$. Ενδιαφέρον παρουσιάζει το πώς ορίζεται η ταχύτητα πτήσης: στο $2^{ο}$ μέλος της 3.2 διακρίνουμε τρεις παράγοντες, τους όρους

Κεκτημένης Ταχύτητας ή *Ορμής* (inertia), *Νοητικής ή Γνωστικής Επιρροής* (cognitive influence) και Κοινωνικής *Επιρροής* (social influence) αντίστοιχα. Εξετάζουμε τον καθένα ξεχωριστά:

Ο όρος ορμής αποδίδει την επίδραση της όποιας κεκτημένης ταχύτητας έχει το σωματίδιο στη νέα του ταχύτητα πτήσης, ώστε να αποφευχθεί μια πολύ απότομη διακύμανση ταχυτήτων πτήσης που θα εξέθετε τη διαδικασία αναζήτησης. Το W σε αυτό τον όρο καλείται *Συντελεστής Ορμής,* καθορίζει τι ποσοστό της παρελθούσης τιμής ταχύτητας θα διατηρηθεί ως ορμή και, ως εκ τούτου, είναι η πρώτη εκ των τριών βασικών ρυθμιστικών παραμέτρων της ΒΣΣ.

Ο όρος νοητικής επιρροής αποδίδει την επίδραση της ίδιας εμπειρίας και αντίληψης του ατόμου στην επόμενή του κίνηση. Εκφράζεται ως μία τάση του σωματιδίου να κινηθεί προς όπου έχει καταφέρει μέχρι τώρα να σημειώσει την καλύτερη προσωπική επίδοση (Personal best - $Pbest_i$) επειδή εκεί θεωρεί ότι θα βρει ακόμα καλύτερη λύση στο πρόβλημα. Ο παράγοντας C_{cogn} καλείται Νοητικός Συντελεστής Επιτάχυνσης και είναι ο δεύτερος σε σειρά βασικός ρυθμιστικός παράγοντας. Ο δε R_{cogn} παίρνει τυχαία τιμές μεταξύ 0 και 1 και εκπροσωπεί το στοχαστικό στοιχείο στο νοητικό όρο.

Ο τρίτος όρος, αυτός της κοινωνικής επιρροής, αποδίδει τον τρόπο με τον οποίο η γενικότερη κατάσταση και πρόοδος σύσσωμου του σμήνους βαρύνει στην επόμενη κίνηση του σωματιδίου. Εκφράζεται ως μία τάση του σωματιδίου να κινηθεί προς τη θέση της καλύτερης μέχρι στιγμής λύσης που έχει εντοπίσει συνολικά ο αλγόριθμος (Global best - $Gbest_i$). Ο παράγοντας C_{soc} καλείται Κοινωνικός Συντελεστής Επιτάχυνσης και είναι ο τρίτος βασικός ρυθμιστικός παράγοντας. Για τον R_{soc} ισχύει ό,τι και για τον R_{cogn}.

Το σχετικό μέγεθος των δύο συντελεστών επιτάχυνσης καθορίζει το κατά πόσο το σωματίδιο «συμμορφώνεται» με τις επιταγές της συλλογικής συμπεριφοράς και κατά πόσο ακολουθεί το δικό του «ένστικτο». Όπως θα συζητηθεί και παρακάτω, έχει παρατηρηθεί ότι η έντονα νοητική συμπεριφορά ευνοεί την - κάπως χονδροειδή αλλά γρήγορη - εξερεύνηση του χώρου αναζήτησης που είναι περισσότερο επιθυμητή κατά τα πρώτα στάδια της βελτιστοποιητικής προσπάθειας. Η συμπεριφορά βάσει κοινωνικής επιρροής, αντίθετα, ευνοεί την πιο εκλεπτυσμένη και στοχευμένη αναζήτηση στην περιοχή των βελτίστων λύσεων, αφού ο αλγόριθμος έχει εντοπίσει τις υποπεριοχές αυτές, προς το πέρας της εκτέλεσής του. Αξίζει να αναφερθεί ότι έχουν γίνει πειραματισμοί με όλα τα πιθανά εναλλακτικά σχήματα της ΒΣΣ που περιλαμβάνουν μόνο δύο εκ των όρων ορμής, νοητικής και κοινωνικής επιρροής, που επιβεβαιώνουν την ανωτερότητα του πλήρους σχήματος.

Επισημαίνεται ότι ο λόγος που και ο παράγοντας $Gbest_i$ έχει δείκτη i είναι ότι, σε προβλήματα ΠΚΒ, δεν αντιλαμβάνονται απαραίτητα όλα τα άτομα την ίδια λύση ως καθολικά καλύτερη. Άρα το Gbest είναι γενικά ίδιον του κάθε ατόμου, όπως ακριβώς και το Pbest.

Το σχήμα 3.1 απεικονίζει τη ροή ενός στοιχειώδη τέτοιου αλγορίθμου ΒΣΣ, κατ' αντιστοιχία με αυτήν ενός στοιχειώδη ΕΑ (σχ. 2.1).

3.2. Ρύθμιση των Κυρίων Παραμέτρων της ΒΣΣ

Η τιμή που θα λάβουν οι τρεις αυτές ρυθμιστικές παράμετροι, δηλαδή οι δύο συντελεστές επιτάχυνσης και ο συντελεστής ορμής, και η μεταβολή, ενδεχομένως, αυτής της τιμής κατά τη ροή της διαδικασίας αναζήτησης επιλέγονται με γνώμονα τις εξής δυο επιθυμητές καταστάσεις:

I. Οι ταχύτητες πτήσης να μην πάρουν ιδιαίτερα υψηλές τιμές και, το κυριότερο, να μην υπάρχουν υπερακοντισμοί και απότομες διακυμάνσεις στις τιμές αυτές. Ειδάλλως, τίποτα δεν εμποδίζει ένα άτομο να βγει και εκτός ορίων του χώρου αναζήτησης ή, ακόμα και αν κάποιος κατάλληλος μηχανισμός το κρατάει κοντά ή εντός των ορίων αυτών, να μην έχει την επιθυμητή συμπεριφορά. Παράλληλα, δεν επιθυμούμε και να γίνουν οι ταχύτητες πολύ μικρές.

II. Ο αλγόριθμος να επιδεικνύει ικανοποιητική εξερευνητική ικανότητα στα πρώτα στάδια της αναζήτησης, να ευνοείται δηλαδή η νοητική συμπεριφορά και οι ταχύτητες να είναι αρκούντως υψηλές ώστε σχετικά μεγάλες αποστάσεις εντός του χώρου αναζήτησης να καλύπτονται γρήγορα. Παράλληλα όμως, να μην πάσχει στα ύστερα στάδια, όταν το σμήνος έχει πιθανότατα συγκλίνει κοντά στις βέλτιστες λύσεις και για να τις εντοπίσει, πρέπει να εκλεπτυνθεί στον απαιτούμενο βαθμό η συμπεριφορά του: να οδηγηθούν σωστά και με αρκετά μικρές ταχύτητες τα σωματίδια προς τις επιδιωκόμενες λύσεις. Αυτή η φάση της στοχευμένης και εξονυχιστικά μικροσκοπικής προσέγγισης των βελτίστων με *εκμετάλλευση* της ήδη συγκεντρωμένης πληροφορίας (πιο δόκιμος είναι ο όρος exploitation - εκμετάλλευση) θεωρείται αχίλλειος πτέρνα της θεμελιώδους ΒΣΣ.

Έχουν προταθεί πολλές διορθωτικές επεμβάσεις στο μαθηματικό υπόβαθρο της ΒΣΣ προκειμένου να επιτευχθούν τα παραπάνω, μεταξύ των οποίων η επιβολή δυναμικά μεταβαλλόμενου άνω περιοριστικού όρου στην ταχύτητα πτήσης. Μία τέτοια προσθήκη ήταν και ο ίδιος ο συντελεστής ορμής W, ο οποίος απουσίαζε από την πρώτη χρονικά πρόταση της ΒΣΣ. Έχει επίσης διεξαχθεί πλήθος παραμετρικών μελετών (περισσότερες

λεπτομέρειες στο κυρίως σώμα της εργασίας).

Μία άλλη πρόταση, που υιοθετείται και στον ΠΑ, είναι αυτή των γραμμικώς μεταβαλλόμενων ρυθμιστικών παραμέτρων C_{soc} ,C_{cogn} και W κατ' αναλογία του ποσοστού των συνολικών επαναλήψεων του αλγορίθμου που έχουν ολοκληρωθεί (εξ. 3.4, 3.5, 3.6). Παρατηρείστε ότι τα W και C_{cogn} μειώνονται με την πάροδο των επαναλήψεων, ενώ το C_{soc} αυξάνεται. Έτσι, ο αλγόριθμος τείνει να ευνοεί όλο και περισσότερο την αναζήτηση υπό συλλογική επιρροή, ενώ περιορίζονται σταδιακά και οι ταχύτητες πτήσης, με τη μείωση του συντελεστή ορμής. Τα άνω και κάτω όρια αυτής της μεταβολής επιβάλλονται κατάλληλα ώστε το άθροισμα των συντελεστών επιτάχυνσης και ο συντελεστής ορμής να μην επιτρέπουν υπερβολική ταχύτητα. Το παρόν σχήμα, όπως είναι προφανές, προσανατολίζεται στο να ενισχύσει την *εκμετάλλευση* χωρίς να συμβιβάσει την πολύ καλή εξερευνητική ικανότητα της ΒΣΣ.

3.3. Αντιπαραβολή ΒΣΣ και ΕΑ

Εξετάζοντας το θεωρητικό υπόβαθρο των δύο ιδεών, παρατηρούμε αμέσως τις εξής ομοιότητες: και οι δύο είναι στοχαστικές μέθοδοι και μάλιστα πληθυσμιακού τύπου, χειραγωγούν δηλαδή ένα πεπερασμένο πλήθος υποψηφίων λύσεων που κινείται με τρόπο κατά μεγάλο ποσοστό τυχαίο εντός του πεδίου ορισμού του προβλήματος αναζητώντας λύση(-εις) του. Με κατάλληλη εφαρμογή ενός συνόλου *τελεστών* επί των ατόμων του πληθυσμού αποσπούν ενδείξεις για την ενδεχόμενη θέση των ζητούμενων βελτίστων, ενισχύοντας την αποδοτικότητα αυτής της διαδικασίας. Διατηρούν, βεβαίως, τα πλεονεκτήματα των στοχαστικών μεθόδων, την ευκολία χειρισμού, την ανεξαρτησία και ευελιξία τους.

Στην πράξη, μοιάζουν στο ότι διαχειρίζονται και δευτερεύοντες πληθυσμούς, όπως οι *επίλεκτοι* στους ΕΑ και το αρχείο των Pbest στη ΒΣΣ, οι οποίοι υποστηρίζουν τη λειτουργία των τελεστών χειραγώγησης του πληθυσμού. Ο δε βασικός τους πληθυσμός είναι σταθερού μεγέθους, αλλά ενώ στη ΒΣΣ διατηρείται αυτούσιος μέχρι τέλους και απλά επανατοποθετείται στο χώρο, στους ΕΑ ανανεώνεται συνεχώς, καθώς νέες λύσεις γεννώνται στη θέση αυτών που απορρίφθηκαν ελέω κακής επίδοσης. Συνεπώς η ΒΣΣ χαρακτηρίζεται από μια δέσμευση να βελτιώνει επιμελώς όλο τον πληθυσμό της, ενώ οι ΕΑ φροντίζουν απλά να διατηρούν έναν αριθμό ατόμων ανά γενιά σε υψηλό επίπεδο.

Έπειτα, υπάρχουν σαφείς αντιστοιχίες ανάμεσα στις διαδικασίες χειραγώγησης των υποψηφίων του ενός και του άλλου (πίνακας 3.1), παρότι αυτοί καθαυτοί οι τελεστές δεν είναι τόσο διακριτοί στη ΒΣΣ όσο είναι στους ΕΑ. Η διασταύρωση θυμίζει έντονα τη διαδικασία επανατοποθέτησης του σμήνους στο χώρο, όπου

στοιχεία από διαφορετικά άτομα (ενίοτε και δευτερευόντων πληθυσμών) συνυπάρχουν στη νέα λύση που προκύπτει. Ο ορισμός Pbest/Gbest θυμίζει τον ελιτισμό, υπό την έννοια ότι και οι δύο εξυπηρετούν την ποιότητα της νέας «φουρνιάς» υποψηφίων λύσεων επιβάλλοντας να ληφθεί υπόψη η προηγούμενη θετική δραστηριότητα. Όπως θα δούμε και παρακάτω, σε περιπτώσεις ΠΚΒ, η ανάδειξη μίας λύσης σε Global Best μοιάζει πολύ με την *επιλογή* καθότι, έστω κι αν στη ΒΣΣ δε διακυβεύεται η επιβίωση του σωματιδίου, αν αυτό επιλεγεί, ο ρόλος του ενισχύεται σημαντικά. Είναι λοιπόν εξίσου μεγάλης βαρύτητας η επιλογή αυτή να γίνεται αξιοκρατικά.

Άλλη σημαντική ομοιότητα είναι η ύπαρξη ζωτικής σημασίας ρυθμιστικών παραμέτρων, οι οποίες μάλιστα προτιμάται να μεταβάλλονται κατάλληλα κατά τη ροή του αλγορίθμου, προκειμένου να υποβοηθήσουν τη μετάβαση από έντονη εξερευνητική δραστηριότητα στη φάση της *εκμετάλλευσης*. Επί παραδείγματι, ομοίως με τα όσα είδαμε για τους συντελεστές επιτάχυνσης, συνηθίζεται να επιβάλλεται και σταδιακή μείωση στην πιθανότητα εφαρμογής μετάλλαξης.

Η συζήτηση που λαμβάνει χώρα στην παράγραφο 3.4 καταλήγει σε απόπειρα σύγκρισης των χαρακτηριστικών ΕΑ και ΒΣΣ όσον αφορά στη συμπεριφορά τους κατά την εκτέλεση, το ρυθμό προόδου, τη γενικότερη επίδοση και το τελικό αποτέλεσμα. Τα σχήματα 3.3, 3.4, 3.5 απεικονίζουν τα αποτελέσματα μίας πειραματικής προσπάθειας να καταδειχθούν αυτά τα χαρακτηριστικά. Χωρίς να μπορούμε να εξάγουμε συμπεράσματα για τις γενικότερες δυνατότητες της καθεμίας μεθόδου, αφού υπάρχουν άπειρες παραλλαγές της καθεμιάς, ποικίλης πολυπλοκότητας και αποτελεσματικότητας, μπορούμε να παρατηρήσουμε κάποια πάγια φαινόμενα: η ΒΣΣ δείχνει να συγκλίνει πιο γρήγορα κατά τις πρώτες επαναλήψεις-γενιές, επιδεικνύοντας εξαιρετική εξερευνητική ικανότητα. Κατόπιν, κάνει την εμφάνισή της η θεμελιώδης προβληματικότητά της στη φάση της *εκμετάλλευσης* και η πρόοδος ανακόπτεται απότομα, ο ΕΑ κερδίζει έδαφος και ενδεχομένως προσπερνά, δίνοντας συγκρίσιμο η καλύτερο τελικό αποτέλεσμα.

Από τα παραπάνω, τα οποία διαπιστεύονται και από τα πειράματα της παρούσης εργασίας (κεφ. 5), μπορούμε να συμπεράνουμε χονδρικά ότι ο ΕΑ είναι ένας πολύ πιο ισορροπημένος μηχανισμός αναζήτησης, χωρίς εμφανή αδυναμία στη μία ή την άλλη φάση αυτής. Η ΒΣΣ, από την άλλη, δείχνει να πλεονεκτεί στη φάση της εξερεύνησης αλλά χάνει το όποιο προβάδισμα λόγω κακής συμπεριφοράς στα τελευταία στάδια. Να τονιστεί, βεβαίως, ότι ο αλγόριθμος ΒΣΣ του ανωτέρω συγκριτικού πειράματος δεν ενσωματώνει μεταβλητές παραμέτρους, οι οποίες θα βελτίωναν σαφώς την απόδοσή του, ειδικά προς το τέλος. Διαφαίνεται σαφώς η προοπτική υβριδισμού των δύο μεθόδων, με τον ΕΑ να προσφέρει την ισορροπημένη και σταθερή του απόδοση και η ΒΣΣ την περιστασιακή ταχύτητα σύγκλισής της. Η ίδια η ΒΣΣ μπορεί να βελτιωθεί σημαντικά αν

δανειστεί στοιχεία από τους ΕΑ ώστε να μπορεί να προσαρμόζει καλύτερα τα χαρακτηριστικά της ανάλογα με την πρόοδο της αναζήτησης.

4. Ο προτεινόμενος αλγόριθμος

Το 4^{ο} κεφάλαιο παραθέτει μία-μία τις διακριτές λειτουργίες του προτεινόμενου αλγορίθμου, όπως αυτά ορίζονται με γνώμονα την διάκριση του κώδικα σε υπορουτίνες, χωρίς απαραίτητα να τηρείται η σειρά με την οποία αυτές εκτελούνται. Όταν δε γίνεται ανάλογη επισήμανση, επεξήγηση ή παραπομπή, θεωρείται ότι ο περιγραφόμενος μηχανισμός είναι πρωτότυπος. Στο τέλος του κεφαλαίου υπάρχει το πλήρες διάγραμμα ροής της ακολουθίας των διάφορων υπορουτινών (4.24).

4.1. Επανατοποθέτηση Σμήνους

Ξεκινάμε με τη διαδικασία ανανέωσης της θέσης των σωματιδίων στο χώρο αναζήτησης, μέσω του επανακαθορισμού της ταχύτητας πτήσης τους. Εφαρμόζονται δηλαδή οι θεμελιώδεις εξισώσεις της ΒΣΣ (εξ. 4.1, 4.2), που εξετάστηκαν στο κεφάλαιο 3 (επαναλαμβάνονται για λόγους πληρότητας). Εντός της ίδιας υπορουτίνας γίνεται και η προσαρμογή των συντελεστών επιτάχυνσης και του συντελεστή ορμής, σύμφωνα με το σχήμα γραμμικής μεταβολής που συζητήθηκε επίσης στο κεφάλαιο 3 και δίνεται από τις εξισώσεις 4.4, 4.5 και 4.3 για κάθε μέγεθος, αντίστοιχα.

4.2. Αρχικοποίηση

Προχωρούμε στη διαδικασία αρχικοποίησης του αλγορίθμου: εδώ αποδίδεται στο κάθε σωματίδιο η πρώτη θέση που θα λάβει στο χώρο άμα τη εκκινήσει της αναζήτησης. Μία γεννήτρια τυχαίων αριθμών επιλέγει τυχαία τιμές εντός του πεδίου ορισμού των μεταβλητών σχεδιασμού και κάπως έτσι συμπληρώνεται ένα πλήρες διάνυσμα σχεδιασμού για κάθε άτομο. Η ταχύτητα πτήσης δεν είναι δυνατόν να αρχικοποιηθεί τυχαία, αλλά προτιμάται να μην αποδοθεί μηδενική αρχική ταχύτητα, όπως θα ήταν μια επιλογή: τα σωματίδια ξεκινούν με μικρή ταχύτητα πτήσης προς το κέντρο του χώρου αναζήτησης, με μέτρο ανάλογο της απόστασής τους από αυτόν. Αποφεύγεται έτσι να βρεθεί κάποιο σωματίδιο εκτός χώρου αναζήτησης ήδη από την πρώτη επανάληψη.

4.3. Χειρισμός Επίλεκτων Λύσεων

Ακολουθεί η περιγραφή των λειτουργιών που έχουν να κάνουν με τη γενικότερη διαχείριση των ανά πάσα στιγμή μη κυριαρχούμενων λύσεων. Προφανώς, αυτές απευθύνονται σε προβλήματα ΠΚΒ. Θυμίζουμε ότι τα δύο κύρια ζητήματα που χρήζουν προσοχής είναι η κατάρτιση του μετώπου των μη κυριαρχούμενων λύσεων και η εξασφάλιση μιας σχετικής ετερογένειας μεταξύ των περιεχομένων του, ώστε να καταλήξουμε σε ένα σύνολο λύσεων που θα καλύπτει μια ποικιλία συμβιβαστικών συνδυασμών μεταξύ των κριτηρίων. Γίνεται μια σύντομη παράθεση των δημοφιλέστερων τεχνικών (πέραν της SPEA 2 που έχει ήδη συζητηθεί), μεταξύ των οποίων η μέθοδος NSGA II, πολύ κοντά στην οποία βρίσκεται και η πρακτική που υιοθετήθηκε στον ΠΑ.

Η κατάρτιση του μετώπου των μη κυριαρχούμενων λύσεων (ή των επιλέκτων, σε ορολογία ΕΑ) γίνεται σε δύο στάδια. Πρώτα συγκρίνονται μεταξύ τους οι λύσεις που αντιστοιχούν στις τρέχουσες θέσεις του σμήνους και επιλέγονται οι μη κυριαρχούμενες ανάμεσά τους (σχ. 4.1). Κατόπιν αυτές εισχωρούν στους ήδη υπάρχοντες επίλεκτους (σχ. 4.2), το σύνολο των οποίων διαμορφώνεται τελικά αφού απορριφθούν όσες λύσεις προέκυψαν κυριαρχούμενες (σχ. 4.3, 4.4, 4.5).

Η διαδικασία προώθησης της ανομοιομορφίας στο σύνολο των επιλέκτων λύσεων παίρνει τη μορφή επιλεκτικής απόρριψης αυτών που «περιττεύουν», υπό την έννοια ότι υπάρχουν κι άλλες πολύ όμοιες λύσεις, δηλαδή πολύ κοντά τους στο χώρο των κριτηρίων. Ως ότου ξεπεραστεί ένα προκαθορισμένο, από το χρήστη, άνω όριο πλήθους των επιλέκτων δε γίνεται καμία παρέμβαση στα περιεχόμενά του. Εφόσον αυτό ξεπεραστεί, απορρίπτεται το πλεονάζον πλήθος λύσεων, ως εξής: αποδίδεται σε κάθε μη κυριαρχούμενο διάνυσμα σχεδιασμού μία τιμή, ίση με την -αδιαστατοποιημένη- απόσταση από το πλησιέστερό του άλλο μη κυριαρχούμενο άτομο. Η απόσταση αυτή μετράται στο χώρο των στόχων, όχι των μεταβλητών σχεδιασμού. Ο υποψήφιος με τη μικρότερη τέτοια τιμή απορρίπτεται (σχ. 4.6), και η διεργασία επαναλαμβάνεται ως ότου οι επίλεκτοι να είναι του επιθυμητού πλήθους (σχ. 4.7, 4.8).

4.4. Ενημέρωση Personal Best & Απόδοση Global Best

Όσον αφορά την ανανέωση του διανύσματος Pbest, υπάρχουν διάφορες επιλογές. Όλες, όπως είναι φυσικό, ξεκινούν από σύγκριση της νέας λύσης-θέσης στο χώρο του σωματιδίου με το υπάρχον Pbest. Κυριαρχεί η μία λύση επί της άλλης? Αν ναι, τότε προφανώς επιλέγεται η κυρίαρχος λύση. Αν καμιά δεν κυριαρχεί επί της άλλης, είναι στη διακριτική μας ευχέρεια να επιλέξουμε μεταξύ αντικατάστασης ή

διατήρησης του τρέχοντος Pbest. Ο ΠΑ, καταρχήν, επιλέγει να αντικαθιστά το Pbest.

Μία άλλη διεργασία, χωρίς νόημα σε μονοκριτηριακά προβλήματα, αλλά κομβική στην ΠΚΒ με ΒΣΣ, είναι η απόδοση σε κάθε σωματίδιο ενός Gbest διανύσματος, το οποίο προφανώς θα επιλεγεί από τους επίλεκτους, αλλά πώς; Αναπτύσσονται μέθοδοι που δε θα παρεμποδίζουν, αλλά θα υποστηρίζουν την ετερογένεια του συνόλου των τελικών λύσεων. Εξετάζονται συγκεκριμένα οι εξής εναλλακτικές, με αύξουσα αποτελεσματικότητα, όπως αποδεικνύει η σύγκρισή τους, τα αποτελέσματα της οποίας απεικονίζει το σχ. 4.13:

- **Η μέθοδος της Ρουλέτας**: Επιλέγεται για κάθε σωματίδιο, ως Gbest αυτού, ένας οποιοσδήποτε εκ των εκλεκτών, εντελώς τυχαία. Εδώ η ετερογένεια εξυπηρετείται αλλά η μέθοδος δεν είναι ιδιαίτερα στοχευμένη, με επίπτωση στο ρυθμό προόδου.

- **Η μέθοδος της Εγγύτητας**: Υπολογίζεται η απόσταση, στο χώρο των στόχων, του εξεταζόμενου σωματιδίου από κάθε μέλος των επιλέκτων. Επιλέγεται το πλησιέστερο. Η λογική της εγγύτητας είναι να συνδεθεί το άτομο με μια κοντινή του μη κυριαρχούμενη λύση, που αναμένεται να έχει και παρόμοια χαρακτηριστικά, δηλαδή να δίνει όμοια σχετική βαρύτητα στον ένα ή τον άλλο στόχο. Όπως όμως εξηγούν τα σχ. 4.10, 4.11, η εγγύτητα δεν αποκλείεται να συνδέσει το σωματίδιο με μια επίλεκτη λύση η οποία ούτε καν κυριαρχεί επί αυτού! Κάτι τέτοιο δεν είναι επιθυμητό, καθώς δε συνάδει με το ρόλο του Gbest ως οδηγού προς βελτίωση.

- **Η «συνδυαστική» μέθοδος**: Εδώ εφαρμόζεται πάλι ρουλέτα, αλλά μόνο μεταξύ των κυριάρχων επί του εξεταζόμενου σωματιδίου εκλεκτών. Έτσι μπορούμε τουλάχιστο να εγγυηθούμε ότι το προκύπτον Gbest θα είναι μια συνολικά καλύτερη λύση από το εξεταζόμενο σωματίδιο στην τρέχουσα θέση του, ή εξίσου καλή, αν το σωματίδιο βρίσκεται σε μη κυριαρχούμενη θέση-λύση. Χρησιμοποιείται στην τρέχουσα έκδοση του ΠΑ.

Η παράγραφος 4.4 ολοκληρώνεται με προτάσεις για περαιτέρω βελτίωση της διαδικασίας απόδοσης Global Best.

4.5. Άλλες Λειτουργίες

Ο χειρισμός περιορισμών που τίθενται τυχόν από το πρόβλημα γίνεται από κατάλληλο τελεστή που συγκρίνει τις τιμές των περιοριστικών συναρτήσεων, για

κάθε υποψήφια λύση, με τα όρια που έχουν τεθεί. Όλοι οι περιορισμοί αντιμετωπίζονται ως άνω κλειστές ανισώσεις (εξ. 4.6), όπως αποδεικνύεται από τις εξ. 4.7, 4.8, 4.9 ότι μπορεί να γραφεί κάθε είδους περιορισμός. Ο τελεστής περιορισμών δε λαμβάνει αυστηρά υπόψη του το καθορισμένο άνω όριο αλλά δίνει και ένα επιπλέον περιθώριο, πέραν αυτού. Αν μια τιμή συνάρτησης περιορισμού βρίσκεται μεταξύ των ορίων, τότε η αντίστοιχη λύση δεν απορρίπτεται μεν, αλλά υφίσταται δυσμενή προσαρμογή των τιμών συναρτήσεων στόχων της (εξ. 4.10), σύμφωνα με το εκθετικό σχήμα της εξ. 4.11.

Το σημαντικότερο εντελώς νέο στοιχείο του ΠΑ είναι ο *Τελεστής Ανάδευσης* (shuffle operator). Ονομάζεται έτσι διότι επεμβαίνει βίαια στη ροή της βελτιστοποίησης, επαναρχικοποιεί («ανακατεύει») το σμήνος ενώ υποδεικνύει και νέες, συγκεκριμένες κατευθύνσεις αναζήτησης για τις εναπομένουσες επαναλήψεις του αλγορίθμου. Το ιδεατό σημείο εφαρμογής της *ανάδευσης* είναι κατά τα τελευταία στάδια της βελτιστοποίησης, όταν η πτήση του σμήνους έχει σχετικά ανακοπεί και τα σωματίδια έχουν κατακαθίσει λίγο πολύ στις τελικές του θέσεις. Η καλή πρώιμη συμπεριφορά της ΒΣΣ μας επιτρέπει την πολυτέλεια να ξοδέψουμε μερικές αξιολογήσεις με το να επαναρχικοποιήσουμε το σμήνος, χωρίς βέβαια να διαγράψουμε την καταγεγραμμένη πρόοδο (το αρχείο των εκλεκτών διατηρείται), κίνηση που μπορεί να αποφέρει σπουδαία οφέλη, κυρίως ως προς την ετερογένεια και ισορροπία του τελικού μετώπου, όπως δείχνουν τα σχ. 4.14 και 4.22. Ανάδευση εκτελείται μία φορά σε καθορισμένο από το χρήστη σημείο και, προαιρετικά, μία δεύτερη, εφόσον πληρούνται συγκεκριμένα κριτήρια σχετικά με την κατάσταση του μετώπου των μη κυριαρχουμένων λύσεων. Οι δύο αυτές πρέπει να απέχουν μεταξύ τους αρκετά, για να έχουν αποτέλεσμα.

Ο μηχανισμός καθορισμού ζωνών υψηλής προτεραιότητας λειτουργεί ως εξής: εντοπίζονται εκείνοι οι επίλεκτοι - προκαθορισμένου πλήθους - που είναι περισσότερο απομονωμένοι στο μέτωπο (σχ. 4.15) και χρίζονται «σημεία βαρύτητας». Παράλληλα με την επαναρχικοποίηση, το αρχείο των Pbest επανακαθορίζεται και πλέον, σε κάθε σωματίδιο αποδίδεται ως Pbest υποχρεωτικά ένα εκ των σημείων αυτών (σχ. 4.17). Στη συνέχεια αφήνεται ο αλγόριθμος να κυλήσει κανονικά, με την παρέμβαση αυτή να έχει ως αποτέλεσμα η αναζήτηση να ενταθεί στις προηγουμένως «παραμελημένες» αυτές περιοχές (σχ. 4.17 - 4.20). Το πλήθος των σημείων βαρύτητας είναι καλό να οριστεί σε χαμηλή τιμή (2:5), ώστε να εξασφαλιστεί ο επιθυμητός, υψηλής κατευθυντικότητας χαρακτήρας του μηχανισμού.

4.6. Είσοδος - Προεπιλεγμένες Ρυθμίσεις

Στο σχήμα 4.23, τέλος, απεικονίζεται το αρχείο εισόδου του ΠΑ, όπου διακρίνονται μάλιστα οι προτεινόμενες ρυθμίσεις. Επισημαίνονται οι λειτουργίες που είναι στη διάθεση του χρήστη μέσω του αρχείου, όπως η δήλωση μεταβλητών σχεδιασμού και επιβολή περιορισμών, ο ορισμός των ρυθμιστικών παραμέτρων και του πληθυσμού του σμήνους κλπ.

5. Πειράματα και Πιστοποίηση

5.1. Παρουσίαση δοκιμαστικών προβλημάτων

Τα προβλήματα που επελέγησαν για να δοκιμαστεί πειραματικά ο προτεινόμενος αλγόριθμος (ΠΑ) είναι όλα ελαχιστοποίησης δύο συναρτήσεων-στόχων. Αφενός, δηλαδή, αποτελούν περιπτώσεις πολυ-κριτηριακής βελτιστοποίησης (ΠΚΒ), στην αντιμετώπιση των οποίων κυρίως προσανατολίζεται ο ΠΑ, αφετέρου οι δύο μόνο στόχοι διευκολύνουν την επίδειξη των αποτελεσμάτων, αφού ο χώρος των λύσεων είναι δι-διάστατος. Τα δύο εξ αυτών των τριών προβλημάτων προέρχονται από μια οικογένεια μαθηματικών συναρτήσεων (η γενική μορφή των οποίων δίνεται από την εξ. 5.1) που προορίζονται για τέτοιες δοκιμές λογισμικού βελτιστοποίησης, ενσωματώνοντας η καθεμία διαφορετικές προκλήσεις για τον αλγόριθμο. Η τρίτη εφαρμογή, περισσότερο πρακτικού ενδιαφέροντος, αφορά στη βελτιστοποίηση, υπό περιορισμούς, του περιγράμματος της αεροτομής ενός πτερυγίου στάτορα από συμπιεστή ελεγχόμενης διάχυσης.

Η ακριβής μαθηματική διατύπωση της πρώτης περίπτωσης, με την κωδική επωνυμία ZDT-1, δίνεται από την εξ. 5.2. Είμαστε, επιπλέον, σε θέση να υπολογίσουμε την αναλυτική λύση του προβλήματος, να εξαγάγουμε, δηλαδή, την αναλυτική έκφραση του μετώπου Pareto των μη κυριαρχούμενων λύσεών του (εξ. 5.3). Το μέτωπο αυτό, που παρουσιάζει συνέχεια στο χώρο των λύσεων και κυρτή μορφή, απεικονίζεται στο σχήμα 5.1.

Στη δεύτερη μαθηματική συνάρτηση (ZDT-3), της οποίας η ακριβής διατύπωση δίνεται από την εξ. 5.4, υπεισέρχεται κατάλληλα ένας τριγωνομετρικός όρος που της δίνει ασυνέχεια στο χώρο των λύσεων. Συγκεκριμένα, το αναλυτικά υπολογισμένο (εξ. 5.5) μέτωπο των μη κυριαρχούμενων λύσεων αυτής αποτελείται από 5 κυρτά, μη παρακείμενα τμήματα. Η αυξημένη δυσκολία του συγκεκριμένου προβλήματος σε σχέση

με το προηγούμενο συνίσταται στον εντοπισμό λύσεων από όλα τα επιμέρους τμήματα του μετώπου, που απεικονίζεται στο σχήμα 5.2. Η ZDT-3 είναι, ως εκ τούτου, ένα πολύ καλό μέτρο της επίδοσης ενός αλγορίθμου κατά τη φάση της *εκμετάλλευσης*.

Η τρίτη εφαρμογή, βγαλμένη από το χώρο των Στροβιλομηχανών, συνίσταται στην εύρεση του βέλτιστου περιγράμματος αεροτομής πτερυγίου από την ακτίνα ποδός του στάτορα αξονικού συμπιεστή. Βέλτιστου, με κριτήρια την καλή αεροδυναμική απόδοση του μεμονωμένου πτερυγίου (ελαχιστοποίηση του συντελεστή απωλειών ολικής πίεσης της ροής γύρω του) αλλά και τη συνεισφορά του στην επιθυμητή λειτουργία του συμπιεστή ως συνόλου, δηλαδή τη στροφή της ροής και συνακόλουθη αύξηση της στατικής της πίεσης. Ο μαθηματικός ορισμός των κριτηρίων γίνεται στις εξ. 5.6 και 5.7.

Μεταβλητές σχεδιασμού του προβλήματος είναι οι συντεταγμένες των 14 ελεύθερων (εκ των 18 συνολικά) σημείων ελέγχου των καμπυλών Bezier με τη βοήθεια των οποίων σχηματίζεται το περίγραμμα των αεροτομών. Τα σημεία αυτά μοιράζονται μεταξύ των πλευρών υπερπίεσης και υποπίεσης, που αποδίδονται από ξεχωριστή καμπύλη η καθεμιά (σχ. 5.3).

Για να μπορούμε να εγγυηθούμε την κατασκευασιμότητα και επαρκή μηχανική αντοχή του πτερυγίου που θα προκύψει από τη διαδικασία, επιβάλλουμε περιορισμό στο ελάχιστο πάχος που μπορεί να έχει ένα οποιοδήποτε τέτοιο πτερύγιο σε διάφορες θέσεις κατά μήκος του. Τα ελάχιστα αυτά πάχη, που εκφράζονται ως ποσοστό του μήκους της χορδής του, δίνονται στην εξ. 5.8. Τέτοιους περιορισμούς στο πάχος, θέτει και το ίδιο το λογισμικό αξιολόγησης, που μάλιστα εξετάζει την ικανοποίησή τους αμέσως μόλις διαμορφωθεί η γεωμετρία, πριν η υποψήφια λύση προωθηθεί στον επιλύτη της ροής. Οι περιορισμοί αυτοί είναι λιγότερο αυστηροί από τους παραπάνω και το σκεπτικό τους είναι να απορρίψουν τις εντελώς απαράδεκτες γεωμετρίες πριν αυτές δεσμεύσουν τους σημαντικούς υπολογιστικούς πόρους που απαιτεί η επίλυση της ροής γύρω τους.

Ένας επιπλέον περιορισμός που τίθεται είναι η στροφή της ροής που επιτυγχάνουν τα πτερύγια να είναι τουλάχιστον 20°, για να μην επιτραπεί υπερβολικός συμβιβασμός της ικανότητας του πτερυγίου να λειτουργεί ως συνιστώσα συμπιεστή, στην προσπάθεια για καλή αεροδυναμική απόδοση (εξ. 5.9).

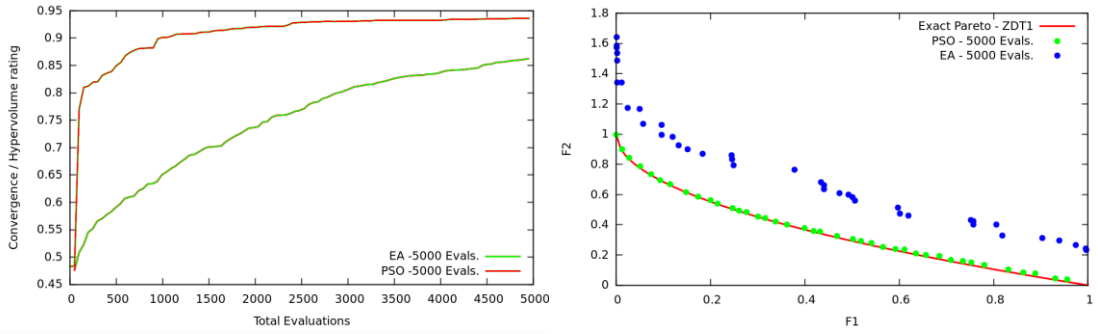
Ο επιλύτης ροής που χρησιμοποιήθηκε, του M. Drela, πραγματοποιεί μια ικανοποιητική πρόβλεψη των χαρακτηριστικών της διδιάστατης ροής γύρω από το πτερύγιο, επιλύοντας το οριακό στρώμα με χρήση ολοκληρωματικής μεθόδου και, αριθμητικά, τις εξισώσεις Euler για το εξωτερικό πεδίο ροής.

5.2. Αποτελέσματα

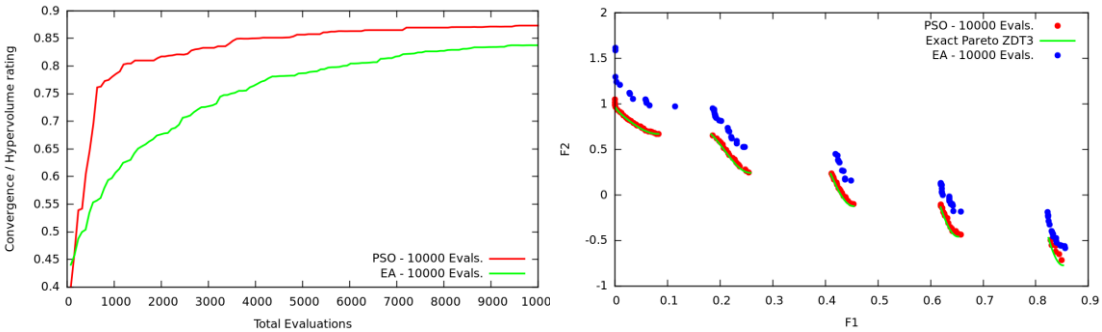
Παράλληλα με τον ΠΑ δοκιμάστηκε και ένας αρκετά πλήρης και δοκιμασμένος ΕΑ της υποκατηγορίας των Εξελικτικών Στρατηγικών, που έχει αναπτυχθεί από το ΕΘΣ, όχι τόσο για λόγους απευθείας σύγκρισης, όσο για να εξαχθούν ποιοτικά συμπεράσματα για τη συμπεριφορά των δύο μεθόδων και, δίπλα στο δεδομένης αποτελεσματικότητας λογισμικό ΕΑ, να διαπιστευτεί και η ανταγωνιστικότητα του παρόντος εκπονήματος.

Τα αποτελέσματα παρουσιάζονται με τη μορφή των μετώπων μη κυριαρχούμενων λύσεων στα οποία κατέληξε η εκτέλεση κάθε αλγορίθμου, για δεδομένο άνω όριο πλήθους αξιολογηθέντων υποψηφίων. Πραγματοποιήθηκαν 5 διαφορετικές εκτελέσεις με διαφορετική γενέτειρα τυχαίων αριθμών για καθεμία, ώστε να εξαλειφθεί η επιρροή της τυχαιότητας στο ενδεικτικό αποτέλεσμα. Επιπλέον διαμορφώθηκε και παρουσιάζεται, για κάθε πείραμα ξεχωριστά και καθεμία εκ των δύο μεθόδων, το αντίστοιχο διάγραμμα *Δείκτη Υπερόγκου*. Ο δείκτης υπερόγκου είναι μια μετρική συνάρτηση που παρέχει μία ποιοτική εικόνα της προόδου της αναζήτησης, καταγράφοντας το ποσοστό ενός προκαθορισμένου τμήματος του χώρου των λύσεων επί του οποίου «κυριαρχεί» ανά πάσα στιγμή το σύνολο των μη κυριαρχούμενων λύσεων που έχει βρει ο βελτιστοποιητής. Το τελικό διάγραμμα που επιδεικνύεται είναι προϊόν εξαγωγής του μέσου όρου των διαγραμμάτων καθεμιάς εκ των 5 εκτελέσεων ανά περίπτωση.

Στα σχήματα 5.5 και 5.6 φαίνεται η επίδοση του κάθε βελτιστοποιητή στο πρόβλημα ZDT-1, εκπεφρασμένη μέσω του διαγράμματος δείκτη υπερόγκου και τα σημεία του μετώπου Pareto, όπου διακρίνεται και η αναλυτική λύση για καλύτερη εποπτεία. Τα δύο λογισμικά εκτελέστηκαν με τις προεπιλεγμένες ρυθμίσεις τους και με ίδια χαρακτηριστικά πληθυσμού, ενώ επετράπησαν 5000 αξιολογήσεις στο καθένα. Ομοίως και για τη δοκιμή έναντι της ZDT-3, όπου επετράπησαν 10000 αξιολογήσεις το πολύ. Παρά την αυστηρότητα, για τις απαιτήσεις του προβλήματος, του ορίου αυτού, και οι δύο αλγόριθμοι τα πήγαν περίφημα στον εντοπισμό λύσεων και από τα 5 διακριτά τμήματα του μετώπου. Ο δε ΠΑ συνέπεσε σχεδόν εξ ολοκλήρου με την αναλυτική λύση (σχ. 5.7, 5.8).

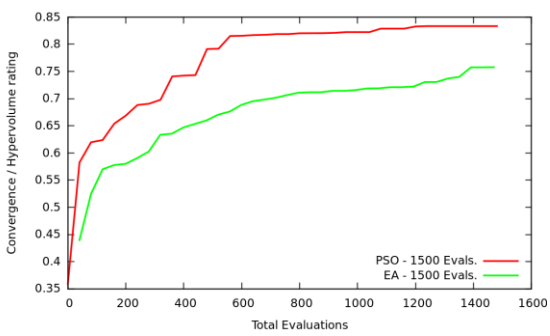


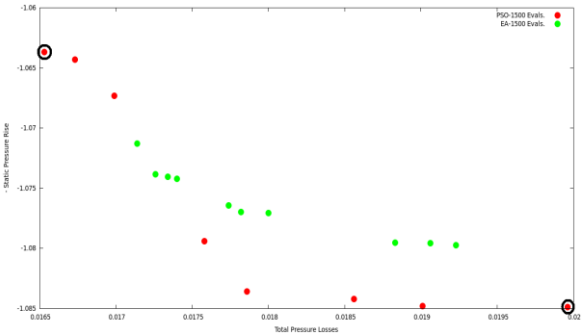
Σχ. 5.5, 5.6. Καμπύλες υπερόγκου συναρτήσει πραγματοποιηθεισών αξιολογήσεων και τελικά μέτωπα μη κυριαρχουμένων λύσεων ΠΑ και ΕΑ ύστερα από 5000 αξιολογήσεις (πρόβλημα ZDT-1).



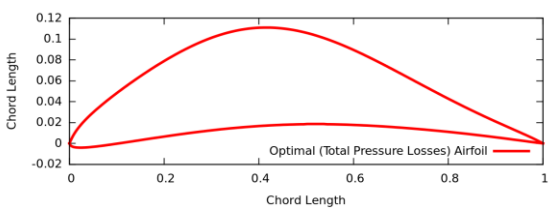
Σχ. 5.7, 5.8. Καμπύλες υπερόγκου συναρτήσει πραγματοποιηθεισών αξιολογήσεων και τελικά μέτωπα μη κυριαρχουμένων λύσεων ΠΑ και ΕΑ ύστερα από 10000 αξιολογήσεις (πρόβλημα ZDT-3).

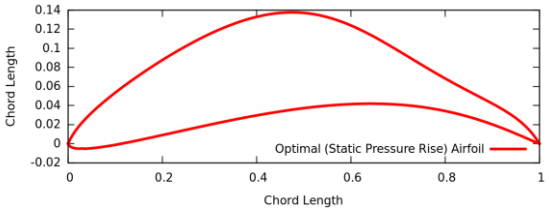
Όσον αφορά, τέλος, την πρακτική εφαρμογή βελτιστοποίησης της αεροτομής του πτερυγίου, επετράπησαν μόνο 1500 αξιολογήσεις, δεδομένου του υπολογίσιμου, αυτή τη φορά, υπολογιστικού κόστους αυτών, εντός των οποίων, πάντως, οι δύο αλγόριθμοι δείχνουν να συνέκλιναν. Τονίζεται η προφανής εξάρτηση των αποτελεσμάτων από τα προεπιλεγέντα όρια μεταβλητών σχεδιασμού. Τα σχ. 5.9 και 5.10 απεικονίζουν το διάγραμμα δείκτη υπερόγκου και τις τελικές λύσεις που απέδωσαν οι προσπάθειες των δύο λογισμικών. Από το τελευταίο, για την περίπτωση του ΠΑ, επιλέγονται δύο ακραίες λύσεις και στα σχ. 5.11 αναπαράγεται, χωρίς να τηρηθεί κλίμακα, η γεωμετρία τους, όπου φαίνεται ο διαφορετικός προσανατολισμός (ελάχιστες απώλειες το ένα, μέγιστη συμπίεση ροής το άλλο) δύο, κατά τα άλλα «βέλτιστων» εναλλακτικών επιλογών.





Σχ. 5.9, 5.10. Καμπύλες υπερόγκου συναρτήσει πραγματοποιηθεισών αξιολογήσεων και τελικά μέτωπα μη κυριαρχουμένων λύσεων ΠΑ και ΕΑ ύστερα από 1500 αξιολογήσεις (πρόβλημα σχεδιασμού αεροτομής). Οι δύο ακραίες κυκλωμένες λύσεις αντιστοιχούν στις παρακάτω αεροτομές (σχ. 5.11).





Σχ. 5.11. Δύο ενδεικτικές τελικές λύσεις (δεν έχει τηρηθεί κλίμακα).

Τα παραπάνω πειράματα κατέδειξαν την ανταγωνιστικότητα του προτεινόμενου λογισμικού, αφού αυτό στάθηκε επάξια δίπλα στον ΕΑ, ξεπερνώντας τον στα σημεία. Το ευχάριστο είναι, ότι ενώ η παραλλαγή αυτή της ΒΣΣ επέδειξε και πάλι την πολύ γρήγορη εξερευνητική συμπεριφορά που χαρακτηρίζει τη μέθοδο, δεν φάνηκε να προβληματίζεται καθόλου στα τελευταία στάδια της αναζήτησης, δείγμα ότι οι σχετικές παρεμβάσεις απέδωσαν καρπούς.

6. Συμπεράσματα - Προτάσεις για μελλοντική εργασία

Τα ικανοποιητικά δείγματα των δυνατοτήτων του ΠΑ που λάβαμε κατά τις δοκιμές τον καθιστούν μια στέρεα βάση για πειραματισμούς και υποδεικνύουν καλές προοπτικές για περαιτέρω ανάπτυξή του. Ήδη από το κεφάλαιο 4 έχουν γίνει κάποιες νύξεις για τις πρώτες διαφαινόμενες βελτιώσεις. Υπάρχουν αρκετές προσθήκες που μπορούν να τον αναβαθμίσουν σημαντικά, τόσο από άποψη επίδοσης όσο και πληρότητας και ευελιξίας, κάποιες εκ των οποίων μπορούν να γίνουν άμεσα και κάποιες, πιο φιλόδοξες και μεγαλύτερης κλίμακας, που ίσως να μπορούν να συμπεριληφθούν στους στόχους μελλοντικής διπλωματικής εργασίας:

→ **Περαιτέρω βελτίωση του μηχανισμού απόδοσης Gbest.** Όπως έχει επανειλημμένα τονιστεί, η διαδικασία αυτή είναι κομβικής σημασίας. Κάθε βελτίωσή της έχει άμεση θετική επίπτωση στην ταχύτητα προόδου του αλγορίθμου. Οι δυσκολίες και προκλήσεις που παρουσιάζει η επιλογή Gbest είναι κοινές με το γενικότερο πρόβλημα της ΠΚΒ, αυτό της επιλογής βάσει πολλαπλών κριτηρίων. Ως εκ τούτου, μπορούμε να στραφούμε προς το γενικότερο χώρο της ΠΚΒ για να αντλήσουμε ιδέες για αποτελεσματική αναβάθμισή της. Στην παράγραφο 4.4 αναφέραμε τα οφέλη που θα έχει πιθανή εμπλοκή μιας μετρικής συνάρτησης που θα λαμβάνει υπόψη τη μορφή του μετώπου Pareto. Σκοπός είναι να επιλέγουμε Gbest για κάθε σωματίδιο, βάσει εξασφάλισης όχι μόνο των καλύτερων προοπτικών βελτίωσης της συγκεκριμένης λύσης αλλά και της καλύτερης δυνατής ετερογένειας του μετώπου των μη κυριαρχούμενων λύσεων.

→ **Ασύγχρονη αναζήτηση - Συμβατότητα με πολύ-επεξεργαστικά περιβάλλοντα.** Καθότι η πλειονότητα των υπολογιστικών προβλημάτων σήμερα αντιμετωπίζεται υπό καθεστώς παράλληλης επεξεργασίας, είναι απαραίτητο να γίνουν όλες οι δυνατές προσαρμογές ώστε ο ΠΑ να μπορεί να λειτουργήσει με τη μέγιστη αποδοτικότητα σε ένα τέτοιο περιβάλλον. Κάτι τέτοιο προϋποθέτει να μη χρειαστεί επ' ουδενί να μείνει κάποιος εκ των διαθέσιμων επεξεργαστών αδρανής. Πρέπει λοιπόν να απεμπλέξουμε την αξιολόγηση των σωματιδίων από την έννοια της «επανάληψης» του αλγορίθμου. Στην παρούσα μορφή του ΠΑ, ένας κύκλος λειτουργίας του ολοκληρώνεται όταν ολοκληρωθεί η αξιολόγηση όλων των θέσεων στις οποίες βρίσκεται σωματίδιο. Ακολούθως, όλο το σμήνος επανατοποθετείται στο χώρο και προχωρά προς επαναξιολόγηση. Για να αποφύγουμε το φαινόμενο να «περιμένει» το σύστημα την καθυστερημένη ολοκλήρωση της αξιολόγησης ενός υποψηφίου για να ολοκληρωθεί ο κύκλος, θα επαναπρογραμματιστεί ο ΠΑ ώστε ένας κύκλος του να συμπίπτει με την εξέταση ενός και μόνο υποψηφίου: με την «επιστροφή» ενός υποψηφίου από αξιολόγηση, το σωματίδιο θα επανατοποθετείται αμέσως, ενώ οι εξισώσεις 3.1 και 3.2 θα λαμβάνουν υπόψη τη μέχρι στιγμής υπαρκτή πληροφορία, ασχέτως του τι κάνουν οι υποψήφιοι που αξιολογούνται εκείνη τη στιγμή.

→ **Προσαρμογή του τελεστή *ανάδευσης*.** Στην παράγραφο 4.5 εξηγήθηκε ο ρόλος του τελεστή ανάδευσης και δείχθηκε ότι μπορεί να οδηγήσει σε σημαντική βελτίωση του τελικού αποτελέσματος. Παρ' όλα αυτά, παραμένει ένας υπερβολικά παρεμβατικός μηχανισμός, ενώ, στην παρούσα μορφή του, απαιτεί να έχει προχωρήσει αρκετά η αναζήτηση προτού αυτός εμπλακεί, ειδάλλως δεν έχει την επιθυμητή συμπεριφορά. Αξίζει να εξετάσουμε μια εναλλακτική προσέγγιση, όπου ο τελεστής *ανάδευσης* θα απασχολεί μόνο ένα ποσοστό του σμήνους. Αυτό

θα περιορίσει το ρίσκο εμπλοκής του, με αποτέλεσμα να μπορούμε να τον χρησιμοποιήσουμε νωρίτερα. Σκοπός είναι να μεταμορφωθεί ο τελεστής σε βασική συνιστώσα του αλγορίθμου, συμμετέχοντας πιο αρμονικά στη διαδικασία βελτιστοποίησης και καθ' όλη την έκταση αυτής.

→ **Διερεύνηση των ρυθμιστικών παραμέτρων.** Ο συντελεστής ορμής και οι δύο συντελεστές επιτάχυνσης χρήζουν προσοχής. Εξηγήσαμε ήδη γιατί διαφορετικές τιμές αυτών ταιριάζουν καλύτερα σε διάφορες φάσεις της διαδικασίας βελτιστοποίησης και υιοθετήθηκε ένα σχήμα γραμμικής προσαρμογής αυτών συναρτήσει των αξιολογήσεων που έχουν ολοκληρωθεί. Το σχήμα αυτό είναι μεν αποτελεσματικό, ειδικά κατά το τελικό στάδιο της αναζήτησης, αλλά δε συγκρίνεται με την εξαιρετική προσαρμοστικότητα που παρουσιάζουν σε ανάλογες περιπτώσεις οι ΕΑ (στρατηγικές μετάλλαξης κλπ.). Πρέπει να αναπτυχθούν πιο εξεζητημένα μέσα ανίχνευσης των αναγκών του προβλήματος σε κάθε φάση και δυναμικής προσαρμογής των παραμέτρων σε αυτές. Πέραν αυτού, πρέπει να διεξαχθεί και μια πλήρης παραμετρική διερεύνηση για αυτά τα τρία μεγέθη, καθώς και για τις διάφορες ρυθμίσεις του τελεστή *ανάδευσης*.

→ **Εφαρμογή κάποιας μεθόδου *μη ακριβούς προ-αξιολόγησης*.** Υπάρχουν μαθηματικά μοντέλα που δίνουν τη δυνατότητα να επιτύχουμε μια καλή προσέγγιση της τιμής της αντικειμενικής συνάρτησης μίας υποψήφιας λύσης εκμεταλλευόμενοι την πληροφορία που είναι διαθέσιμη για τον περιβάλλοντα τη λύση αυτή χώρο αναζήτησης, δηλαδή τις προηγουμένως εξετασθείσες λύσεις που γειτονεύουν με αυτήν [41]. Μία τέτοια μέθοδος μπορεί να υποκαταστήσει περιστασιακά το διαθέσιμο λογισμικό ακριβούς αξιολόγησης, με προφανές όφελος, αφού αυτή η διαδικασία παρεμβολής (ή όμοια) έχει πολύ μικρότερο υπολογιστικό κόστος στην πλειοψηφία των πρακτικών εφαρμογών. Μία τέτοια μέθοδος θα μπορούσε να χρησιμεύσει για να μας δώσει -χωρίς ιδιαίτερη επιβάρυνση- μια πρώτη ένδειξη του πως θα εξελιχθεί ένα σωματίδιο εφόσον του αποδοθεί κάθε ένα εκ των υποψηφίων Gbest, καθιστώντας αυτή τη δύσκολη απόφαση «εκ του ασφαλούς». Ένα ενδεικτικό τέτοιο μαθηματικό εργαλείο είναι το λεγόμενο *Kriging*, εκ του G. Krige που το πρότεινε. Το *Kriging* διαφοροποιείται από άλλα τέτοια εργαλεία λόγω της ιδιότητάς του να παρέχει, εκτός της εκτίμησης της συνάρτησης κόστους, το κανονικά κατανεμημένο πιθανό σφάλμα αυτής. Η εμπλοκή του *Kriging* σε προβλήματα ΠΚΒ παρουσιάζει προκλήσεις όμοιες με αυτές που αντιμετωπίσαμε με την πολύ-κριτηριακή επέκταση της ΒΣΣ, ο δε γράφων έχει μία σχετική εμπειρία στην εφαρμογή του σε συνδυασμό με ΕΑ.

1. Foreword - Abstract

The main objective of this work has been the development and subsequent validation of a complete optimization tool based on the concept of the Particle Swarm. Particle Swarm Optimization (PSO), as which, the entirety of optimization-oriented applications of the Particle Swarm is referred to, is a subcategory of the great family of Swarm Intelligence techniques. As such, it introduces processes inspired from the collective activity of a swarm of insects, flock of birds and school of fish or similar to assess the search for optimal solutions to a variety of problems. The **proposed algorithm** (**PA**) borrows the original core idea of PSO, and applies a series of additions and adjustments, some of which original, some inspired from trends in the ongoing advances in the field of optimization.

Swarm intelligence itself is a subcategory of the Stochastic Methods, which essentially encapsulate all optimization techniques that rely, to some extent, on randomized search within all specified variable ranges to locate the optima. This thesis extends its perspective beyond Swarm Intelligence and approaches Stochastic Optimization holistically, attempting to outline the common features among its various aspects and extract clues as to how each one can be enhanced. Particular attention is given to the most popular and widely applied Stochastic Methods branch, that of Evolutionary Computation and Evolutionary Algorithms (EA's). After EA's and PSO have been introduced and discussed in depth in **chapter 2** and **3**, a long discussion is conducted to highlight the similarities or equivalences between the two, as far as both their philosophical and mathematical background and their practical application is concerned.

The purpose of this is not only to determine the adjacencies between the various components and defining features of these two paradigms, but also to gain insight into possible improvements, either by borrowing principles from each other or by hybridizing. At this point, suffice to say that the prominent product of this analysis is that PSO has a relatively faster rate of progress through the earlier stages of a run, while EA's in general shine at a later stage, the phase of *exploitation*, namely the phase when search space has almost been exhausted and the optimizer focuses on refining the located solutions by searching in their immediate vicinity, thus slightly improving the end result. This rough observation greatly impacts this entire work and its efforts in improving the generic PSO optimizer are focused on moderating this fundamental disadvantage.

In PSO, the members of the swarm, or *particles*, are driven by two main forces: the particle's individual perception of search space, as it is shaped by its own progress thus far (cognitive influence), and its interaction with the rest of the swarm, its awareness of the progress of the swarm as a whole (social influence). The relative effect of these two driving forces is dependent upon a series of tuning parameters. Their choice of value is

therefore crucial, especially so since it is understood that cognitively and socially influenced behaviors relate to performance in different stages of the optimization process. In **section 3.4**, these governing parameters are discussed: Their impact is analyzed, relevant experiments and literature are surveyed and the various existing trends are reviewed. The choice of parameters for the PA is elaborately justified, especially from the perspective of addressing the lacking exploitation capabilities. A scheme that dynamically alters these parameters is adopted, inspired by similar beneficial practices in EA's.

Chapter 4 provides an overview of the entire PA: each section examines a major aspect and its internal processes in depth. A short survey of popular equivalents comes with the introduction of each feature. Unless explicitly stated otherwise, the various processes and features are original. Similarities to existing techniques are present in some cases, while others deviate from common practice. Occasionally, a few alternative approaches to a certain issue will be presented, and their distinctive characteristics will be discussed. The optimizer was generally developed and programmed from scratch. The most notable novelties are the highly directional and strategic social influence structure and the *shuffle* operator, a scheme designed to intervene late in the algorithm's progress by appropriately re-positioning the swarm and determining certain directions in which to intensify search, thus maximizing its efficiency. Other main points, like the *constraint operator*, responsible for administering candidate solutions in breach of any constraints imposed by the problem, and the *initialization* phase are also worth mention.

Emphasis was placed on multi-objective optimization (MOO) problems, namely problems where the optimality of a solution is judged on multiple criteria. As was explained, the multi-objective regime is completely different to the single-objective one and poses additional challenges, some of which are specific to PSO and pertain to the elevated roles of cognitive and social influence. The reader is introduced to the details of MOO and the current trends in dealing with such problems (the Pareto concept, non-dominated solution sorting methods etc.) in **chapter 2**. In **section 4.3** I specifically elaborate on the approaches adopted in the PA to facilitate a successful transition to MOO: A solution selection/sorting procedure determining the best solutions so far, wherein to invest. A solution *spacing* routine is designed and incorporated to guarantee the sought diversity among the various optimal solutions.

The PA is tested against three problems: each of two objectives, with its individual peculiarities. The first two are benchmark mathematical function cases, especially developed by optimization researchers for exactly this purpose: ZDT-1 and ZDT-3. The latter, with its challenging non-contiguous set of optima is a very popular experimental tool. One last test, of a more practical orientation, utilizes the PA for the optimization of a

cascade compressor's stator airfoil, with regard to individual aerodynamic efficiency and good static pressure rise qualities. This case features strict constraints and a higher computational cost per examined candidate solution, thus, a more demanding problem. The PA is subjected to these tests alongside EA-based optimization software of established competitiveness, serving as a point of reference. The demonstrated results showcase the earlier speculated differences in behavior between EA's and PSO, and how the added features have somewhat bridged this gap. They also grant the PA validation as a fully functional and competent optimizer and a decent foundation for further experimentation.

In **chapter 6**, a few suggestions for future work are laid out; various adjustments to the existing features, possible on a short-term basis, as well as more ambitious enhancements that may be achieved as part of a larger project.

2. On Evolutionary Computation

2.1. Stochastic Methods in Optimization

This major branch of Numerical Optimization techniques, under which falls Particle Swarm Optimization, has been growing rapidly in popularity over the last decade or two, with a few becoming "industry standard" approaches for solving challenging optimization problems [45]. This is thanks to their ease of implementation, user-friendliness, non-strictly-mathematical background and high adaptivity to any problem.

Perhaps the best way to define *Stochastic Methods* is in direct contrast to their 'rival' family of optimization paradigms, the *Deterministic Methods*. Deterministic methods are entirely dependent upon knowledge - exact or approximate - of the gradient of the objective function of the problem throughout search space; there is no strict demand for the gradient function to be continuous or perfectly smooth (or for the objective function itself), but its value must be generally calculable in the region of any candidate solution, for the optimization algorithm to benefit from the evaluation of said candidate. The algorithm handles this information appropriately to determine the direction in which lie the minima or maxima of the target function, or, in other words, points in space where the objective gradient verges on zero, to at least one of which, convergence is guaranteed, and at a high rate. It should be noted, without loss of generality, that Deterministic Methods are not able to provide any indication of whether the discovered optimum is a global optimum (which is the desired outcome) or a local one. Unless some additional mechanism is engaged to keep the algorithm from getting trapped in such a local optimum, the optimizer will be terminated when it achieves (relative) a near-zero gradient. Therefore, deterministic methods carry the indisputable advantage of a very fast convergence (usually with respect to the number of solutions which will have to be individually examined, or *evaluated*, for the optimizer to reach an optimum solution), at the peril of ultimately settling for a false optimum. Additionally, in most practical problems the gradient value is difficult (extremely complicated mathematical representation of the examined phenomenon) or even impossible (non-linear, convex, non-contiguous systems) to extract, and therefore, incorporation of such a method is prohibited. One should not overlook the added obstacle of industrial confidentiality, which may not allow the revelation of sufficient information as to the specifics of the problem.

Conversely, stochastic methods [40] have no requirement that any details are known on the nature of the problem (although a general understanding of the case at hand is always beneficial). All that is needed is a list of the associated variables

and the range within which to search. All stochastic methods are principally *search* methods or *heuristics*. In their majority, they are also population-based: they depend on a finite population of *'agents'* initially unleashed into variable space in random fashion. Subsequently, an iterative process is spawned, which determines new eligible agent destinations. This process also encapsulates randomness to one extent or another, but is profoundly deterministic in nature. It essentially manipulates all data gathered by the agents in their venture, inspecting their current whereabouts, as well as their history, from both an individualistic and a holistic perspective. It intelligently combines the information from various sources to form a visualization of the problem space which provides clues as to the possible location of the optima, or at least indications of the more promising subregions, in which to intensify search. The qualities that govern this process along with the overall behavior of agents are the distinguishing element of each heuristic. It has become a trend for these qualities to be inspired by natural or other everyday-life processes, which is the case for both PSO, and the prominent Evolutionary Algorithms presented in the very next segment.

In summary, stochastic methods are advantageous in their universality, as they can almost instantly, with few, if any, alterations deal with any optimization problem, regardless of the technical discipline it falls under, the dimensions of search space, the availability of the various objective function gradients and other particular specifications. A single competitive stochastic optimizer can find numerous and very diverse applications, in finance, research & design and other areas. The price paid for the lack of specificity is the relatively low convergence rate and high total computational expense needed to reach a satisfactory result.

Deterministic methods boast a considerable convergence rate margin over most stochastic paradigms, presuppose, however, that the problem lends itself to derivation of its objectives. Ultimate success is not guaranteed, even when a careful study precedes the optimization process, for careful selection of an appropriate start point. Research is constantly focused on developing intelligent counter-measures against entrapment in local optima. The competitiveness of a procedure centered round a deterministic method is rather associated with the means of calculating the gradient than with the method itself. When a particular optimization process is bound to be repeated on numerous occasions to deal with the same or a very similar object, depending on the size of the project and the resources it occupies, the development of a specialized optimizer, tailored to the peculiarities of this particular case, may be justified. The resulting algorithm, however, will have limited utility beyond this particular application.

2.2. Evolutionary Algorithms

In this segment, we are introduced to the most popular and widely incorporated stochastic method subdivision to date, the *Evolutionary Algorithms* (EA's). An overview of the EA's basics is deemed mandatory, because, even though this work studies PSO, this is done in juxtaposition with the EA. The reader can take advantage of this article to familiarize with principles of Evolutionary Computation in general, and with specific terminology that will be used, often arbitrarily, over the entire length of this work.

Evolutionary Algorithms are not a new concept at all; in fact they date back to the 60's and are originally attributed to John H. Holland [46], who not only proposed the paradigm, in its *Genetic Algorithm* (GA) variation, but also pointed out its great potential as a heuristic optimization scheme. Other prominent sub-classes of EA's are *Evolutionary Strategies* (ES) and *Evolutionary Programming*. It is nowadays common practice for different aspects of the range of Evolutionary methods to hybridize or borrow features from each other, to the point that the separating boundary is hazy and they cannot strictly be classified.

EA's, including the genetic variation, saw considerable advances over the past two decades, rising in popularity. As a population-based technique, an EA acts on populations of strings –more commonly binary- that represent chromosomes from living organisms. The distinguishing manipulation procedure outlined in the previous segment here resembles the Evolution of Species, as it is portrayed by Charles C. Darwin [47]. Evolution is perceived as the process via which a species adapts to its natural environment, to meet its demands and, ultimately, *survive*. All known elements of the evolutionary process are present in an EA, albeit simplified: *Survival of the Fittest*, *Natural Selection*, mating and *reproduction*, competition over available resources and, last but not least, *Mutation*.

In nature, *the fittest is going to survive*. Not only will he better manage to secure the necessary resources, food, shelter etc. but he will also defend himself properly, avoid death/elimination and attract mates for breeding, consequently securing the perpetuation of his genes. The more successful he is at doing the above, the more offspring he will eventually produce, the more his genes will be present in future individuals. Should both mates be of the same high quality, which is highly probable, then chances are they will produce exceptional offspring that might even outperform them and then, go on to breed to produce and further improve the species. Weak and under-average members of a population, on the other hand, will

have difficulty surviving, as they are bound to be eaten or killed in their search of food and shelter, stroke by disease etc. Even if they avoid elimination, their genes are destined for long-term elimination as they will boast limited attractiveness and their chances of mating, much less with a mate of certain prospects, are low. The flourishing of fitter individuals and eventual extinction of underperforming ones gradually improves the overall quality of the population. The specimen has evolved thanks to the survival of those members of the population who demonstrated higher adaptivity and potential.

As with nature, the governing theme in EA's is ***Survival of the Fittest***. The quality of a solution to the problem represented by a gene/chromosome/individual is commonly referred to as *Fitness* of this individual or the corresponding *variable vector.* Each additional iteration of the algorithm coincides with a new generation for the specimen. The genes present in a future generation are for the *genetic operators (Selection, Crossover/Recombination, Mutation)* to determine. The specifics of the mating/breeding process are equally important: Who will mate with whom and exactly how will each parent impact the offspring? How often will one reproduce? In one way or another, the algorithm is made to enforce the position of promising individuals-solutions in the struggle for survival by granting them increased mating chances, selecting them over those of lower standards, who are replaced by the brood of the survivors. The purpose of the algorithm is to constantly improve the overall fitness of the population, in hope that, by enhancing the entire population we are approaching the optimum in whatever our population's quality is measured with; the Objective(s).

As the internal nuts and bolts of evolution are not known to us in their entirety, stochasticity must be present as a substitute for every unknown or unpredictable factor that impacts the form of the genetic progression. The random element is first encountered in the initialization of the population, which is scattered over search space, as dictated by a random number generator, an integral part of the EA. Another area where randomness is key is in mutation; part of the chromosome is tweaked, in an attempt to enrich the population with solutions of added diversity. The process of mutation is traditionally chaotic (although certain variations are very strategic and targeted) but is necessary to stir the population, to avoid jamming into a particular area of search space. Aside from those two cases, stochasticity is, as we will witness, implicated in every single aspect of the EA, in one way or another.

A common practice is for the candidate solution to be represented as an array of bits (0s and 1s). Arrays of other types and structures can be utilized similarly. The main property that makes these genetic representations convenient is that their parts are easily aligned to each other due to their fixed length, which facilitates simple crossover operations and greatly accommodates mutation, as it can be performed simply by *flipping bits* (details in following following paragraphs). Variable length representations may also be used. Tree-like representations are also experimented with, but are more commonly exclusively associated with Evolutionary Programming. These binary strings are handled in accordance to the *schemata theorem* (we shall not speculate further, as it is not the purpose of this work).

Let us now briefly go over the flow of a basic EA:

The algorithm handles three subpopulations, stored in separate archives that are potentially renewed every generation $\boldsymbol{g}$:

i. Parents, let $\boldsymbol{\mu}$ be the size of the respective archive.
ii. Offspring/children, of size $\boldsymbol{\lambda}$.
iii. Elites, of size $\boldsymbol{e}$, where the best solutions thus far are stored.

Directly following the random initialization of λ chromosomes (g=0), the iterative generation process is entered. Candidates are evaluated by the specific evaluation software that must be available externally. *Evaluation* is the calculation of fitness of every candidate solution, in effect the exact calculation of the objective function value adjacent to its variable vector. The elite archive is updated accordingly. The selection operator T_μ is engaged to pick out those μ individuals that will assume parenthood status at the reproduction phase. The offspring that comprise the next generation (g+1) are shaped out of a mix of chromosomes resulting either from the application of the mutation operator T_m upon members of the current population, or *Elitism* (T_e), or regular offspring produced by the crossover/recombination operators T_r. A new generation ensues, starting with evaluation of the new $\boldsymbol{\lambda}$ chromosomes, unless a termination criterion is satisfied. A termination criterion in EA's can either be the exhaustion of an upper limit of allowed evaluations, or the surpassing of a desired target value, or the algorithm reaching a state where it has registered any progress over a given number of consecutive generations or other. In fig. 2.1, a flow chart of a standard EA illustrates the above. Below, the three main operators, along with a fourth – elitism – are shortly reviewed:

- **Selection** is usually implemented in one of the following ways:
  - *Proportional selection*, where each candidate has a chance to be selected, directly proportional to his normalized fitness (his fitness value divided by the sum of all candidates' fitness values) or inversely proportional to it, depending on whether the problem is of maximizational or minimizational orientation, respectively.
  - *Linear ranking*, where the population is sorted by descending fitness, and μ (or fewer) are selected from atop the list.
  - *Tournament selection*, where linear ranking is essentially applied to a number ***k*** of stochastically preselected candidates.

  It should be noted, it is not uncommon for any of the above mechanisms to deliberately pick out a few of the not-so-promising individuals, even those with bottom rankings, to reflect the earlier mentioned natural unpredictability; perhaps these seemingly unimpressive individuals bear impressive potential…

- **Crossover** also comes in numerous variations, the most popular being *1-point crossover*, where the chromosome is divided in two, and the resulting child inherits one part from each parent (in the simplified case of 2 parents producing one child). It should be emphasized crossover is addressed very differently should solutions not be binarily represented.

- **Mutation** is responsible for introducing and preserving added solution diversity, by unpredictably modifying members of the existing population. Among other contributions, it serves as an additional counter-measure against entrapment in local optima. A basic approach is *flipping* one (or more) random bits of a chromosome (turning 0 to 1 and vice versa). Crucially, the probability of a mutation occurrence changes dynamically during an EA run, mostly depending on the state the optimizer is in, its recent rate of progress etc.

- **Elitism**, finally, is the act of *de facto* introducing elite members of the population into the parent archive, without putting them through selection. Elitism can be seen as a reassurance that every next generation will at least not be a step backwards from its preceding, since top solutions have participated in the generation of the new population.

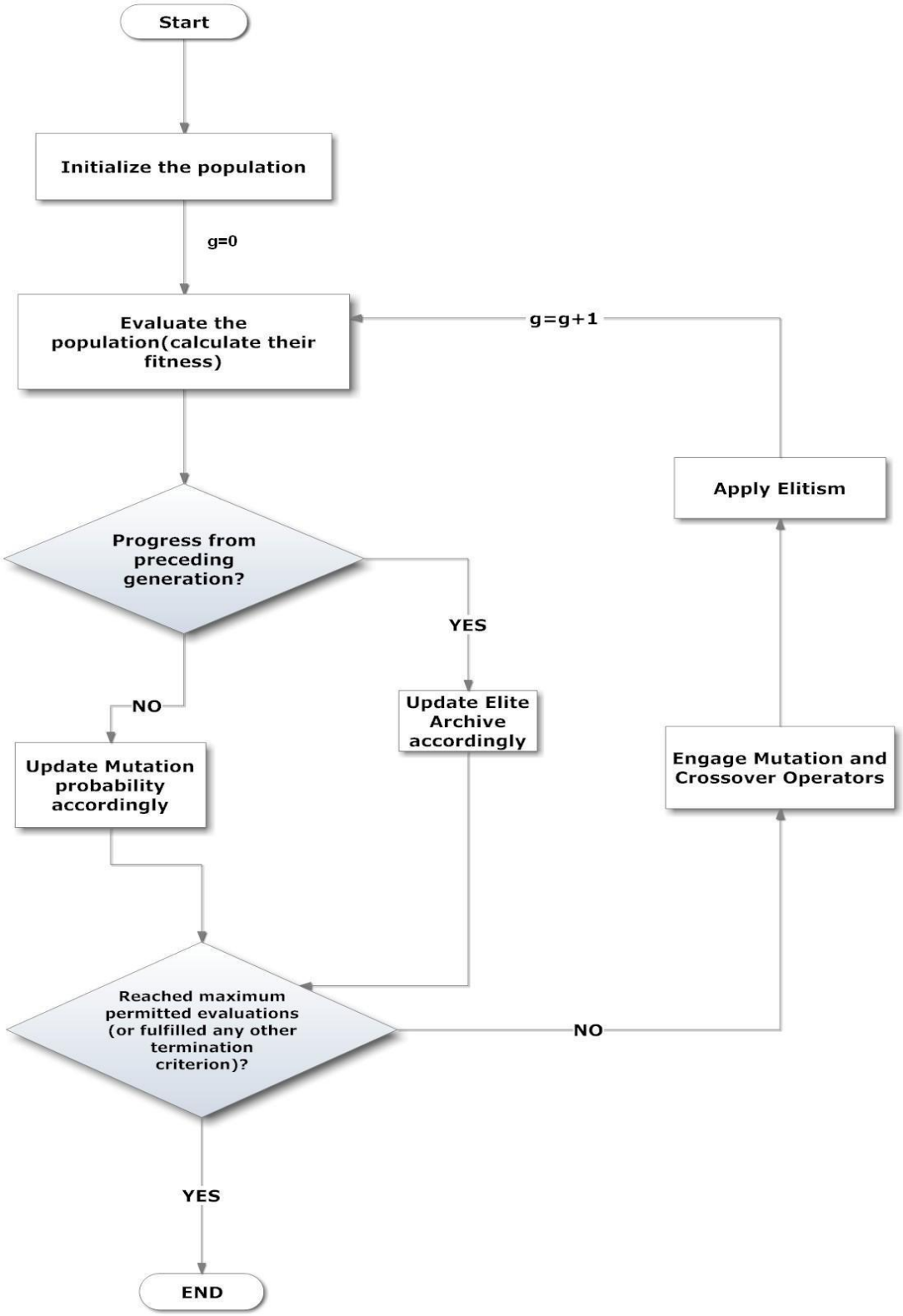


Fig.2.1 Flow chart of a generic EA.

2.3. Multi-Objective Optimization – Pareto optimality

Optimization problems with multiple criteria of optimality (**K>1**) are given special mention in this segment. It is imperative for the reader to comprehend the major differences in the way such problems are dealt with, compared to single-objective cases, as the later proposed optimizer specializes in Multi-objective Optimization (MOO) and its performance will exclusively be demonstrated and tested against such problems.

The main concern regarding problems with K>1 objectives is the classification of solutions. It is not possible to rank two objects based on two or more criteria, except if either is better or worse by all said criteria. This issue extends to Optimization: How are we to decide, for example, which solution will join the elite archive, or participate in reproduction as a parent, in an EA? Additionally, it is understood that an optimization method's purpose and measure of performance should not only be its ability to deliver a single satisfactory solution, but numerous solutions of sufficient diversity, offering us choice among multiple solutions which will either be of balanced quality according to all criteria or favor some over other in a distinct blend. Research efforts are, and have been, centered on accomplishing these two things:

- ✓ Develop solution-sorting techniques that will achieve the greatest possible meritocracy in selecting the candidates that will not only be preserved, but also depended upon to guide the search further. For lack of an absolute means of comparison of individuals in a K-dimensional objective domain, the reliability and efficiency of these techniques can only be evaluated by the quality of the final outcome of the algorithm.

- ✓ Encourage the optimizer to propagate solutions from as wide an area of objective space as possible, thus achieving a set of *fit* variable value combinations that are spread out, as universally as possible, across the perceived multi-dimensional *surface* of overall optimality.

In resolution of the first concern raised, we introduce the concept of *Pareto Dominance*. According to Vilfredo Pareto's principle [23], a solution *dominates* another if it clearly outperforms it with regard to at least one criterion, while being at least equal to it, if not better, according to each and every other criterion.

Formally put, a variable vector $\vec{x}_1 \in \Re_N$ dominates a variable vector $\vec{x}_2 \in \Re_N$, $\vec{x}_1 \prec \vec{x}_2$ (for a minimization-oriented problem, which, unless stated otherwise, will be the case for the entire length of this segment) if and only if:

$$\left[F_k^{(1)} \le F_k^{(2)}, \forall k \in 1,...,K\right] \cap \left[\exists k \in 1,...,K : F_k^{(1)} \prec F_k^{(2)}\right] \quad \text{(eq. 2.1)}$$

where $F_k^{(i)}$ is the fitness of the ith candidate according to the kth objective, $\Re_N$ being N-dimensional feasible search space.

Conversely, the solution is labeled as *non-dominated* should there not be at least one other solution that beats it to every objective set by the problem. In a set of candidates, there will always be a minimum of one which is non-dominated, but it is not a given that there will be any completely dominated (dominated by all) individuals. The set of non-dominated solutions, namely those that can better anyone else (including each other) by at least one criterion, form what is referred to as the *Pareto Front*. In MOO, being included in the Pareto front is the closest an individual can come to earning 'Best' status. As should be evident by now, no comparative consideration is viable between two habitants of the Pareto front, as, if examined in pairs, either will have the upper hand by at least one objective rating:

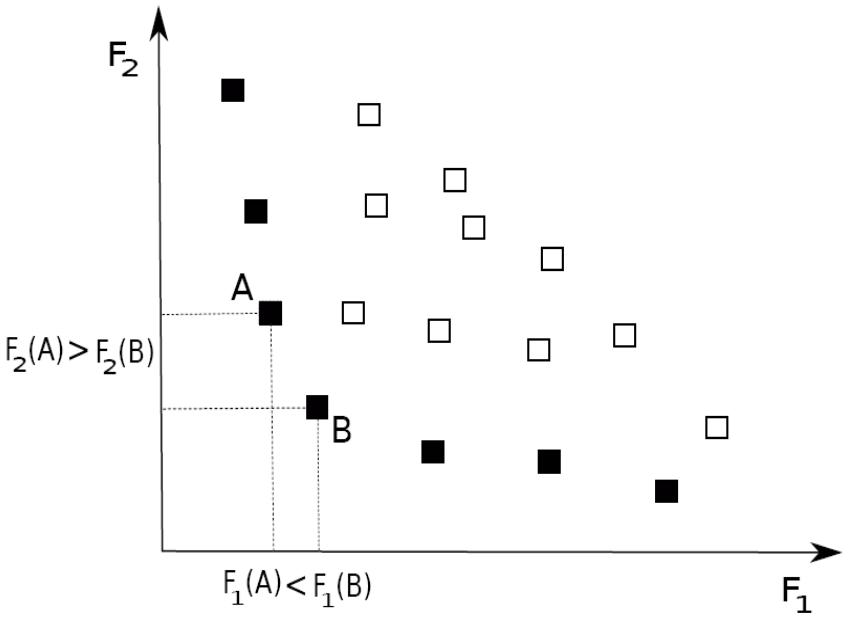


Fig 2.2 An illustration of the objective domain in a 2-objective minimization problem. Black points comprise the current front of non-dominated solutions.

By far the two most established solution sorting schemes in practice are NSGA (Non-dominated sorting algorithm), especially in its advanced version NSGA II, by Deb et al. [2, 3], and SPEA (Strength-Pareto Evolutionary Algorithm). The approach adopted in this work (see chap. 4), though original, is quite similar to NSGA, which will thus be reviewed at a later stage. We will now take a brief look at SPEA, to exemplify the function of such techniques:

SPEA 2 [35] establishes a hierarchy among candidate solutions by sorting them in descending order of Φ_i -as it would otherwise (single-objective problem) sort them according to their fitness value-, which is calculated separately for each solution, a follows:

1. Each candidate is assigned a value S_i, equal to the number of individuals it dominates (fig 2.3).
2. Each candidate is then assigned an additional parameter R_i, the sum of S of all other solutions that it is dominated by (fig 2.4).
3. A third factor is calculated, D_i, which reflects the density of the surrounding the candidate region of objective space. D_i is roughly proportional to the average of distances of all other solutions from the examined one.
4. Lastly, Φ_i is calculated by adding R_i and D_i.

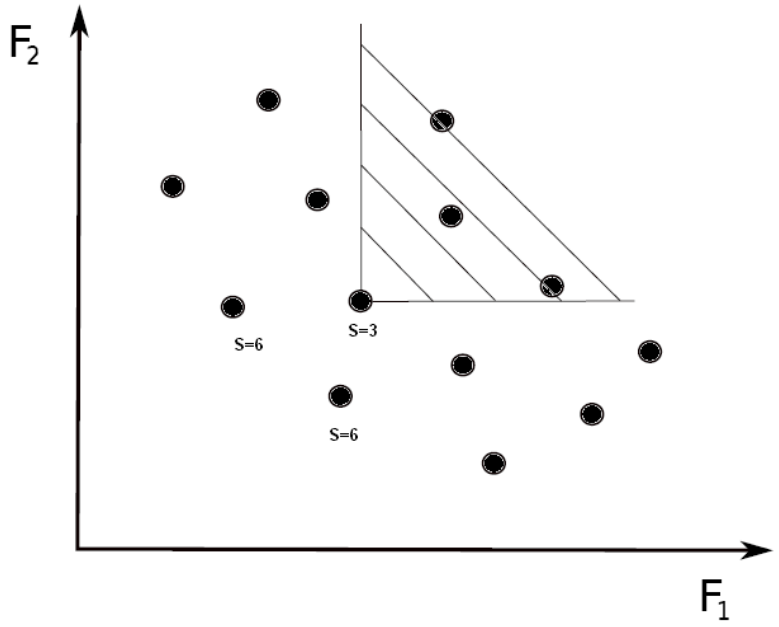


Fig 2.3 The calculation of S_i in SPEA 2, in a 2-objective minimization problem.

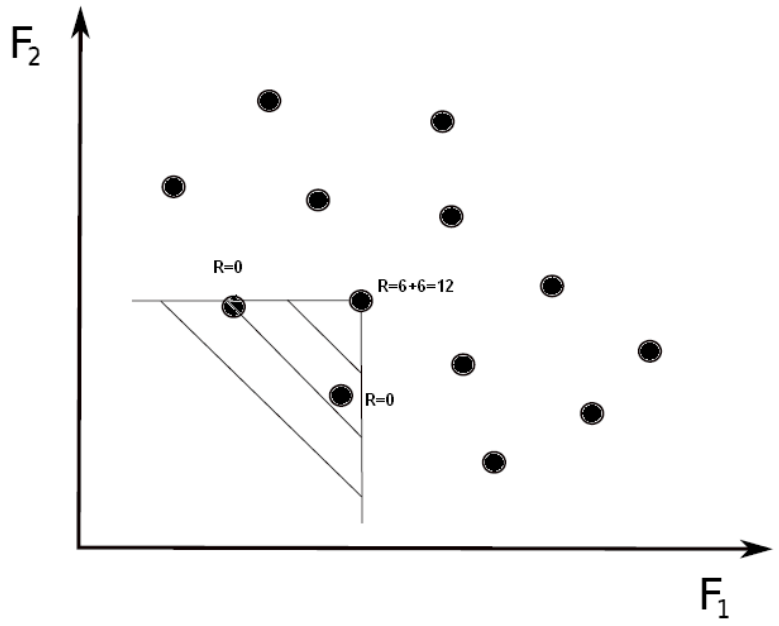


Fig 2.4 The calculation of R_i in SPEA 2, in a 2-objective minimization problem.

Factors S_i and R_i clearly represent the dominance theme in SPEA. Notice how all non-dominated solutions will have an R value of 0. Density factor D_i, on the other hand, is purposed to favor isolated candidates over solutions that fall too close to one another, to help achieve the desired disparity of ultimate solutions that was highlighted earlier in this segment. All similar techniques, NSGA included, incorporate some sort of mechanism that boosts the sorting status potential of such 'rarer blends' of objective function values.

3. On Particle Swarm Optimization

3.1. Swarm Intelligence and the Particle Swarm Paradigm

Particle swarm optimization (from here onward referred to as PSO) is a distinct member of the rather large family of *Swarm Intelligence* methods for optimization. The *particle swarm* paradigm is originally attributed to James Kennedy and Russell Eberhart, who captured the concept while approaching stochasticity in social behavior from a rather philosophical point of view [28]. Hence, the resulting algorithm was initially developed as a social behavior study and simulation tool. A simplified version was observed to be able to serve as an optimization heuristic and it was proposed as such in 1995 in its namesake work: "*Kennedy, J. and Eberhart, R.: Particle Swarm Optimization"* [1].

In 1989, Gerardo Beni and Jing Wang first defined Swarm Intelligence as ***"the collective behavior of decentralized, self-organized systems, natural or artificial"***. Swarm intelligence systems are typically made up of a population of simple agents interacting locally with one another and with their environment. As there is no centralized control structure dictating how individual agents should behave, certain interactions between said agents lead to the emergence of 'intelligent' global behavior, whereof these agents may be - and usually are - completely unaware. Depending upon the nature of the agents themselves and on that of their interactions, which are, in large part, *stochastic* (meaning they feature randomness), the different concepts, whether of philosophical or technical interest, are labeled under 'bird flocking' intelligence, 'ant colony' intelligence, 'fish schooling' intelligence etc.

> *"There is some degree of communication among the ants, just enough to keep them from wandering off completely at random. By this minimal communication they can remind each other that they are not alone but are cooperating with teammates. It takes a large number of ants, all reinforcing each other this way, to sustain any activity - such as trail building - for any length of time. Now my very hazy understanding of the operation of brains leads me to believe that something similar pertains to the firing of neurons…."*
> *(Douglas Hofstadter, 1979)*

From a purely technical standpoint, the term 'Swarm Intelligence' is used to describe algorithms and distributed problem solvers inspired by and modeled after this intelligent collective behavior. PSO in particular is inspired by the choreography of a bird flock or fish school in search of food. Its creators opted to

nickname the agents 'particles' due to their appearance as dots on screen in the early illustrations of the algorithm during execution. From conception to establishment as an efficient optimizer, PSO went through numerous phases via trial and error: After the initial social behavior model showed optimization potential, several new features, all in attempt to simulate a force that presumably drives a member of a bird flock or fish school, were incorporated and tested. Kennedy [28] quotes Craig Reynolds [29] as arguing that a very realistic simulation of a bird flock is achieved by assuming individual birds are driven by these 3 local forces:

- ✓ collision avoidance, namely pulling away before they crash into one another
- ✓ velocity matching, trying to maintain about the same speed as their flock neighbors
- ✓ flock centering, trying to converge towards a perceived 'center' of the flock in motion

It is also understood that, while a flock is flying randomly in search of edibles, every single member depends largely on observing other birds eating or seeing another member of the flock descending toward something it has found. Such socially inherited information weighs almost as heavily as seeing the food itself. Most of the flock may be unaware of the exact location of food, but readily responds to such social signals, indications that a member of the flock may have run into something interesting. The utility of this for optimization purposes is evident: Besides having a population of search agents randomly exploring problem space in search of a better solution to the problem, these search agents can continuously exchange clues as to their current status in approaching the target. This social awareness, coupled with the individual's ability to search with their own senses and maintain a memory of its own recent experience perfectly lends itself to the development of a population-based stochastic optimizer. As soon as this was realized, the social simulation tool was simplified and PSO was officially proposed.

3.2. Particle Swarm Optimization

According to the earlier described analogy, a *swarm* (let *S* be its size) is essentially a set of *particles* released within search space and allowed to move about freely. By reaching any position in space, a particle takes up a unique set of *decision variable* values, therefore representing a potential or *candidate* solution to the problem at hand.

Each individual particle is, at all times, described by its own position vector **X** and velocity vector **V**. It should be noted that, at the absence of *time* in its strictest sense, or, if you will, considering the time step to equal one time unit, both position and velocity are measured as distance. Velocity is numerically taken equal to the distance that will be traveled by the particle from its current position, to reach its next, as is illustrated by the formula of position update [1]:

$$\vec{X}_{i,k+1} = \vec{X}_{i,k} + \vec{V}_{i,k+1} \qquad \text{(eq. 3.1)}$$

where position and velocity values are directly added.

Particles are characterized by *cognitive* and *social* memory, meaning they can recognize the points in space where good solutions were located in the past either by themselves or by any other member of the swarm. The various interactions between particles, as well as the fashion in which this memory is taken advantage thereof, dictate the velocity that a given particle travels with at any given moment. This process of updating velocity is described by the following formula:

Updated Velocity = Carried Momentum + Cognitive Influence + Social Influence

Or, in its numeric form [1]:

$$\vec{V}_{i,k+1} = W \cdot \vec{V}_{i,k} + C_{cogn} \cdot R_{cogn}(\vec{P}best_i - \vec{X}_{i,k}) + C_{soc} \cdot R_{soc}(\vec{G}best_i - \vec{X}_{i,k}) \qquad \text{(eq. 3.2)}$$

where $\vec{X}_{i,k}$ is the position vector of the i^{th} swarm member during the k^{th} iteration of the algorithm and $\vec{V}_{i,k}$ is the velocity vector of the i^{th} swarm member during the k^{th} iteration. At this early stage, an iteration of the algorithm can be perceived as a

cycle between two consecutive speed and position updates of a given particle. On the right hand side of eq. 3.2, the first term represents the momentum that the particle in motion is already carrying, and is known as *inertia* or *momentum*, whereas *W* is known as *inertia weight* or the *momentum coefficient*.

The second term matches *cognitive influence* and expresses the way in which the particle's past experience impacts its course of 'flight', independently of the rest of the swarm's whereabouts. It is also encountered in relevant literature as 'local drive' or 'local accelerator'. A few slightly different approaches have appeared over the years as to how this cognitive influence will be numerically represented. This 'Personal Best' alternative, or 'Pbest' for short, which is by far the most prominent, suggests that the particle is drawn towards the location of its best so far achieved solution ($Pbest_i$), or *fitness rating*, or simply *fitness*, as which, solution relative quality is commonly referred to in Evolutionary Computation.

$$C_{cogn} \cdot R_{cogn}(\vec{P}best_i - \vec{X}_{i,k})$$

The other two factors that complete the cognition term are C_{cogn} and R_{cogn}, *cognitive acceleration coefficient* or *local acceleration coefficient* or *cognitive learning rate* and *random cognition coefficient*, respectively. R_{cogn} takes random values, uniformly distributed between zero and one, $\in (0,1)$. As such, it provides the necessary *stochasticity* in the process of iterative relocation of the particles. C_{cogn} can take up various values, either constant or variant throughout the algorithm's execution. The higher its value, the more the cognition term factors in the particle's behavior over other stimula from the swarm or the environment. A higher C_{cogn} is also observed to promote exploration of search space, as it encourages the particle to deviate from the swarm and rely on its own perception of the environment. Such behavior is traditionally sought at the earlier stages of the optimization session.

The third term from the right hand side of eq. 3.2 matches *social influence* and expresses the way in which the general behavior of the swarm as a whole and its overall progress impacts the course of every individual member. Other, less frequent labels are 'global drive' or 'global accelerator'. As for cognitive influence, there are various practices in computational applications of the PSO concept. Again, the most prominent is utilized in this work, and that is the 'Global Best' scheme, or 'Gbest', which suggests that, similarly to Pbest, the particle is drawn towards the best solution among all that the whole swarm, namely any one particle in it, has achieved so far. *Dynamic Neighborhood PSO [7]* also deserves to be

mentioned. In this technique, the examined particle is influenced by its close neighbors, as opposed to the whole swarm. A 'Local Best' solution, picked among the most *fit* of those immediately surrounding the particle is considered, in place of the Global Best. An interesting concept, with considerable utility especially for Multi-Objective problems, it has not yet been very widely endorsed.

$$C_{soc} \cdot R_{soc}(\vec{G}best_i - \vec{X}_{i,k})$$

The social term is completed by C_{soc} and R_{soc}, *social acceleration coefficient* or *global acceleration coefficient* or *social learning rate* and *random social coefficient*, respectively. The same as for R_{cogn} applies for R_{soc}. C_{soc} can also be set to a constant value or vary and, along with C_{cogn} and W, is crucial to the algorithm's convergence characteristics and efficiency. The choice of these parameters must be carefully made, if possible exclusively for every problem. Certain trends have naturally been established and recommended value sets proposed - a more in-depth review will follow. Suffice to say, the higher the C_{soc} value, the more the social term weighs in on the particle's behavior. Contrary to C_{cogn}, a higher C_{soc} is observed to promote *exploitation*, focusing the swarm's efforts on refining the most promising already found solutions. Obviously, as we approach the later stages of the optimization session, the optimizer's performance benefits tremendously from such behavior.

In [32] Kennedy introduces four models of PSO, defined by omitting or restricting components of the velocity formula. The complete formula above defines the 'Full Model'. Dropping the social component results in the 'Cognition-Only Model', whereas dropping the cognition component defines the 'Social-Only Model'. A fourth model, the 'Selfless', is essentially the Social-Only Model, operating similarly to Dynamic Neighborhood PSO, with a slight twist: It does not consider the individual particle's Pbest vector at all but instead considers a 'Best' solution chosen exclusively among its neighboring fellow particles. Therefore, the particle's very own experience has no impact whatsoever on the swarm's course of flight, hence why the variation was tagged 'Selfless'. In [17], Carlisle and Dozier experimentally compare these models, even against dynamic environment problems (problems where the sought optimum is motive throughout the process) and confirm the superiority of the Full Model.

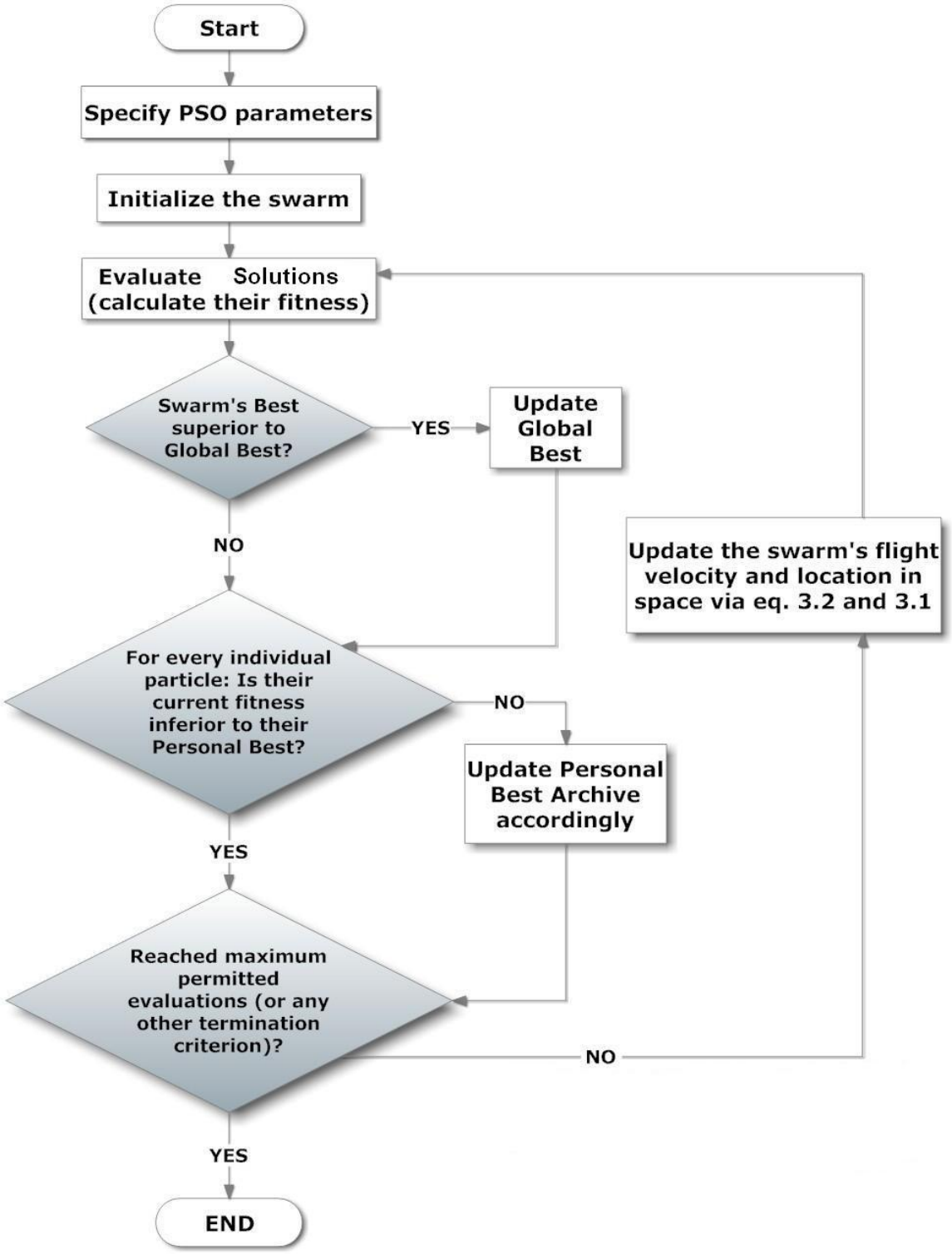


Fig.3.1 Flow chart of a generic PSO optimizer.

3.3. PSO Parametric Tuning

In this segment, we discuss the parameters that govern Particle Swarm Optimizers. These are none other than the 3 major parameters in the original PSO formulae: C_{cogn} (cognitive acceleration coefficient), C_{soc} (social acceleration coefficient) and inertia weight W, from eq. 3.2.

➢ Inertia Weight

Inertia weight W is examined prior to the acceleration coefficients, as it indirectly impacts their choice. It should be noted, at this point, that inertia weight was absent from the initial proposal of PSO [1], but was added [11] soon afterwards, as it was witnessed that PSO suffered from severe instability: Flight velocities occasionally took values too high for the population to be contained within the boundaries of allowed search space. Additionally, the high velocities meant that PSO, despite boasting considerable speed of convergence during the earlier iterations, encountered difficulties in properly benefiting from the spotted solutions by thoroughly searching the space around them, for refined results. This process, often referred to as *exploitation*, is equally vital to the general performance of a search scheme as is swift and universal exploration at the start. In spite of the considerable advances since its invention, exploitation is still considered PSO's Achilles heel, especially in comparison to EA's, which feature excellent exploitation capabilities [9].

As mentioned, W was hastily implemented as a means of keeping velocity below a given threshold and particles from overshooting space boundaries. Up to that point, this role was filled by an imposed maximum value for velocity or relocation step size. A novel touch was to equalize this velocity threshold to the maximum variable range, thus ensuring that variables, in the worst possible case that they would exceed those variable boundaries, would not do so by much. This 'V_{max}' is attributed to the original authors of PSO [12, 13] and found use even after W came into play. A more sophisticated approach was proposed by Clerc [30, 31]: The '*Velocity Constriction Factor'* scheme employed a coefficient K<1, a function of C_{soc}, C_{cogn}, A simplified version, for demonstrative purposes only, is shown below:

$$\vec{V}_{i,k+1} = K \cdot \left[\vec{V}_{i,k} + C_{cogn} \cdot R_{cogn} (\vec{P}best_i - \vec{X}_{i,k}) + C_{soc} \cdot R_{soc} (\vec{G}best_i - \vec{X}_{i,k}) \right] \quad \text{(eq. 3.3)}$$

$$K = \frac{2}{\left| 2 - \varphi - \sqrt{\varphi^2 - 4\varphi} \right|} \qquad \varphi = C_{cogn} + C_{soc} \ , \ \varphi \geq 4$$

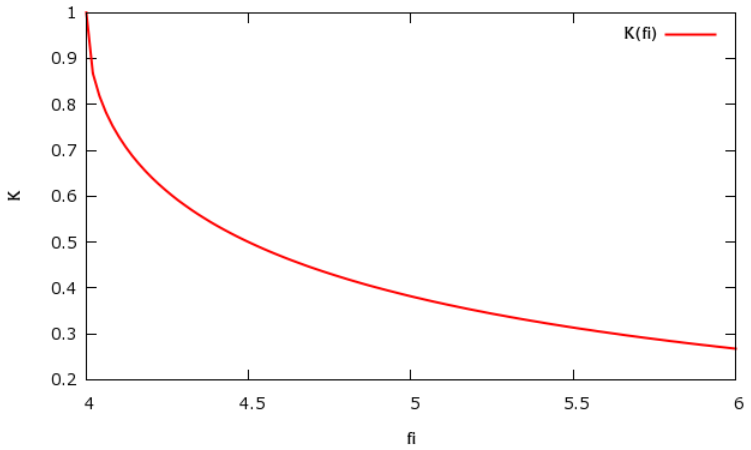


Fig. 3.2 The decrease of the Constriction factor K with the sum Φ of the acceleration coefficients, meant to contain the ultimate velocity value should the acceleration coefficients rise too high.

Notice how it is decreed that it should be φ>4. Many PS optimizers still abide by this guideline [6] that the sum of the two acceleration coefficients should be about 4.0 or more. After its implementation, W was observed to additionally serve as a tuning parameter to balance between global and local exploration. Since a maximum velocity constraint (whether imposed immediately or via the constriction factor) affects global exploration ability indirectly while inertia weight affects it directly, it is obviously preferable to dictate search characteristics through inertia weight only. Focus was placed on experimentally pinpointing a suitable value that would facilitate the desired swiftness in the earlier stages but would not compromise the necessary fine search towards the optimizer's termination. It was found [13] that PSO with an inertia weight in the range [0.9:1.2] had the desired balanced behavior, which was highlighted by the fact that it achieved the best solution for a given number of evaluations. These experiments also rendered the maximum velocity imposition obsolete and this practice was abandoned.

The next major advance was adopting a linearly variant (decreasing) inertia weight to match the much-discussed velocity requirements of each phase (high at the start, steadily lowering as we approach the global optima). As reported in [24], this 'adaptive' inertia weight was inspired by the variant scheme successfully

implemented in the *Simulated Annealing* technique, another popular evolutionary optimizer, for its, reminiscent of inertia, *temperature parameter*. Eberhart and Shi [12] conducted an in-depth parametric analysis and concluded that a linearly decreasing (with iterations/generations) W in the range [0.9:0.4] is a reliable, versatile choice. Other, more extreme applications see W nullified before or upon fulfillment of the maximum iterations criterion. Others maintain a variant approach for part for the session's duration while keeping inertia constant over the remaining iterations [17].

$$\text{variant inertia weight , } W = (W_0 - W_1)\frac{(N_{\max iter} - N_{iter})}{N_{\max iter}} + W_1 \qquad \text{(eq. 3.4)}$$

➢ Acceleration Coefficients

The *cognitive acceleration coefficient* $\mathbf{C_{cogn}}$ and *social acceleration coefficient* $\mathbf{C_{soc}}$ serve as tuning parameters to determine the balance between one particle's drive towards its Pbest and towards the Global Best, or, if you will, between *learning* cognitively and socially. In earlier bibliography, including the initial PSO proposal by Eberhart [1, 5, 8], it is recommended that both coefficients are set at **2.0**. In practice, depending on the case at hand, this may vary and one may take on a greater value than the other, in which case, to avoid uncontrollably high velocity values, it is also recommended [26 , 30] that $\mathbf{C_{cogn} + C_{soc} \approx 4.0}$. This is based on the results of the Velocity constriction factor validation experiments (see velocity constriction formulae eq. 3.3). In [12, 13], it is argued that even lower values, around **1.4**, must be used to ensure competitive efficiency without jeopardizing convergence.

Soon, in the same spirit as with inertia weight, concerns were raised on the poor late performance of PSO and its shortcoming in conducting finer grain search as we approach the end of a run. Inspired by the linearly decreasing mutation and crossover probability factors in popular EA's [9,17], a similar variant scheme as for W, was proposed by Eberhart and Shi [13] and first applied by Ratnaweera and Halgamuge [27] and Tripathi et al. [21]. The latter extensively investigate the effect of a range of W and C_{soc}/C_{cogn} values on the performance of the algorithm and concur that a decreasing inertia weight and C_{cogn} along with an increasing C_{soc}, all within an appropriate range, [0.9:0.4], [2.5:0.5] and [0.5:2.5] respectively, yield better results.

$$\text{Cognitive accel. coef., } C_{cogn} = (C_{cogn,0} - C_{cogn,1})\frac{(N_{\max iter} - N_{iter})}{N_{\max iter}} + C_{cogn,1} \quad \text{(eq. 3.5)}$$

$$\text{Social accel. coef., } C_{soc} = (C_{soc,0} - C_{soc,1})\frac{(N_{\max iter} - N_{iter})}{N_{\max iter}} + C_{soc,1} \quad \text{(eq. 3.6)}$$

The increasing C_{soc}, decreasing C_{cogn} theme is in similar sense to the exploration/exploitation scheme described earlier. It is understood that cognitive influence is linked to the earlier stages of search, where we aim for coverage of the greatest possible part of feasible space and social influence to the latter stages, where we want to make the most out of the experience already gained in order to capture the illusive optima. May I add, near the end of the run, where a great percentage of particles have come to a halt and progress, if any, is very slow, the cognitive term has limited utility, as most particles are effectively their own Pbests i.e. current fitness is also the best achieved thus far. This fact nullifies cognitive acceleration and magnifies the importance of the social term in keeping the optimizer going.

It should be emphasized, at this point, that throughout the relevant literature, most proposals, if not all, adopt equal value for both acceleration coefficients, and in doing so, favor none of the two acceleration terms over the other. The trend is to keep PSO perfectly balanced between global and local search. However, there is no evidence that both terms contribute equally to the overall performance of the algorithm, and I would like to raise the point that C_{soc}=C_{cogn} may not facilitate PSO achieving the ultimate in search efficiency. On the contrary, in light of the experimental work of Dozier et al. in [17], where he validates the superiority of the Social-only model over all other restricted variations of PSO, one is led to consider that the Social term may indeed have a higher contribution to the quality of the final solution, and should therefore be favored. Additionally, as almost any significant innovation of this work is in the direction of advancing the sophistication of the Gbest selection process and the overall functionality of the Social learning procedure, I have opted to endorse C_{soc} and would recommend a C_{soc}>C_{cogn} scheme. Should a variant scheme be adopted, as is also the case with this proposal, the above recommendation does not apply for the entire ranges, i.e. it only has to be: {average C_{soc}}>{average C_{cogn}}.

➢ Population Size

Contrary to EA's, where extensive analyses and surveys have been published with regard to parent-offspring ratios and other population-related data, no particular guidelines are set in relevant literature as to the impact of the size S of the swarm on the performance of a particle swarm optimizer. This is left to be decided according to our best knowledge of the examined problem, which indicates an appropriate compromise between a large population and a capable number of iterations (for a given maximum allowed evaluations).

3.4. PSO versus Evolutionary Algorithms

Despite the slightly competitive implications in the title, this segment is not intended to draw a comparison between the two paradigms, at least not one that will deliver a conclusion of the 'better/worse' kind. It is understood that both heuristics are highly representative members of the family of Population-based Stochastic Methods in Optimization, and as such, feature commonalities that may not be discernible at first, but are present. Studying these similarities helps better comprehend the philosophy of population-based stochastic optimizers and may inspire the observation of other natural phenomena, besides the evolution of species (EA's) and social interaction of sentient beings (PSO), that may also lend themselves to the conception and development of new approaches to evolutionary thinking and optimization. It is indeed fascinating how identifying similar intermediate processes (referred to as operators in strict evolutionary computation terminology) in both, emphasizes the unified fashion in which every mechanism in nature assesses the search for the better, the stronger, the closer, the faster, the *optimal*.

There are also distinctions between the two - distinctions in philosophy that extend into practical application, as well. By studying the structure and the mathematical background of either, we gain insight into what is later validated by experiments: competitive results from both, but very different means of getting there, fundamental differences in convergence behavior and interesting dissimilarities in sheer performance at various stages of an optimization session. Reviewing these distinctions can send us in the right direction when it comes to future development. Each paradigm's shortcomings and strengths are highlighted, pointing out the necessary additions or the features that one can borrow from the other to improve its overall utility, to complement its advantages and counter its disadvantages. We

are particularly interested, fittingly, in all the ways in which PSO can benefit from the major developments in the area of Evolutionary Computation.

A comparative review of philosophy and practice

We begin by closely examining some philosophical aspects of both methods, highlighting those that distinguish them:

First and foremost, both methods are stochastic. That means they are governed by randomness, and in great part rely on it to extract clues as to the location of optima, or even to discover the optima themselves. Crucially, this information may not be obtainable via any other, more orthodox, deterministic means. Their role is to improve upon basic random search, as exhaustive exploration of all ranges of problem space is not possible or computationally viable. They incorporate intelligent criteria, which indicate the areas in which this search must be focused, for the identification of optimal solutions to be accelerated.

What is more, they do not require the knowledge of the gradient of the objective function, or any knowledge on the specifics of the problem at hand, whatsoever. This is of vital importance when such information is impossible or very hard and costly to extract. This also means they feature remarkable adaptability to any optimization problem, and that they address every such problem in the same way, regardless of its specifications. This adaptability is reminiscent of the way in which all things progress self-reliantly in the natural environment, as far as both their long-term characteristics and short-term activity are concerned.

Both are population-based, depending heavily upon a finite number of search agents let loose into search space. Even more fundamental is the form of ***interaction*** of these agents, which is essentially the identity of each method and the core of its functionality. Without any loss of generality, all such methods are based on intelligently comparing and combining the achievements of various agents, to collectively achieve a sum greater than its parts. This is done by applying various operators that manipulate the population and its gained experience. On close inspection of the two paradigms, we notice that:

- Both schemes traditionally adopt secondary populations. EA's carry the elite archive, and access its content whenever they see fit and PSO features particle experience or memory, with its Pbest (personal best solution thus far).

- Both, unless otherwise specified, have a population of standard size. Particles, however, are retained throughout the 'hunt' process and all effort is centered on delivering them all to a rewarding spot in search space. Genetic candidates may not survive for more than a few generations each, as the algorithm reallocates its available search resources from those individuals performing poorly to new individuals generated from those performing relatively well…

- …hence the essence of intense competition in EA's: The 'survival of the fittest' concept means that one individual's life means another's death. PSO has a more coordinative character, an essence of teamwork and fellowship, as particles are bound to support and guide each other, for the process to culminate. On the other hand, in PSO, parent information is partly (Pbest) contained within each particle, while it is freely shared in evolutionary optimization.

One point worth of mention is the 'generation' element in EA's. PSO also works iteratively and most existing configurations copy the generation trend, but is not as constricted by this obligation to simultaneously update its population. 'Asynchronous' variations have been developed for EA's, to rid them of this setback, but PSO is naturally asynchronous. The absence of selection, with elimination in mind, allows for particles to be updated independently of the progress of the rest of the swarm, relying only on what data is available upon completion of their evaluation and fitness assignment. Therefore, it lends itself to parallelization (see section 6.2).

Optimization methods all perform the following procedure:

$$S^{k+1} = T_m(T_n(T_o(T...(S^k)))) \ , \ S^k\text{: the population during the k}^{th}\text{ iteration}$$

where T_m, T_n, T_o etc. are manipulation operators that act on the population to extract its successors.

As we have seen, three are the main classes of operators in evolutionary computation, which are implemented in one way or another in every application of an EA: ***Selection***, ***Crossover/Recombination*** and ***Mutation***. The ***Elitism*** feature is also popular enough to be considered a must-addition. Interestingly, there is an adjacent operation in PSO, for very one of those in EA's. The most easily

recognizable equivalence is that of mutation and crossover with the swarm update formulae of PSO: There, information and experience stemming from more than one individual is taken into consideration, resulting in the formation of a new candidate decision variable vector. These two operations, however, that of randomized, chaotic mutation and that of well-structured, deterministic crossover, are not easily separable in PSO. The author of the paradigm, R. Eberhart often makes the remark that PSO seems to be performing *'mutation with a conscience'* [12,14,24], in the sense that there is some added randomness in the way that particles are relocated in space, very reminiscent of 'bit flipping', but that this relocation is still governed by determinism to a sufficient extent. The crossover element is present in the way that two particles, a Gbest and a Pbest are combined to dictate the particle's new course.

In PSO, there is no selection operator per se, as the same S particles carry on throughout a run, without elimination or generation of additional swarm members. Selection does occur, however, when, in multi-objective problems, we choose one non-dominated solution to act as a Gbest, thus serving as a global guide for other swarm members. This solution earns Gbest status, not at the expense of a fellow particle's survival, but its role in the swarm is elevated.

As for elitism, it is encountered in the Pbest scheme: A secondary population, kept in an external archive, also of size S, is comprised of the best solutions discovered thus far for every swarm member and represents the particles' memory of their past achievements. Pbest vectors do not generally coincide with their adjacent particle's current location space, and are therefore reminiscent of the elite archive, which stores the best overall solutions attained and is utilized when and if the crossover operator requires it, for mating. The equivalences between the operators of each method are summarized in table 3.1:

EA operators	PSO equivalents
Mutation	**PSO velocity adjustment formula**
Crossover	**PSO velocity adjustment formula**
Selection	**Gbest selection scheme**
Elitism	**Pbest scheme**

Table 3.1

Both search techniques are governed by parameters, the cautious selection of which is crucial in taking the maximum out of the search technique's potential. We have already taken a thorough look at the major tuning parameters of PSO, and emphasized their impact, especially on the dynamic characteristics of convergence and how the desired gradual reallocation of the swarm's resources from exploration to exploitation must be facilitated. Although a direct analogy between said parameters and those in EA's is not possible, we can observe certain common trends in the way they are handled: Mutation probability also decreases with generations, to signal the passage from intense exploration to the prioritization of solution refinement. We witnessed the benefits of such a linear variance scheme for inertia weight and the acceleration coefficients, already.

Performance-related distinctions

This segment highlights differences in performance and convergence characteristics. This survey helps comprehend the ways in which either heuristic is strong or weak, the areas in which either can be enhanced and the benefits a possible hybridization can yield. For this, we refer to past empirical studies and experimental results that overlook the latest advances and illustrate, as much as possible, the *bare* optimization methods, to better point out these distinctions, which still exist but may have been bridged somewhat by the ongoing development of these methods.

Peter J. Angeline [9] was the first to address EA's (GA's in particular) and PSO in a comparative context. This work made evident the exploitation weaknesses of the then-newborn PSO and stimulated the addition of inertia weight. He tested both methods using popular benchmark mathematical functions, namely the *Sphere*, *Rastrigin* and *Rosenbrock* functions (see [9]), all of the form: $f = \sum_{i=1}^{n} g(x_i)$, hence why he repeated every experiment for 3 different *n* values. The outcome was, among other, figures 3.2, 3.3, 3.4:

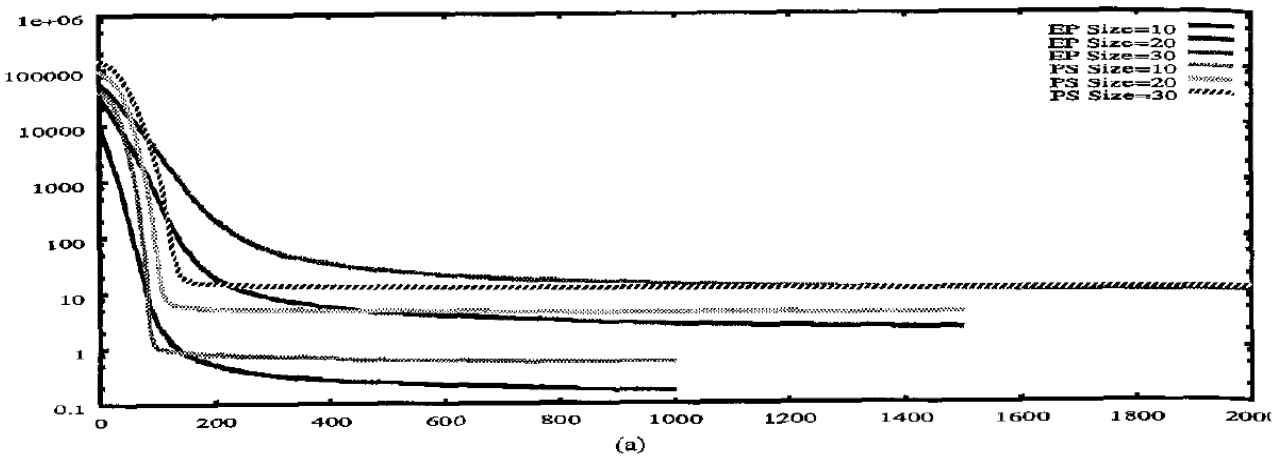


Fig. 3.3 Mean fitness per generation over 50 trials (sphere model function).

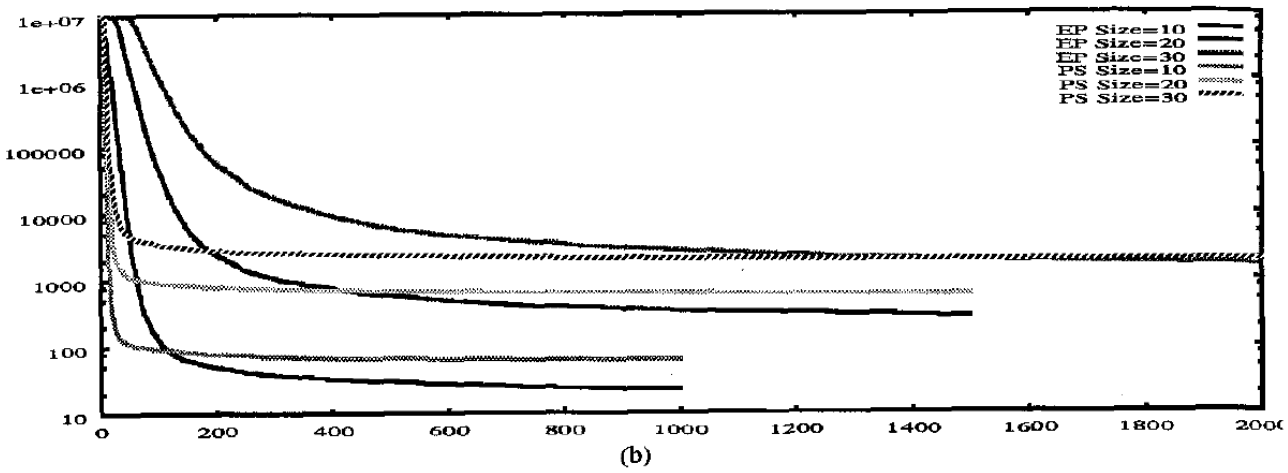


Fig. 3.4 Mean fitness per generation over 50 trials (Rosenbrock function).

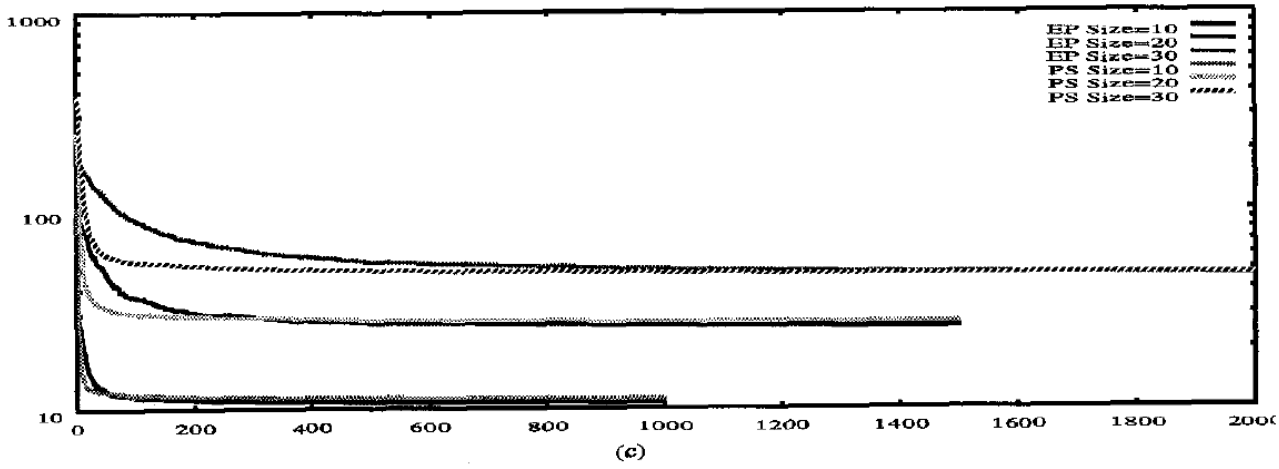


Fig. 3.5 Mean fitness per generation over 50 trials (Rastrigin function).

Over all three functions, PSO shows a significantly faster rate of convergence at the very early stages of search, then quickly halts its progress. The GA decreases its fitness at a slower, steadier pace and ultimately manages to slightly surpass PSO's performance. This occurs at all three vector widths (*n*) tested. These figures are highly indicative of the fact that PSO is principally a faster search method and the GA a more thorough, adaptive and fine heuristic. Under particular circumstances, they both excel, at exploration and exploitation respectively. The Particle Swarm's disadvantage is its inability to dynamically adjust its velocity and particle step size to the demands of the problem once in the general region of the optima. This causes its convergence curve to flatten out dramatically. The Genetic optimizer finds optima at a slower rate but has no trouble tuning its operators so that it can better reflect the granularity of the local search region.

To attempt to justify this ineptitude of *basic* PSO (emphasis on *basic*, as later additions and developments have, to a large extent, eradicated these shortcomings) at coping with demanding localized search is not an easy task. Quoting P. Angeline: "*Particle Swarm's commitment to a highly directional manipulation formula can occasionally hinder its performance on some problems under certain conditions. Conversely, when the gradient information supplied by the personal and global best correctly indicates the direction to the optima, performance of particle swarm is exceptional…Consequently, particle swarm optimization will perform best in environments where the average local gradients point towards the global optima but may have difficulties when the average local gradient point in the wrong direction or is constantly changing. Evolutionary optimizations, however, will be able to perform well in environments where the local average gradient does not point towards the global optima.* ".

Conclusion

The above have hopefully shed some light on the distinctive peculiarities of the two heuristics; this pair represents the cream of the crop of Stochastic methods and Evolutionary Computation in general, and since they demonstrate such converse qualities, future research will – already has – focus on hybridizing them [33,34]. Possible hybrids could include a self-adaptive technique similar to those used in evolutionary computation (mutation *strategies*), capable of dynamically adjusting velocity step size during the course of a session.

However, even for PSO on its own, the potential for improvement is astounding, especially so since it is relatively recent, contrary to EA's, who date back almost

three decades, have received considerable attention from the academic and industrial society, and occupied ample research resources, becoming the prominent search optimization method today. The preceding survey evidences the need to enhance PSO's exploration capabilities, to match the competence of EA's in that area, while retaining the marginal convergence speed advantage. PSO has manifested issues with premature convergence [17, 24] to local optima and a general tendency to collapse towards the first non-dominated solutions found, instead of exhausting its chances to cover the full Pareto front. Implementation of mutation-like techniques can help moderate such phenomena (see chap. 4).

4. The Proposed Algorithm

4.1. Swarm update

(In this segment, unless otherwise stated, we speak exclusively in terms of single-objective optimization, hereon referred to as SOO. Additionally, throughout the entirety of this work, we consider all optimization problems as minimization problems.)

This is where the next generation is derived from the previous one, where all swarm members are given new individual flight velocity values and are, consequently, relocated in search space, assigned new position vectors. This procedure is described by the two core formulae of PSO, repeated here as they were featured in chapter 3:

$$\vec{\mathrm{X}}_{i,k+1} = \vec{\mathrm{X}}_{i,k} + \vec{\mathrm{V}}_{i,k+1} \qquad \text{(eq. 4.1)}$$

$$\vec{\mathrm{V}}_{i,k+1} = \mathrm{W} \cdot \vec{\mathrm{V}}_{i,k} + C_{cogn} \cdot R_{cogn}(\vec{P}best_i - \vec{X}_{i,k}) + C_{soc} \cdot R_{soc}(\vec{G}best_i - \vec{X}_{i,k}) \qquad \text{(eq. 4.2)}$$

In accordance to Evolutionary Optimization terminology, particle position vectors $\mathbf{X_{i,k}}$ are from here may also be referred to as '***Chromosomes***' and an iteration referred to as a '**generation**'.

In the algorithm proposed in this work, inertia weight decreases linearly with iterations (eq. 4.3). The spirit of this is to allow for faster, more dynamic search (maximized *exploration*) during the earlier stages of the optimization process while gradually limiting the speed of the particles to encourage more focused search (maximized *exploitation*), as the algorithm approaches its ultimate convergence.

$$W = (W_0 - W_1)\frac{(N_{\max iter} - N_{iter})}{N_{\max iter}} + W_1 \qquad \text{(eq. 4.3)}$$

As has been elaborated previously, **Pbest** of an individual swarm member, or *personal best*, signifies the position in search space in which this particular member achieved its best fitness (objective value vector) so far. Fittingly, **Gbest**, or *global best*, signifies the position that holds the overall best fitness up to this point in the search session, attained by any one of the members of the swarm. These two concepts may be fairly self-explanatory when it comes to single-objective optimization: each swarm member carries with it the memory of its best fitness achieved and where that occurred, whereas all swarm members are aware of where the best solution so far is located, and are drawn towards it. In multi-objective optimization, things are a bit more complicated: on one hand, there lies the common issue of solution comparison and ranking: which is best

among different solutions in *N*-dimensional space? How will it be possible to compare two individuals based on more than one conflicting criteria? How will Gbest or the individual Pbest be determined? Moreover, it is understood that there cannot be one Global Best by its strict definition, since we aim to achieve solutions as uniformly diverse as possible and to cover the biggest possible range of combinations of good fitness in different objectives. Therefore, in MOO, the Gbest concept is similar to Pbest in that, every swarm member carries its very own Gbest vector, that may not be the same for any other swarm member [5, 6, 7].

In this work, as for W, a linearly variant approach was adopted for both the *cognitive acceleration coefficient* $\mathbf{C_{cogn}}$ and the *social acceleration coefficient* $\mathbf{C_{soc}}$ (eq. 4.4, 4.5). A decreasing W and C_{cogn} along with an increasing C_{soc}, all within an appropriate range, [0.9:0.4], [2.5:0.5] and [0.5:2.5] respectively, yield better results [21, 27]. The choice of parameter values for this work was made along those lines, with no further parametric analysis conducted other than tests for validation of the above.

$$C_{\text{cogn}} = (C_{cogn,0} - C_{cogn,1}) \frac{(N_{\max iter} - N_{iter})}{N_{\max iter}} + C_{cogn,1} \quad \text{(eq. 4.4)}$$

$$C_{\text{soc}} = (C_{soc,0} - C_{soc,1}) \frac{(N_{\max iter} - N_{iter})}{N_{\max iter}} + C_{soc,1} \quad \text{(eq. 4.5)}$$

The increasing C_{soc}, decreasing C_{cogn} theme is in similar spirit to the exploration/exploitation scheme described earlier. It is understood that cognitive influence is linked to the earlier stages of convergence, where we aim for coverage of the greatest possible part of feasible space and social influence to the latter stages, where we want to make the most out of the experience already gained in order to capture the illusive optima.

4.2. Initialization

(From here onward, we address the Multi-Objective version of the proposed optimizer)

During the initialization phase, swarm members are assigned to randomly distributed locations in feasible space. The process is governed by a random number generator and thus, all performance-related data demonstrated later in this work are products of averaging the outcome of multiple sessions utilizing different random generator seeds, to

guarantee an independent of any randomness, dependable verdict.

Additional care is taken with the initialization of flight velocity of each particle: with Pbest fitness vector assigned infinite starting values and Gbest equalized with their respective members' starting location, assigning zero starting velocity is not an option. Also, an initial velocity that would drive the member outside feasible space, activating the *constraint operator*, must be avoided at all costs as it would severely hinder the algorithm's prospects. Velocity is therefore set equal to a percentage of the distance between the member's starting *chromosome* and the center of feasible space, and directed towards it. This percentage must be kept low, or the whole swarm will collapse towards the same area on the first generation. A mere 'push' is only intended.

It should be noted that this particular initialization procedure is far from common practice: Most PSO alternatives [5, 6, 15, 18, 21] set initial velocity at 0 and equal Pbest to its corresponding chromosome. The first evaluations that come as the very next step, determine the Gbest archive and the algorithm proceeds normally. In ref. [25], an overly sophisticated scheme is proposed, incorporating varied Probability Distributions and Low-Discrepancy sequences into the initialization of the swarm.

4.3. Elite-related operations

In this section we address the issues of *non-dominated solution sorting* or *elite sorting* and *solution diversity*, which are innate to any stochastic MOO algorithm, be that an EA, swarm-intelligence-based or other. Two representative examples of the state-of-the-art in Multi-objective Evolutionary Computation are shortly surveyed before we proceed to discuss the practices of this work:

- *Non-dominated Sorting Genetic Algorithm II* (NSGA II): Proposed by Deb et al. [3, 4], this algorithm is a revised version of the original NSGA also proposed by Deb et al. [2]. Both are based on several layers of classifications of the individuals. Before *selection* is performed, the population is ranked on the basis of non-domination: all non-dominated candidate solutions are classified into one category and receive a dummy fitness value, which is a function of the number of individuals dominated by each such solution. All candidates are likewise classified in additional layers, as this first group of classified individuals is ignored and another layer of non-dominated individuals is considered. The process continues until all individuals in the population are classified. The chance of a particular individual to be *selected* is proportional to this dummy value, which portrays its *strength* among other members of the population. The areas

upon which NSGA II is an improvement to the original NSGA are computational efficiency, the addition of elitism and the implementation of a diversity operator without requiring additional parameters to be input by the user (unlike the original NSGA and its 'fitness sharing' factor).

- *Pareto Archived Evolution Strategy (PAES)*: This algorithm was introduced by Knowles and Corne [23]. PAES consists of a (1+1) evolution strategy (i.e. a single parent that generates a single offspring) in combination with a historical archive that records some of the non-dominated solutions previously found. The utility of this archive is that of a reference point to which each individual is compared. Such a historical archive is the elitist mechanism adopted in PAES. However, an interesting aspect of this algorithm is the procedure used to maintain diversity which consists of a crowding procedure that divides objective space in a recursive manner. Each solution is placed in a certain grid location based on the values of its objectives (which are used as its 'coordinates' or 'geographical location'). A map of the said grid is maintained, indicating the number of solutions that reside within each grid sub-space. Since the process is adaptive, as in NSGA II, no extra parameters are required.

4.3.1. Non-dominated sorting

The proposed algorithm (**PA**) handles elite sorting in two phases, which take place during each generation: First, after all swarm members have undergone evaluation, it extracts **(fig. 4.1)** those that are found non-dominated among the current population, with regard to fitness during the current generation. Those individuals, flagged as '*GND*' (for '*generation-non-dominated*') for the remainder of this generation, are obviously the only ones among the current population that have a chance to enter the updated elite archive, or Pareto front, as being globally non-dominated also presupposes not being dominated by any in the current swarm. Then, the *GND* particles join the current elite archive in a unified temporary archive, and all compete against each other **(fig. 4.2)** for Pareto dominance, forming the new Pareto front after all dominated solutions, old and new, are ousted **(fig. 4.3, 4.4, 4.5)**. The simpler and more straightforward alternative of immediately subjecting the entire current swarm to a Pareto dominance test against the entire elite archive was avoided to promote the overall computational efficiency. The dual phase scheme yields the additional benefit that the Pareto front update process is easier to monitor.

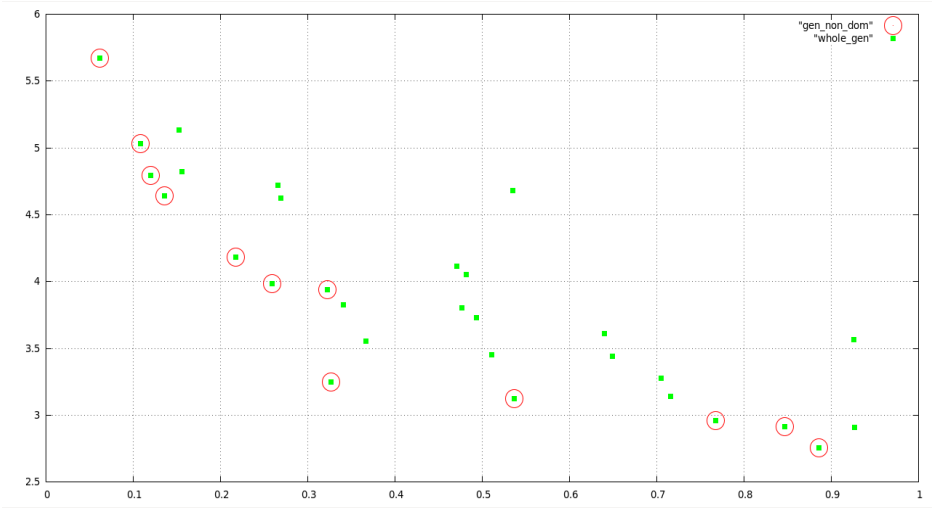


Fig. 4.1 An entire 'generation' of the swarm, illustrated in the domain of objectives. Circled are the GND particles (Case ZDT-1, axes refer to its two objectives, abscissa for F_1, coordinate for F_2).

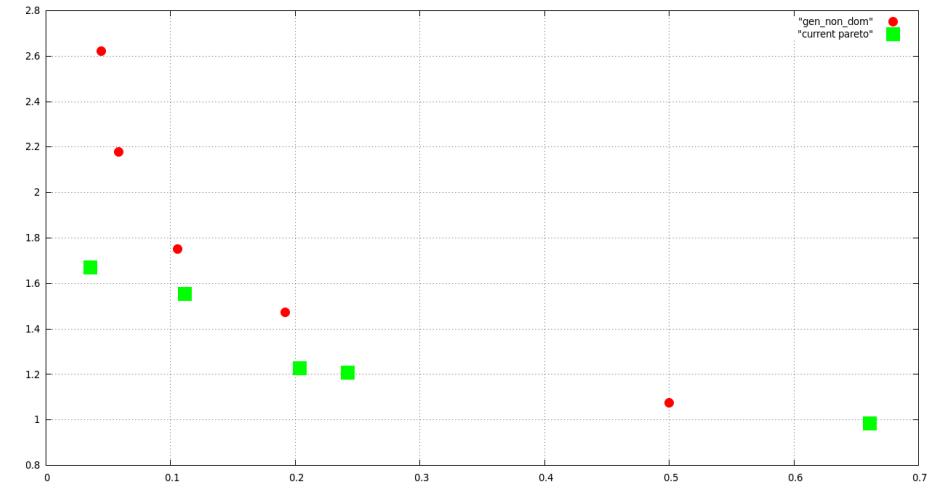


Fig. 4.2 The non-dominated of a given generation are joined by the current elites (Case ZDT-1, axes refer to its two objectives, abscissa for F_1, coordinate for F_2).

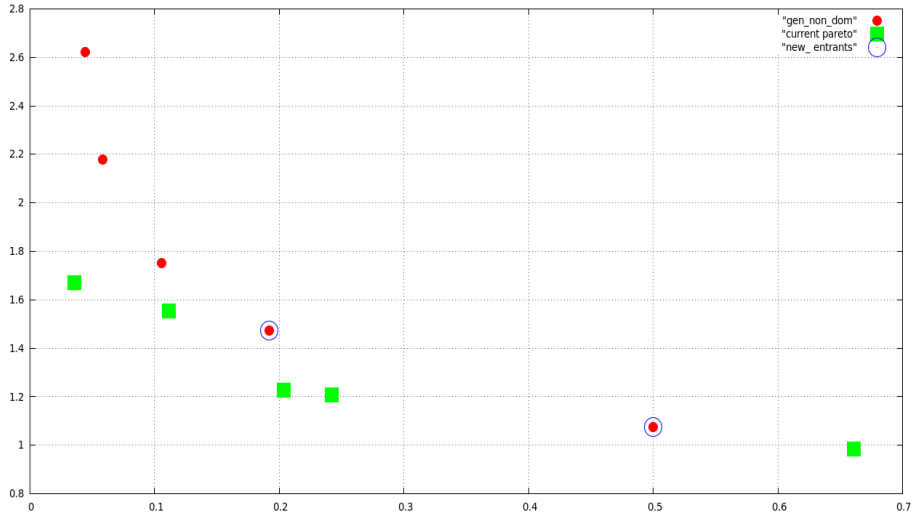


Fig. 4.3 The non-dominated of a given generation alongside the current elites (Case ZDT-1, axes refer to its two objectives, abscissa for F_1, coordinate for F_2). Circled are those among the GND who are also globally non-dominated and will be in the new, updated elite archive.

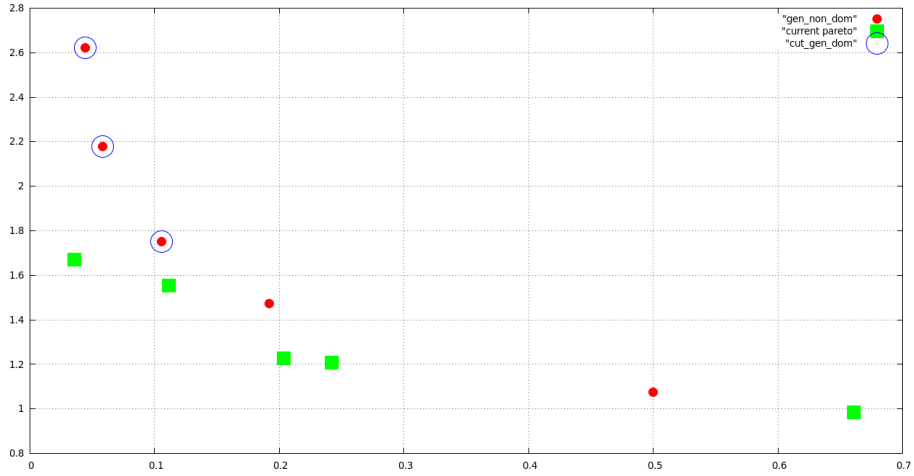


Fig. 4.4 The non-dominated of a given generation alongside the current elites (Case ZDT-1, axes refer to its two objectives, abscissa for F_1, coordinate for F_2). Circled are those among the GND who fail the dominance check and will not be in the new, updated elite archive.

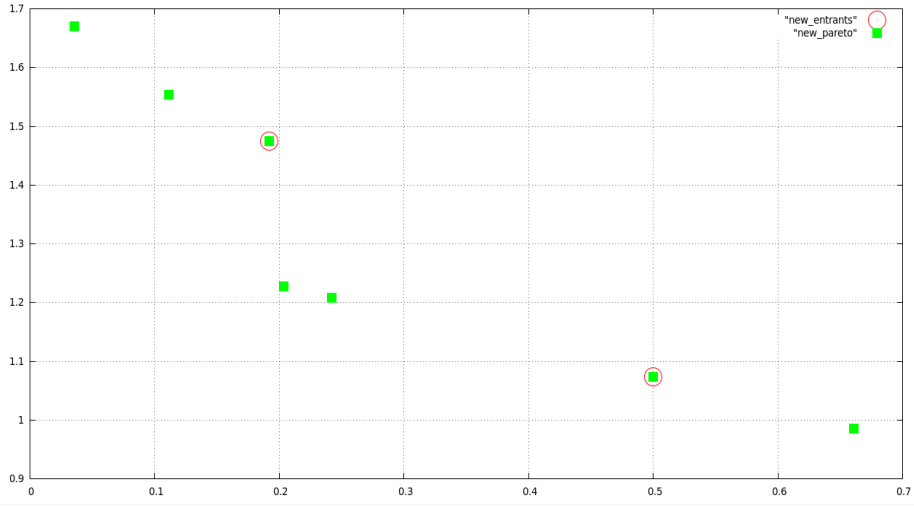


Fig. 4.5 The new, updated elite archive (Case ZDT-1, axes refer to its two objectives, abscissa for F_1, coordinate for F_2). Circled are the new entrants, compared to the previous generation.

4.3.2. Elite Spacing

By default, the size of the elite archive cannot exceed a given value, specified a priori by the user. Upon reaching this threshold, and every time that a newly formed Pareto front contains more individuals than dictated by the elite archive size limit, *elite spacing* is engaged, to indicate which members will be retained and which dropped. *Elite spacing* not only serves the purpose of decreasing the archive's dimension, but is also responsible for the disparity and quality spread of elite solutions in objective space. Therefore, it boasts a certain degree of sophistication, rather than choose randomly.

Our aim is to 'cut' individuals from relatively crowded areas of the front, as opposed to scarcer parts. The closer two elite individuals are to one another in terms of fitness, the greater the chance that only one will be retained, as they present a very similar set of objective function values and we seek as many differently 'weighted' final solutions as possible. The adopted scheme is labeled '*density'* and is similar in practice to most popular diversity operators, like NSGA and NSGA II [2, 3, 4].

Elite individuals are ranked based on closeness to their nearest elite neighbor on the front. The *density* of a particular member is derived exclusively from the distance, measured in the objective domain, to its closest partner; in fact it is equal to that distance. After density has been calculated for all current Pareto front occupants, the one with the lowest value is ousted from the front. The process is repeated until we reach the desired elite

archive dimension. Prior to those calculations, distances are of course non-dimensionalized by absolute objective value range (derived from the absolute greatest and lowest value difference documented so far). Please note that the most 'dense' individuals always appear in pairs as, if the smallest distance documented during an iteration is that of member A from member B, such is also the case for the distance of B from A. Deleting both A and B would not have the desired effect, which justifies that only one individual is removed and the procedure is repeated from scratch, i.e. distances calculated all over again.

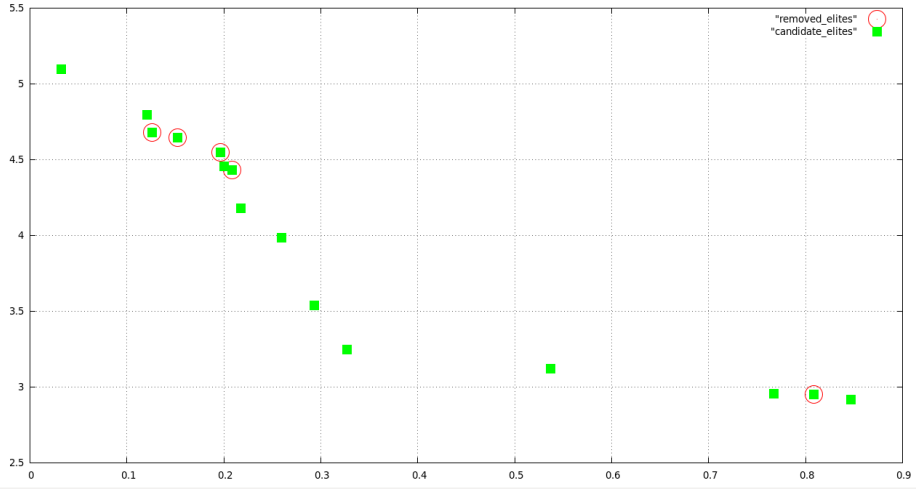


Fig. 4.6 The current Pareto front through the *spacing* process (Case ZDT-1, axes refer to its two objectives, abscissa for F_1, coordinate for F_2). Circled are the individuals to be dropped.

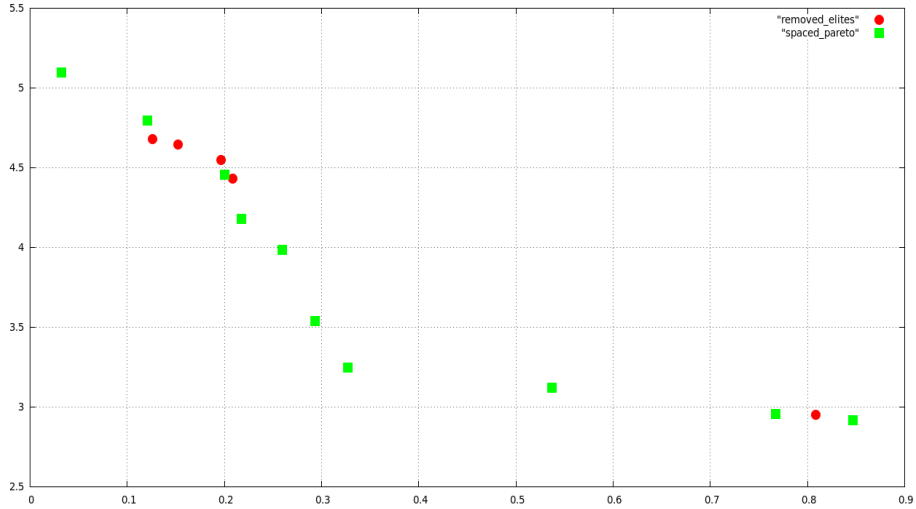


Fig. 4.7 The updated Pareto front through the *spacing* process (Case ZDT-1, axes refer to its two objectives, abscissa for F_1, coordinate for F_2).. As dots appear the particles which were dropped.

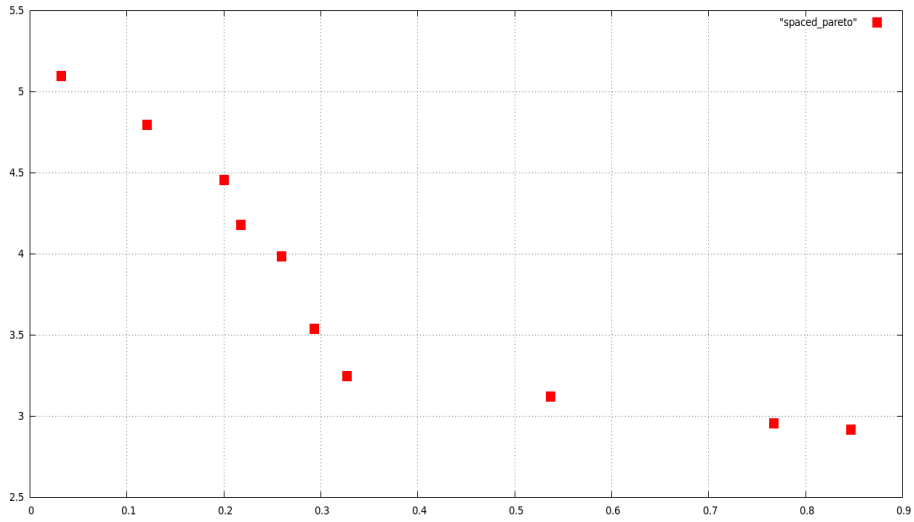


Fig. 4.8 The ultimate Pareto front, post *spacing*, limited by the maximum elite archive size (here, 10). (Case ZDT-1, axes refer to its two objectives, abscissa for F_1, coordinate for F_2)

4.4. Gbest & Pbest assignment

This section of the algorithm is of great importance: It signifies the transition to MMO, as the update of Gbest and Pbest archives is self-explanatory in SOO, whereas in multi-objective cases it is, in the author's opinion, a vast difference maker and sets apart a simply functional MO PS Optimizer and a really competitive alternative. Hence, this is where most novelty of this work lies. It should also be noted, that there is no direct equivalent to this phase in Evolutionary Optimization, unlike what has been presented up to this point. Let us shortly review the most standout attempts to extend PSO's utility to MOO:

- The algorithm of Parsopoulos and Vrahatis [22]: This algorithm adopts an aggregating function (three types of approaches were implemented: a conventional linear aggregating function, a dynamic aggregating function and a special weighted aggregation approach in which the weights are varied in such a way, that concave portions of the Pareto front can be generated).

- *Dynamic neighborhood PSO* proposed by Hu and Eberhart [7]: The dynamic neighborhood PSO has limited utility for problems of a maximum of 3 objectives. This concept assumes considerable prior knowledge in terms of the problem's properties. Instead of a single Gbest, a local Lbest is obtained for each swarm

member that is selected from the closest two swarm members. The closeness is considered in terms of one of the objectives, while the selection of the optimal solution from the closest two is based on the other objective. The selection of the objectives for obtaining the closest neighbors and local optima is usually based on the knowledge of the problem being considered for optimization. Usually the simpler objective is considered for the detection of the closest partner. A single Pbest solution is maintained for each member that gets replaced by the present solution only if the present solution dominates the current Pbest. Regardless of its limited utility, Dynamic Neighborhood is a novel and very interesting concept.

- *MOPSO*, an alternative proposed by Coello et al. [5, 6]: This very interesting proposal maintains an external non-dominated solution repository. Hypercubes of the objectives space explored so far are generated, and each particle is located using hypercube topology to define a new coordinate system. Of all those hypercubes, those containing particles that are also in the repository are ranked based on how many such particles they contain (the fewer, the higher up the order they get). Gbest particle assignment is made randomly from one of those hypercubes, after they are subjected to weighted roulette selection.

As was clarified earlier, every individual swarm member is assigned not only his own Pbest, but Gbest also, in direct contradiction to the notion of 'One Globally Best Solution'. To properly accommodate the pursuit of a spread out Pareto front would not be possible, if only one elite was appointed Global Best (remember, Gbest's act as gravitational points during swarm position update). The process of selection of a suitable Gbest for every current swarm member is designed on the basis that: *i)* it must not interfere with elite spreading and *ii)* it must endorse the progress of every individual *deterministically*, to some extent, i.e. the choice of Gbest must consider the direction in which each swarm member has the greatest potential. Regardless of what method we opt for, Gbest's are always picked out from among globally non-dominated solutions, namely *elites*. Therefore, from now on, we can address Gbest's simply as index numbers of the elite archive, instead of as stand-alone variable and fitness vectors.

4.4.1. Gbest selection alternatives

4.4.1.1. Overview

A few different approaches were tried and, even though we have settled down to one and fully endorse it, three of them will be presented in detail, to showcase how each added feature enhances the overall performance, followed by suggestions on possible further improvements.

i. 'Roulette' method

Exactly as the title suggests, this method assigns a Gbest to each and every swarm member simply by randomly selecting any one elite from the current Pareto front, without any determinism involved in any way. This randomness guarantees a very good degree of disparity in the final solution front, but due to the lack of a 'smart' search feature, it falls behind in convergence speed and overall performance compared to other, more sophisticated alternatives **(fig. 4.13)**.

ii. 'Proximity' scheme

This method is deterministic to some extent, as it assigns as Gbest that elite, which is closest, in terms of fitness, not in search space, to the examined swarm member **(fig. 4.9)**. In other words, the non-dimensionalized distance of the said member from each and every current elite in the domain of objective functions is calculated, and the one found closest is appointed Gbest of this member (in **fig. 4.9** the selected elite appears to be the second closest, but that is not the case after the non-dimensionalization of distances). This is an attempt to locate a 'projection' of the said swarm member on the Pareto front, which is theoretically its direction of natural progress. Depending on the shape of the Pareto front, its curvature etc., the proximity scheme does not always work as intended: On close inspection of **fig. 4.10**, we observe that of all non-dominated solutions, the red particle is closest to the circled elite, which will consequently be picked as Gbest. However, as is easily noticed, the Gbest which was assigned on the basis of proximity does not dominate the adjacent swarm member (red particle). Rather, it only is 'better' according to one of two objectives. To the left of the chosen elite lies another solution which completely dominates the red particle **(fig. 4.11)** and would, in that sense, be a better option, as moving towards it would universally improve the examined individual.

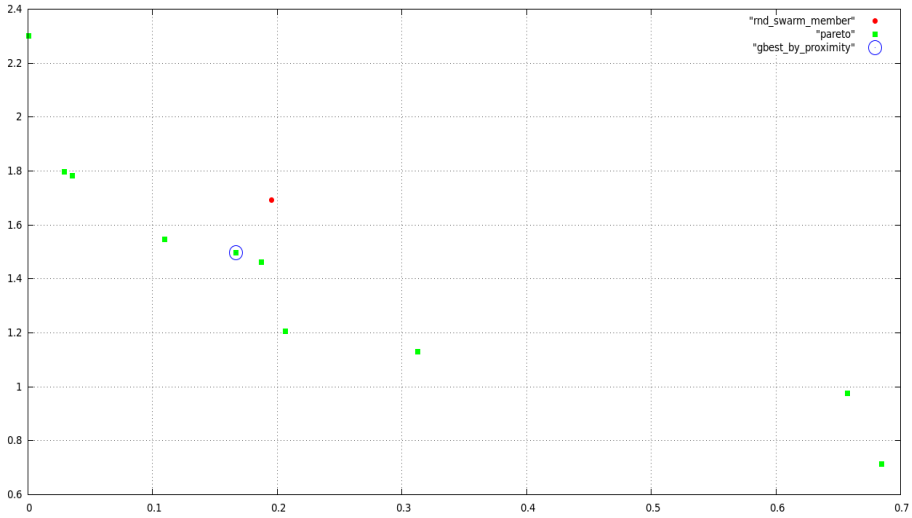


Fig. 4.9 Demonstration of the *proximity* scheme. Circled is the selected Gbest (Case ZDT-1, axes refer to its two objectives, abscissa for F_1, coordinate for F_2).

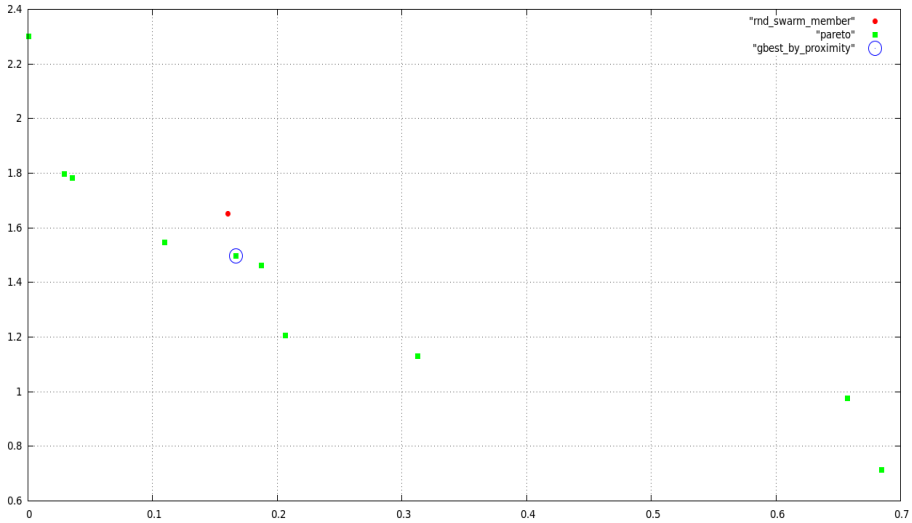


Fig. 4.10 The shortcoming of the *proximity* method. Circled is the selected Gbest, which, however, does not dominate the examined particle (Case ZDT-1, axes refer to its two objectives, abscissa for F_1, coordinate for F_2).

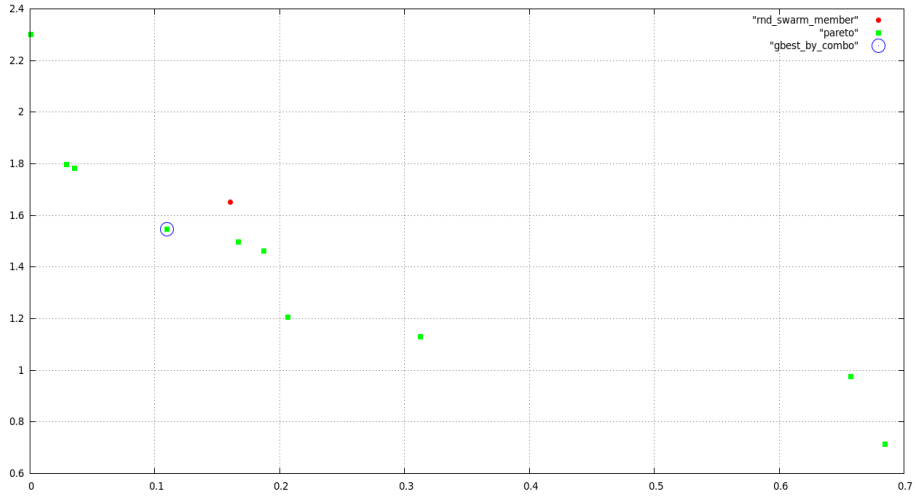


Fig. 4.11 Circled is the manually selected and recommended Gbest, as opposed to the one selected by the proximity scheme (Case ZDT-1, axes refer to its two objectives, abscissa for F_1, coordinate for F_2).

iii. Combo* method

* A combination of the principle of Pareto dominance and roulette selection.

The last and most advanced scheme adopted, and the one the final algorithm uses by default, is the combo method, in which roulette selection is also applied, but only among elites that dominate the examined swarm member. Every single one of the solutions in the elite archive are checked for dominance over the examined individual, and out of those which pass **(fig. 4.12)**, one is selected randomly. This method is already an improvement compared to the previous one, as the picked Gbest is guaranteed to be an overall better solution than the examined swarm member. By using roulette selection, we also allow for the stochastic element to be present.

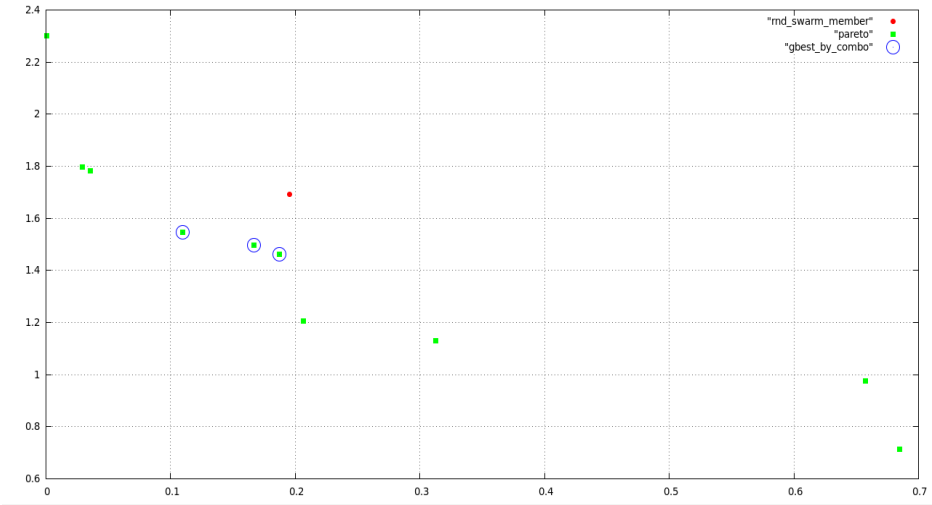


Fig. 4.12 Demonstration of the *combo* method. Circled are the candidates for the roulette phase of the method (Case ZDT-1, axes refer to its two objectives, abscissa for F_1, coordinate for F_2).

4.4.1.2. Direct comparison

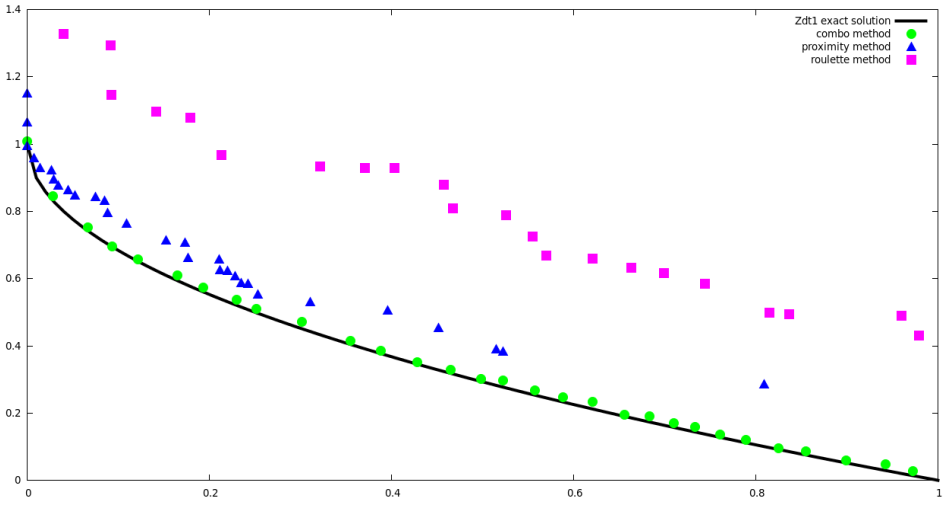


Fig. 4.13 Direct performance comparison between the three presented methods (Case ZDT-1, axes refer to its two objectives, abscissa for F_1, coordinate for F_2), conducted using default settings and identical population data (swarm size S=50, max. Elite archive size=35).

A first performance comparison, the results of which are shown in **fig. 4.13**, confirmed the above speculation as to the potential of each scheme. The MO version of the algorithm was set for 5000 maximum evaluations, featuring a 50 particles population and

permitting a maximum elite archive size of 35, with every initialization and swarm update related parameter at default (more on this default set to follow). As predicted, *roulette Gbest selection*, despite achieving a very good disparity of final solutions, is weak in terms of convergence speed and general performance. The *Combo* method is evidently the better option, followed by *proximity*, as it not only boasts the fastest convergence but also gives a nicely spread out ultimate set of solutions, which is not the case for the proximity operator, which despite converging at a competitive rate, fails in achieving a decent disparity in the final solution front. This can be attributed, in part, to the shortcoming described earlier. It should be duly noted that the performance gap between *proximity* and the *Combo* is larger than illustrated in **fig. 4.13**, as the latter is limited by the analytically calculated optimum of ZDT-1, to which it converged before reaching the threshold of 5000 evaluations.

4.4.1.3. Suggestions for immediate improvement

Already, potential improvements to the algorithm are evident: By eliminating or limiting the leftover stochastic element in favor of an even more deterministic approach, performance can be further enhanced. That can be achieved through combining proximity and dominance and dropping roulette selection:

- ✓ All elites dominating the examined swarm member are subject to the proximity operator, which is responsible for the final pick, based on closeness.

- ✓ Elite solutions are ranked on proximity, in increasing order of distance from the examined swarm member. In the same order, they are all checked for Pareto dominance upon the said member. The first one to pass this check (the closest dominant elite) earns Gbest status.

Even further improvement can be achieved by implicating the *density* of an elite candidate for Gbest status: Of all elites dominating the examined swarm member, the one with the highest *density* value is picked. Not only does this criterion lead to an elite which is a definite upgrade compared to said swarm member, but it is also a substantial step in the direction of solution spread and diversity!

4.4.2. Pbest update alternatives

Things are much simpler when it comes to the update of the archive of personal best solutions, or 'cognitive memory', as it is referred to in Swarm Intelligence literature. On

most occasions, the simple principle of Pareto dominance is applied: The fitness of the current solution is compared to its current personal best vector; should the new solution completely dominate the Pbest, then it takes its place. Accordingly, if it is dominated by the reigning Pbest, nothing is changed. When neither is the case, i.e. when neither of the two dominates the other, we are presented with three options:

- ✓ Opt for the new solution. This is the 'confident' alternative, as it invests heavily on newer solutions to drive the optimizer forward.

- ✓ Retain the current Pbest. This is the 'play safe' option, which will only alter the Pbest archive once presented with a definite upgrade.

- ✓ Randomly select one of either and proceed.

All three alternatives were tried on two different problems (cases ZDT-1 and ZDT-3), with no clear winner. Therefore, none receive the author's recommendation. The algorithm is set at the 1st option by default, whereas the 2nd and 3rd are the most common choices in relevant literature [5, 6, 7, 15, 18, 21, 22].

4.5. Additional features

4.5.1. Constraint operator

Optimization problems are generally subject to constraints; apart from the natural constriction of variable values within an upper and lower bound, additional conditions must apply for a solution to be acceptable. Such conditions may include intermediate quantities of the problem, the decision variables themselves, or even the objectives. A typical issue commonly handled by the constraint operator of an optimizer that may otherwise not be clearly classified under constraints, is when we intend for a variable to take values from within an open '()' range, i.e. $\mathbf{b_{lower}}<x_i<\mathbf{b_{upper}}$. This range will be regularly declared as closed in the variable declaration section of the algorithm's input file, and it is then up to the constraint operator to reject values equal to the lower and/or upper bounds.

The K_c constraints of any problem must be expressible in this particular inequality format:

$$C_i(\vec{x}) \le C_i^{perm} \quad , \quad i = 1, K_c \qquad \text{(eq. 4.6)}$$

Where $C_i(\vec{x})$ is the Constraint function value corresponding to a variable vector $\vec{x}$, calculated by the very same external evaluation software that calculates objectives, and C_i^{perm} the permitted upper threshold, beyond which, solution $\vec{x}$ is considered as having failed the constraint. It is possible for any condition to be converted into the above format:

- $$C_i(\vec{x}) \neq C_i^{\mathrm{X}} \;\Rightarrow\; \left|C_i(\vec{x}) - C_i^{\mathrm{X}}\right| \geq \varepsilon \;\Rightarrow\; \text{(eq. 4.7)}$$

$$\Rightarrow \begin{cases} C_i(\vec{x}) - C_i^{\mathrm{X}} \geq \varepsilon \;, & \text{if } C_i(\vec{x}) > C_i^{\mathrm{X}} \\ C_i^{\mathrm{X}} - C_i(\vec{x}) \geq \varepsilon \;, & \text{if } C_i(\vec{x}) < C_i^{\mathrm{X}} \end{cases} \Rightarrow$$

$$\Rightarrow \begin{cases} -C_i(\vec{x}) \leq -\varepsilon - C_i^{\mathrm{X}} = C_i^{perm} \;, & \text{if } C_i(\vec{x}) > C_i^{\mathrm{X}} \\ C_i(\vec{x}) \leq C_i^{\mathrm{X}} - \varepsilon = C_i^{perm} \;, & \text{if } C_i(\vec{x}) < C_i^{\mathrm{X}} \end{cases}$$

- $$C_i(\vec{x}) = C_i^{\mathrm{X}} \;\Rightarrow\; \left|C_i(\vec{x}) - C_i^{\mathrm{X}}\right| \leq \varepsilon \;\Rightarrow\; \text{(eq. 4.8)}$$

$$\Rightarrow \begin{cases} C_i(\vec{x}) - C_i^{\mathrm{X}} \leq \varepsilon \;, & \text{if } C_i(\vec{x}) > C_i^{\mathrm{X}} \\ C_i^{\mathrm{X}} - C_i(\vec{x}) \leq \varepsilon \;, & \text{if } C_i(\vec{x}) < C_i^{\mathrm{X}} \end{cases} \Rightarrow$$

$$\Rightarrow \begin{cases} C_i(\vec{x}) \leq \varepsilon + C_i^{\mathrm{X}} = C_i^{perm} \;, & \text{if } C_i(\vec{x}) > C_i^{\mathrm{X}} \\ -C_i(\vec{x}) \leq \varepsilon - C_i^{\mathrm{X}} = C_i^{perm} \;, & \text{if } C_i(\vec{x}) < C_i^{\mathrm{X}} \end{cases}$$

- $$C_i(\vec{x}) \geq C_i^{\mathrm{X}} \;\Rightarrow\; -C_i(\vec{x}) \leq -C_i^{\mathrm{X}} = C_i^{perm} \quad \text{(eq. 4.9)}$$

where $\boldsymbol{\varepsilon}$ equals 10^{-14} or any tiny quantity, for that matter.

The consequences of failing a constraint are reflected on the ultimate fitness of the particle, which is appropriately modified, dealing a severe blow to the candidate's competitiveness. It is common practice to assign an infinite fitness (ex. 10^{14}) value to a candidate for failing even one of K_c constraints. In this work, a somewhat more 'forgiving' approach is adopted, where the constraint limits are relaxed, containing 'penalization' within reasonable lines for those candidates that have only just surpassed the constraint thresholds. A second, slightly higher threshold C_i^{relax} is introduced, set by the user according to his best knowledge or judgment of the problem's specifications. Fitness of an individual is subject to the following modifications, subsequently to the calculation of constraints:

$$F(\vec{x}) = \begin{cases} F(\vec{x}) & , \text{ if } C_i(\vec{x}) \le C_i^{perm} \\ F(\vec{x}) + h(C_i(\vec{x})) & , \text{ if } C_i^{perm} < C_i(\vec{x}) \le C_i^{relax} \\ +\infty & , \text{ if } C_i(\vec{x}) > C_i^{relax} \end{cases} \qquad \text{(eq. 4.10)}$$

$$\text{where:} \quad h(C_i(\vec{x})) = \exp\left(\frac{C_i(\vec{x}) - C_i^{perm}}{C_i^{relax} - C_i^{perm}} \right) \qquad \text{(eq. 4.11)}$$

The fitness of any particle can additively receive multiple h increments, if the particle has entered the relaxed area in more than one constraint functions. It is self-explanatory that if a particle falls under case (3) of eq. 4.10, it is automatically excluded from the processes of solution sorting and Gbest assignment.

The algorithm can benefit massively from this 'forgiving' approach, especially in heavily constricted problems where the algorithm will have difficulties even assembling a dependable first population of acceptable solutions, from which to proceed. It is not an infrequent occurrence to have an optimization session terminated before it has even set off, due to all initial particles failing to be established as 100% viable solutions. Crucially, if one considers that an optimum often lies at the edge of feasible search space – a problem's constraints are effectively limiting its potential of resolution – it is understandable that, in the long run, we might be rewarded for offering these *pariahs* of the swarm an opportunity to contribute, even from a slightly (h penalty) disadvantageous position.

4.5.2. The SHUFFLE operator

4.5.2.1. Overview

Inspired by mutation operators widely utilized in Evolutionary Computation, the *shuffle* technique was developed to step in at a certain checkpoint and, by stirring the algorithm's database, to reintroduce stochasticity in the search process. It essentially re-initializes the swarm, while retaining records of the progress of the algorithm as far as solutions are concerned. At the same point, it detects areas in search space that have been relatively neglected so far, and ensuing search is made to focus on these areas, by encouraging particles to travel in specific directions.

Various reasons call for the implementation of *shuffle* or similar measures: Due to the nature of the objectives, constraints and other unique characteristics of the problem at hand, the algorithm may encounter obstacles in exploring certain areas of search space [23], which will result in a final set of solutions that is not as diverse as we would like, containing a plethora of, very similar to one another, solutions and failing to locate those illusive 'hidden' optima. Case ZDT-3, presented later in this work, is a perfect example of such a problem, as its convex, non-contiguous analytic solution can be quite a challenge to capture. Moreover, *shuffle* seems to limit the impact that random number generation has on the algorithm's behavior, granting added consistency. Without it, PSO largely depends on its initialization. *Shuffle*, among other things, re-initializes the swarm, so the final outcome is now a product of more than one events of stochasticity.

The innate *premature convergence* that PSO 'suffers' from [9, 17, 24], as opposed to other optimization heuristics that proceed slower and explore more thoroughly, GA's being a good example thereof, poses an additional obstruction. On the other hand, this increased rate of progress of PSO is a privilege, as it secures a decent result early on, allowing us to 'spend' evaluations on the *shuffle* feature, to fine-tune our solution set and take a second look at problematic areas of search space. Let be emphasized that Pareto front refinement is the main problem in the development of a PSO optimizer. As we approach the final stages of the optimization process, it is not uncommon for a large percentage of the swarm to have come to a halt: Individuals that find themselves in a non-dominated position are effectively their own Gbest and Pbest. That means they only rely on their momentum factor to keep them from stopping completely and further progress, if any, is extremely slow. When this phenomenon appears, some drastic measure has to be taken to keep the optimizer in motion, and *shuffle* is one such measure. One such case is **fig. 4.14**, where not much more can be done as far as capturing the analytically calculated optimum

curve. However, solution diversity, especially towards the rightmost edge, could have improved substantially, if exploration of that part had been prioritized.

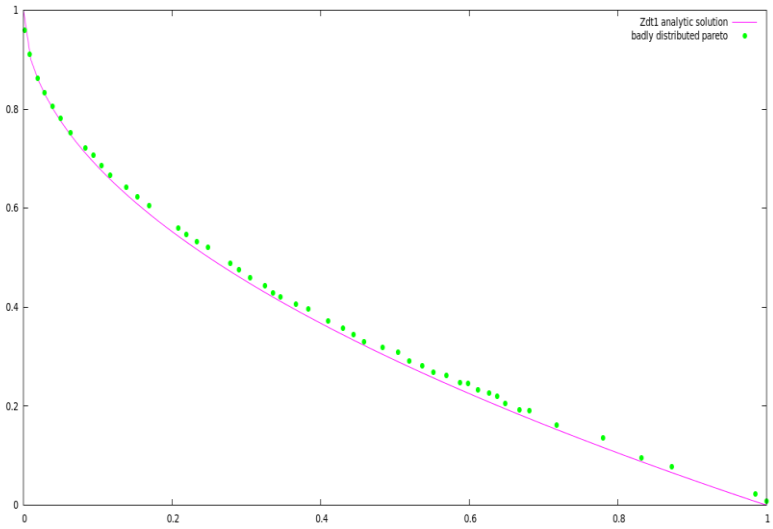


Fig. 4.14 A typical situation where shuffle is missed (Case ZDT-1, axes refer to its two objectives, abscissa for F_1, coordinate for F_2). The PSO optimizer has almost fully converged to the sought front, but the elite solution set suffers from imbalanced diversity (observe the rightmost edge).

The operator is first engaged after a certain % of maximum allowed evaluations have been completed. It is additionally engaged once more before the termination of the algorithm and only if a certain solution disparity criterion is also met, namely when the difference between the density of the most and least dense areas on the non-dominated front surpasses a given threshold. Simply put, if the density value of the elite individual with the lowest such value becomes lower than X times that of the one with the highest such value, the current non-dominated front is considered to be too imbalanced, and *shuffle* is re-engaged. Two *shuffle* sessions cannot occur too close to one another, as the algorithm must be given time to progress from the 1st shuffling or it will not culminate. The algorithm must also have achieved some degree of convergence before the 2nd *shuffle* session or it will occur under inappropriate circumstances (ex. it will be difficult to identify the less dense areas in need of further exploration if the algorithm is still in the wake of the 1st *shuffle*). Therefore, a necessary minimum of in-between generations is imposed.

As was mentioned in the introduction, apart from the re-initialization, there is a special mechanism that conducts highly targeted search, depending on the condition and progress

of the optimizer at the point of *shuffle* engagement. This mechanism recognizes the areas that require more attention and encourages the swarm to prioritize them during exploration. This identification is based on measurements of solution crowding along the non-dominated front. It ranks elite archive members on their density values and selects a given number of those with the highest such value **(fig. 4.15)**. These solutions are essentially from the scarcer parts of the Pareto front and represent the *neglected* areas in space. As you can observe in **fig. 4.16**, where both the non-dominated front and the swarm's fitness are illustrated, the areas that have been chosen do not only feature elite scarcity, but are also being explored by a low percentage of the population. The optimizer must aim for the neighborhood of these solutions in the ensuing generations, which is why they are branded '*gravity points*'.

The re-initialization phase includes relocating the swarm in space in a completely random fashion **(fig. 4.17)**. Velocity vectors are retained, as is the elite archive. The list of Gbest indexes is not altered or emptied, but is left to be updated by the normal flow of the algorithm. The Pbest archive is completely renovated, and this is the core of the *shuffle* process: Each swarm member's Pbest (parameter vector and fitness) is equalized to a random gravity point, similarly to how Gbest's were allocated randomly by the *Roulette* scheme, in the previous segment. Gravity points are, in this way, directly implicated in the swarm update process via the cognitive factor in the PSO formula. Thus, the flight of the swarm is directed towards the neglected areas **(fig. 4.17-4.20)**.

The exact number of the said 'low density' elites that are picked out as gravity points is up to the user, and is input as a percentage of the elite archive size. A low such percentage (<10%) is recommended as the *shuffle* scheme is rather intended to drastically focus on the few scarcest areas. For regular problems, with a contiguous, non-convex pursued front, 2-5 gravity points are more than adequate. For more complicated cases, an excellent example of which is the ZDT-3 case examined in this work, where the pursued front is complicated, a **maximum** of 7-8 gravity points may prove suitable. A greater number would best be avoided as it will result in great disparity of gravity points along the current front and the *shuffle* scheme having a minor effect compared to the intended. That is better comprehended if we take a look at the effect of the relocation of Pbest's: In the few generations that immediately follow the shuffle call, the whole swarm is given a strong push towards said area(s) **(fig. 4.18-4.20)**. Gbest's take on according values and the phenomenon is preserved long enough to allow sufficient exploration, for a more 'just', better laid-out Pareto front **(fig. 4.21)**. This strong drive towards gravity points can only be taken full advantage of, if those points are few.

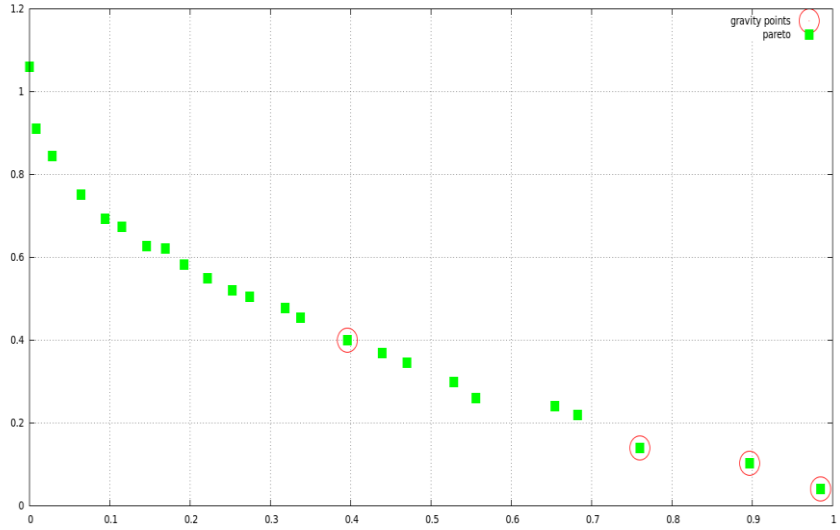


Fig. 4.15 The Pareto front at the point the shuffle operator is first engaged. Circled are the non-dominated individuals designated as 'Gravity Points' (Case ZDT-1, axes refer to its two objectives, abscissa for F_1, coordinate for F_2).

In our demonstrative run, the full benefit of the *shuffle* operator is already evident a mere 10 generations after it is engaged: **Fig. 4.21** shows the non-dominated front along with the swarm particles' fitness. The swarm itself is considerably more spread out, and the imbalances in the elite set (with a size of 25 individuals) have been addressed, as it now features excellent disparity and solution diversity, which is also the case for the ultimate result of the minimization procedure, after the completion of the given maximum evaluations **(fig. 4.22)**.

4.5.2.2. Suggested improvements

- ✓ A shuffle operator that affects a percentage of the swarm particles, instead of them all. This will serve to reduce the current operator's radicalism: A portion of the swarm will resume normally and another (a relatively small percentage) will be shuffled, in an attempt to locate he optima that have eluded the swarm thus far.

- ✓ Implementation of other popular schemes from Evolutionary Computation, such as Mutation (which has already appeared in PSO, as was first proposed by van den Bergh [8] and put to practice by Coello et al. [6] and others [18,21,27]).

- ✓ Development of similar, only milder refinement measures that will complement the regular flow of the algorithm, instead of violently interfering.

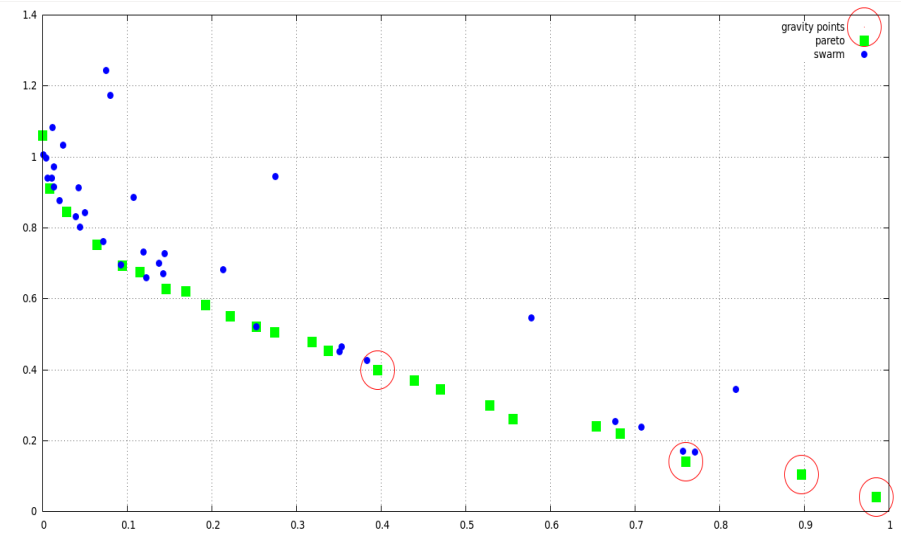


Fig. 4.16 The Pareto front and the swarm particles (illustrated in the objective domain) at the point the shuffle operator is first engaged. Circled are the non-dominated individuals designated as 'Gravity Points' (Case ZDT-1, axes refer to its two objectives, abscissa for F_1, coordinate for F_2).

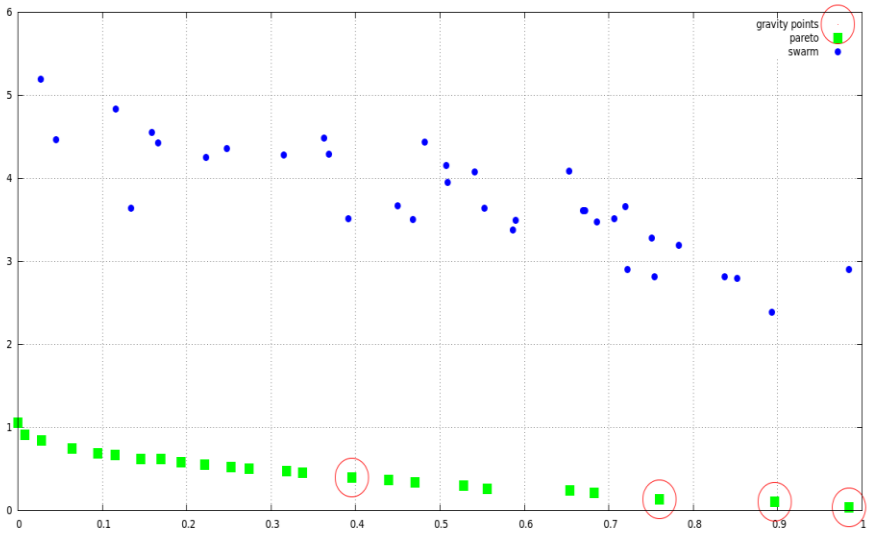


Fig. 4.17 The re-initialization phase. The swarm is randomly relocated in search space, away from optimum solutions, hence the huge deviation from the Pareto front (Case ZDT-1, axes refer to its two objectives, abscissa for F_1, coordinate for F_2).

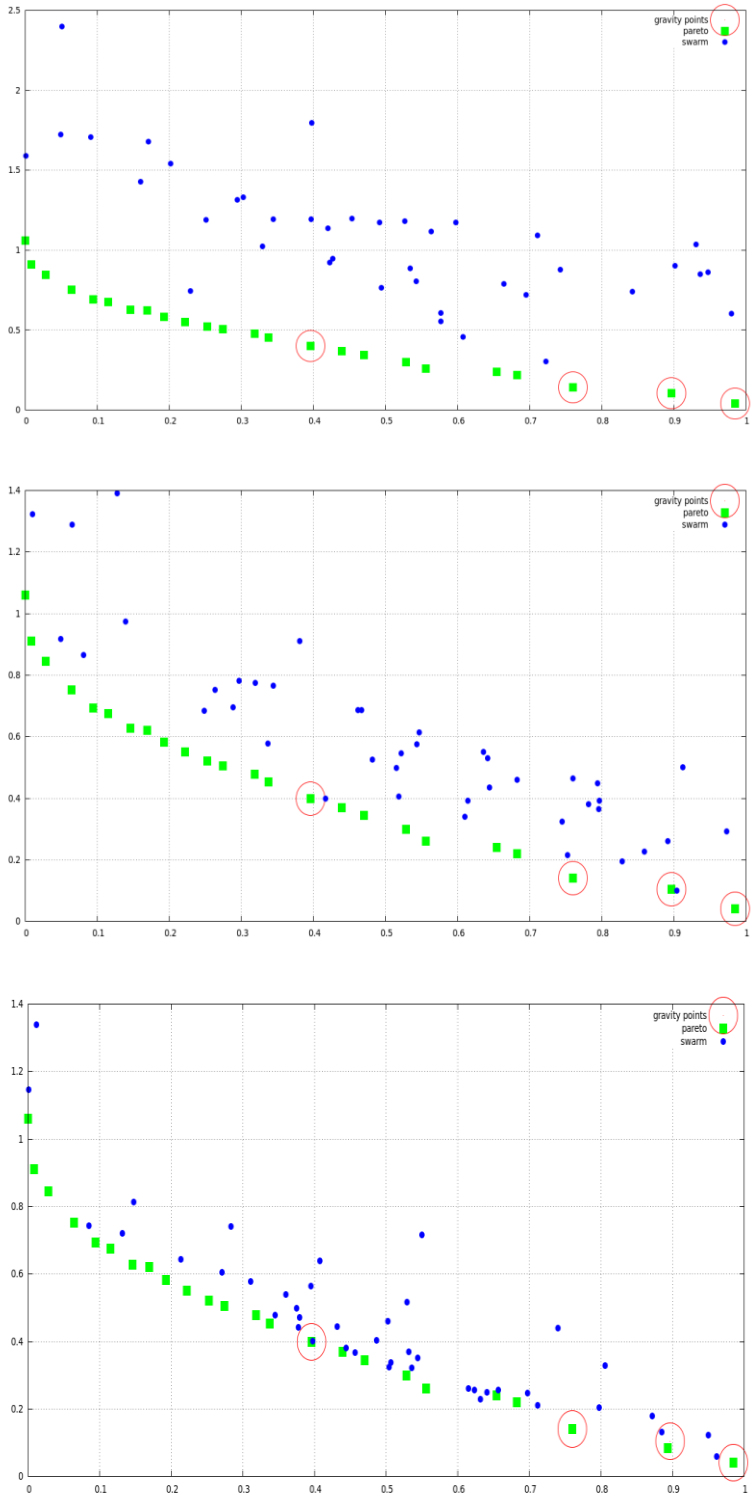


Fig. {4.18, 4.19, 4.20} The optimizer's progress since the point of re-initialization, as recorded with a 2-generation step. Notice the impact of Gravity points (Case ZDT-1, axes refer to its two objectives, abscissa for F_1, coordinate for F_2).

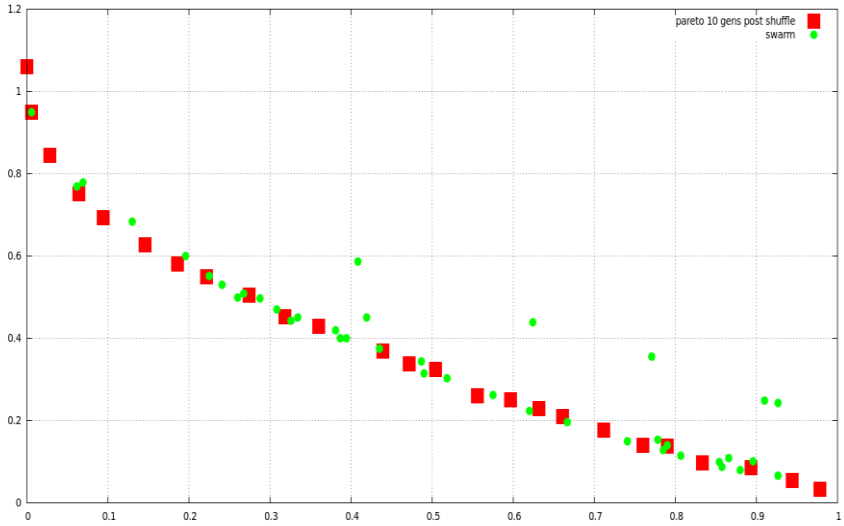


Fig. 4.21 The optimizer's progress a few generations after the re-initialization session (here, 10). (Case ZDT-1, axes refer to its two objectives, abscissa for F_1, coordinate for F_2)

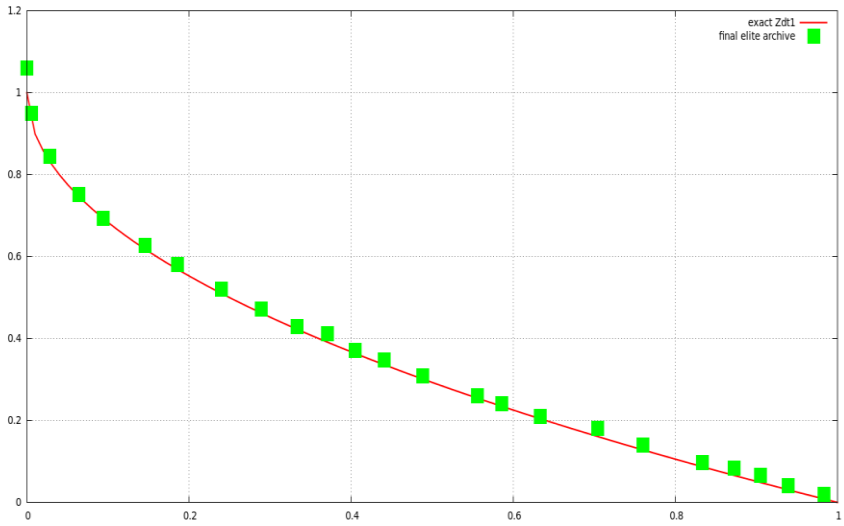


Fig. 4.22 The end result yielded by the optimizer, after the implementation of *shuffle* (Case ZDT-1, axes refer to its two objectives, abscissa for F_1, coordinate for F_2).

4.6. User input – Default settings

```
6        !  number of problem parameters, variable or otherwise           kfreeall
***     kd              bound1          bound2          !
        0               1               1
        1               0.1             1.2
        1               0.4             0.8
        1               0.0             1.0
        1               0.0             1.0
        1               4.0             6.0

80       !      swarm population size           npopl

1.2 .7   !      cognitive acceleration coeff. | starting and ending value      for the variant scheme
0.6 0.2  !      inertia weight                | start/end
1.4 2.1  !      social acceleration coeff.    | starting and ending value      for the variant scheme

50       !      # of consecutive iterations without progress beyond which to terminate              kstop

0.60     !      1st Shuffle threshold (% on total evals)                                          shufflex

0.15     !      2nd Shuffle threshold (leftover % of total evals)                      shufflec

5000     !  Maximum Evaluations                                                 maxeval

2        !  Number of Objectives                                                        nobj

80       !  Maximum elite archive size !                                        maxelite

20       !  Save elites every (#) generations                                   isave

33       !  iseed (random number generation)

0.2      !  initial velocity coefficient

2        !       - Total number of constraints

*** permitted thres.   relaxed thres.
        -1.0E-6         0.0
        0.999999        1.0
```

Fig 4.23 A sample input file, 'psoin.dat'. Every setting is followed by appropriate comments. Shuffle-related data and PSO tuning parameters are set at default (recommended) values.

Most settings available for the PA can be handled by the user and are accessible via 'psoin.dat', the program's expected input file (**fig. 4.23**). By editing the file's contents, the user can, among other things:

- ✓ Declare decision variables, including their respective range, lower and upper boundary, preceded by an integer 'switch', k_d, which specifies if this particular variable will actually vary {1} or remain constant{0} throughout this particular run, in which case an upper boundary needs not be set.

- ✓ Declare any active problem-related constraints. The user has to specify the regular constraint threshold and an additional relaxed threshold for each constraint.

- ✓ Specify a population and elite archive size.

- ✓ Specify the conditions of program termination.

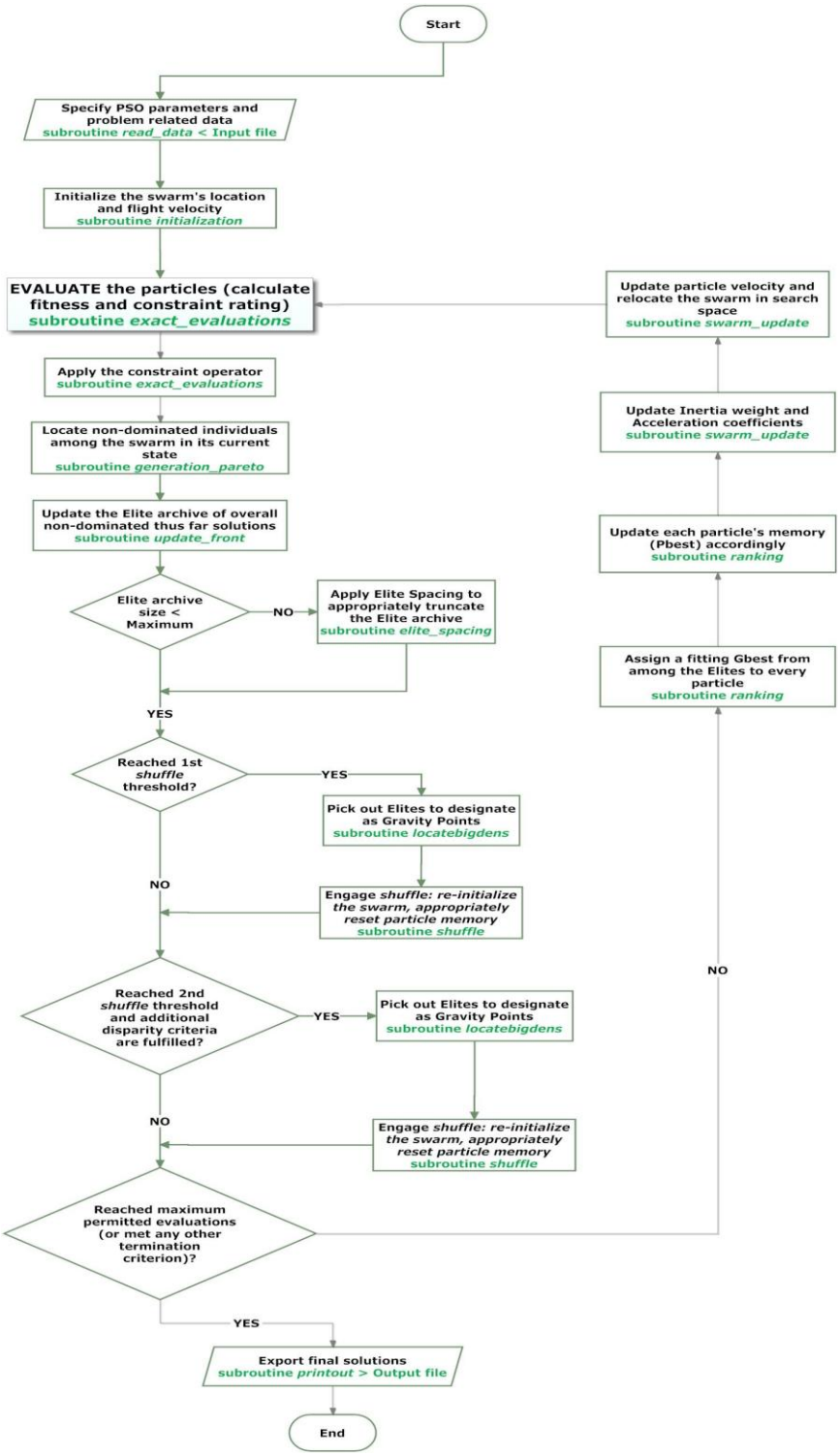
- ✓ Tune the optimizer, by setting the ranges within which W, C_{cogn} and C_{soc} will vary. Notice that inertia weight and C_{cogn} take on an average value of 0.4 and 0.95, respectively, whereas C_{soc} an average of 1.75. These -default- values deviate slightly from the recommendations of relevant literature, as featured in segment 3.3. A choice accounted for previously, it was made to boost the exploitative qualities of PSO, as opposed to its already considerable explorative capability.

Not all possible options available in the algorithm are accessible through the input file. A few can only be tweaked by editing the code itself and therefore require a recompilation for any changes to apply. For instance, the choice of Gbest selection method or Pbest update method is hardcoded, even though all mentioned techniques are available. Another two examples of parameters that are not directly available for tweaking are **densrule** and **gravperc**, both of which relate to the *shuffle* operator.

```
densrule=20.0,gravperc=.08
```

The first, set at 20.0, is the maximum order of difference between the lowest and highest density value among all non-dominated solutions which, if surpassed, will trigger a second shuffle session. The second, set at 0.08 is the percentage (8%) of members of the elite archive that can be given gravity point status over a single shuffle session. If this percentage of elites does not equal an integer number, it is automatically rounded up. Both of these two settings have been discussed earlier in this chapter. It is recommended for any such settings not included in the input file to be left in their preset state.

4.7. Complete Proposed Algorithm (PA) flow chart



5. Tests & performance validation

5.1. Benchmarking function test problems – Overview

There is a huge variety of mathematical problems, commonly oriented towards minimizing the value of complex functions, developed by Optimization researchers to serve as a benchmark for comparison of various new algorithms. These functions are the product of reverse engineering practices and are modeled after the typical obstacles that heuristics are bound to encounter when incorporated to solve challenging industrial 'real-world' problems, especially those with multiple objectives. Such problem peculiarities that may cause difficulty in convergence, even more so in converging to a well spread Pareto front are *multimodality,* optima front *convexity* and/or *discreteness, biased search spaces* and other; a benchmarking problem may involve a single such feature or a plethora thereof and test results not only yield a measure of overall performance but can also determine the suitability of one or another method for a particular type of problem.

In 1999, K. Deb [2], the creator of NSGA and NSGA II, suggested a systematic way of developing test problems for multi-objective optimization. Following these guidelines, along with E. Zitzler (creator of SPEA) and L. Thiele [36], they suggested six test problems under the label ZDT, the initials of the trio's surnames. Of those six problems two are featured in this work, ZDT-1 and ZDT-3, and are used to draw the first conclusions as to the proposed algorithm PA's capabilities and performance, especially alongside a representative EA. As is the case for all six ZDT problems, both examined cases are of 2 objectives:

$$\min(F_1(\vec{x}), F_2(\vec{x}))\text{ , where:}$$

$$\begin{cases} F_1(\vec{x}) = F_1(x_1) \\ F_2(\vec{x}) = G(x_2,\ldots,x_n)\cdot H\left(F_1(\vec{x})\cdot G(x_2,\ldots,x_n)\right) \quad , \quad n \geq 2 \\ x_i \in (0,1) \end{cases} \qquad \text{(eq. 5.1)}$$

Functions $G(x_2,...,x_n)$, H $F_1(\vec{x})\cdot G(x_2,...,x_n)$, as well as *n*, vary from problem to problem.

5.1.1. Case 'ZDT-1'

The simpler test function of the ZDT family, ZDT-1 features a regular convex front:

$$F_1(\vec{x}) = x_1$$

$$G(x_2,\ldots,x_n) = 1 + 9\sum_{i=2}^{n}\left[x_i/(n-1)\right] \ , \ n = 30 \qquad \text{(eq. 5.2)}$$

$$H\left(F_1, G\right) = 1 - \sqrt{\frac{F_1}{G}}$$

The Pareto front of analytically calculated non-dominated optima (fig. 5.1) can be reproduced for:

$$G(\vec{x}) = 1 \Leftrightarrow \left\{ \begin{array}{l} F_1(\vec{x}) = x_1 \\ \\ F_2(\vec{x}) = G \cdot H\left(F_1, G\right) = H\left(F_1, G\right) \end{array} \right\} \Leftrightarrow$$

$$\Leftrightarrow \left\{ \begin{array}{l} F_1(\vec{x}) = x_1 \\ F_2(\vec{x}) = 1 - \sqrt{\frac{F_1}{G}} = 1 - \sqrt{F_1} = 1 - \sqrt{x_1} \end{array} \right. \qquad \text{(eq. 5.3)}$$

Case ZDT-1 has already been introduced in chapter 4, throughout which it was used as a means of demonstration of the various features of the proposed algorithm (**PA**).

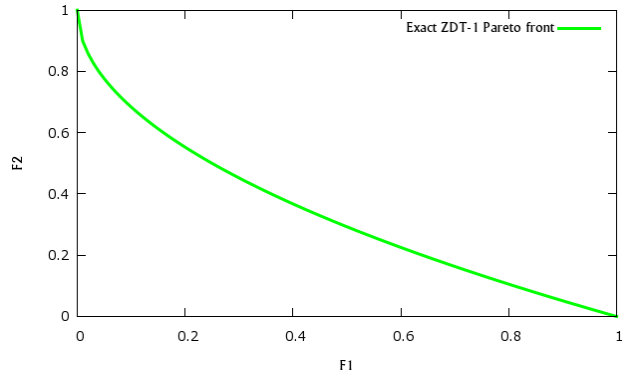


Fig. 5.1 The target front of the ZDT-1 minimization problem ($F_2 = 1 - \sqrt{F_1}$).

5.1.2. Case ‘ZDT-3’

ZDT-3 represents the discreteness element; without any discontinuity whatsoever in problem variable space, its optima form a Pareto front which, though convex, consists of five non-contiguous parts:

$$F_1(\vec{x}) = x_1$$

$$G(x_2,\ldots,x_n) = 1 + 9\sum_{i=2}^{n}\left[x_i/(n-1)\right] \;,\; n = 30 \qquad \text{(eq. 5.4)}$$

$$H(F_1,G) = 1 - \sqrt{\frac{F_1}{G}} - \left(\frac{F_1}{G}\right)\cdot\sin\left(10\cdot\pi\cdot F_1\right)$$

The Pareto front, the discontinuity of which is caused by the introduction of the trigonometric *sinus* function, can, similarly to ZDT-1, be reproduced for:

$$G(\vec{x}) = 1 \Leftrightarrow \left\{\begin{array}{l} F_1(\vec{x}) = x_1 \\ \\ F_2(\vec{x}) = G\cdot H(F_1,G) = H(F_1,G) \end{array}\right\} \Leftrightarrow$$

$$\Leftrightarrow \left\{\begin{array}{l} F_1(\vec{x}) = x_1 \\ F_2(\vec{x}) = 1 - \sqrt{\frac{F_1}{G}} - \left(\frac{F_1}{G}\right)\cdot\sin\left(10\cdot\pi\cdot F_1\right) = 1 - \sqrt{F_1} - F_1\cdot\sin\left(10\cdot\pi\cdot F_1\right) \end{array}\right\} \Leftrightarrow$$

$$\Leftrightarrow \left\{\begin{array}{l} F_1(\vec{x}) = x_1 \\ \\ F_2(\vec{x}) = 1 - \sqrt{x_1} - x_1\cdot\sin\left(10\cdot\pi\cdot x_1\right) \end{array}\right. \qquad \text{(eq. 5.5)}$$

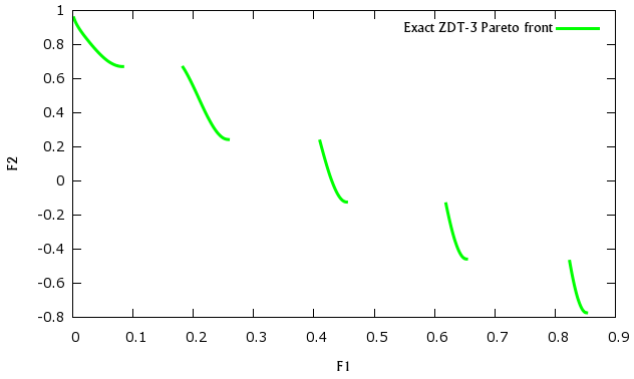


Fig. 5.2 The target front of the ZDT-3 minimization problem. The challenge is to locate solutions on all 5 portions of the non-contiguous Pareto front.

5.2. A turbo-machinery application

The third and final application of the PA presented in this thesis assesses the search for an optimal 2-dimensional stator airfoil design of an axial, cascade, controlled diffusion compressor, at its default operation point (relatively low Reynolds number, transonic flow regime). The airfoil design, belonging to the hub section of the blade, is optimized with two objectives in mind:

i. The minimization of total pressure losses, $\min\ F_1(\vec{x}) \rightarrow \min \omega_{loss}$, or minimization of the *total pressure loss coefficient* ω_{loss} is given by eq. 5.6:

$$\omega_{loss} = \frac{p_{t_{in}} - p_{t_{out}}}{p_{t_{in}} - p_{in}} \qquad \text{(eq. 5.6)}$$

ii. The maximization of static pressure rise, $\max\ F_2(\vec{x}) \rightarrow \max R_p \rightarrow \min\ -R_p$:

$$R_p = \frac{p_{out} - p_{in}}{p_{in}} \qquad \text{(eq. 5.7)}$$

Since the operation point is fixed and all inlet data are given, problem variables exclusively relate to the shape of the airfoil, which is parameterized using two polynomial Bezier curves, one for each side, pressure and suction. Each curve is composed of 160 points, sufficient for the discretization and grid creation needs of the solver. We have opted for 9 control points for each of the two curves and, consequently, 14 ((x_i, y_i) , $i = 2,3,...,8$) variable parameters per side, as the control points at the leading

and trailing edge are fixed, at $(0,0)$ and $(1,0)$ respectively (before stagger angle is considered). The total of 28 design variables is further reduced to 26, as the abscissa for the 2^{st} control point (counting from the leading edge) is also fixed, for both sides $(x_{2,s}=0, x_{2,p}=0)$. Fig. 5.3 better illustrates the airfoil contour generation process:

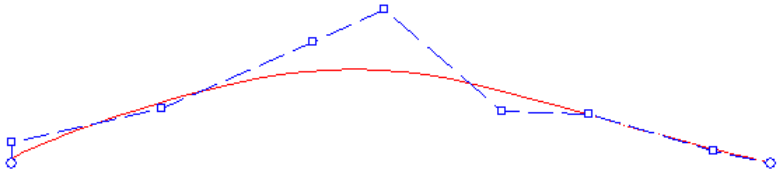


Fig. 5.3 A random suction side of a candidate blade, where the Bezier control points that form the curve are also displayed.

Stagger angle **a**	30°
Pitch	0.65
Flow Inlet angle $\boldsymbol{\alpha_{in}}$	47°
Mach number at inlet $\mathbf{M_{in}}$	0.54
Reynolds number $\mathbf{Re_c}$	$8.41 \cdot 10^5$

Fig. 5.4 Problem specifications.

In order for feasible airfoils to be obtained, as far as constructibility and structural solidity is concerned, we impose certain thickness constraints at different lengthwise points, at 30%, 60% and 90% of chord length, precisely. In this particular run, these constraints took the following form (all lengths un-dimensionalized by the chord's length):

$$Th_{30\%c.l.} \geq C_{\min} = 0.09 \quad , \quad \text{(relaxed threshold : 0.08)}$$

$$Th_{60\%c.l.} \geq C_{\min} = 0.08 \quad , \quad \text{(relaxed threshold : 0.07)}$$

$$Th_{90\%c.l.} \geq C_{\min} = 0.013 \quad , \quad \text{(relaxed threshold : 0.01)} \qquad \text{(eq. 5.8)}$$

We also impose a minimum of flow turning $\hat{a}_1 - \hat{a}_2$, for the resulting airfoil to be an integrable component of the entire compressor, apart from its good individual

aerodynamic properties:

$$\hat{a}_1 - \hat{a}_2 \geq C_{\min} = 20^\circ \quad , \quad (\text{relaxed threshold} : 18^\circ) \qquad (\text{eq. } 5.9)$$

The evaluation software incorporated is the MISES code by M. Drela [37]. Adopting a viscous-inviscid zonal approach to the problem, it is essentially a combination of a two-dimensional integral boundary layer solver with a numerical solver of the Euler equations for the external flow and features a relatively low computational cost whilst producing a very dependable prediction of the flow's characteristics. This advantage enables us to perform as many as 2000-3000 evaluations during a single optimization session, within which range the algorithm is expected to have fully converged.

To further enhance the efficiency of the procedure, the evaluation software performs a thickness check of its own, prior to handing the candidate over to the flow solver. These constraints, with a similar structure as the above (eq. 5.8), and with respective thresholds equal to the relaxed values of eq. 5.8, are intended to immediately reject the absolutely unacceptable geometries before even an evaluation clock unit is expended thereon. Not to mention that, especially if it features negative thickness (an intersection of the suction and pressure sides), a problematic geometry would require additional time, even double as long, for the solver to conclude and proceed to the next candidate.

5.3. Results

Below are demonstrated the results of the various trials the proposed PSO variation underwent along the results yielded by running a certain EA-based optimizer, evolutionary-strategies–based, to be precise. The EA-based program is of adequate depth and features, developed in the Laboratory of Thermal Turbo-machines of the NTUA [indicatively: 38, 41, 42].

To preserve a certain, though qualitative dependability of the results, the duo was run on default settings that are each known to have an overall good performance, independently of the case at hand. No parametric analysis was conducted for the optimizers to conform to each problem’s specifications and iniquities. As far as the EA package is concerned, only those features equivalent in some way to what is implemented in the PA are used: SPEA 2 for non-dominated solution processes, a strategic mutation operator with a dynamically variable mutation probability, Tournament-type selection with elitism.

Besides the non-dominated solution set achieved by every run of either optimizer, we also

introduce a *hyper-volume* figure, which can give a qualitative measure of the convergence rate of the algorithm throughout the duration of a session. For an *n*-objective problem, this hyper-volume metric essentially considers the percentage of a certain, user-defined (see fig. 5.12) portion of *n*-dimensional space that the elite-composed front dominates at any given point during a run. If the hyper-volume grid limits have been introduced appropriately by the user, as the algorithm approaches the Pareto front hyper-volume rating approaches **1.0**. It must be emphasized, at this point, that the hyper-volume curve displayed in each segment is the product of averaging the outcome of 5 different runs, each using different *random number generator seeds* (RNG's) in an attempt to eradicate any impact that the RNG would otherwise have on the ultimate conclusions drawn.

The Pareto fronts displayed were chosen from among the average-performing of each five of runs, according to their respective hyper-volume rating curves.

5.3.1. ZDT-1

Both programs were limited to a maximum of **5000** total evaluations as the only termination criterion. The default set of options was applied to the PA, as in section **4.6**, except for *shuffle*, which was only engaged once, at **70%** of total evaluations. The swarm population was set to **S=50** particles, offspring population **λ** and parents **μ** to **50** and **20** respectively. Therefore, both optimizers completed about **100** iterations, during which the elite archive size could not exceed **40**.

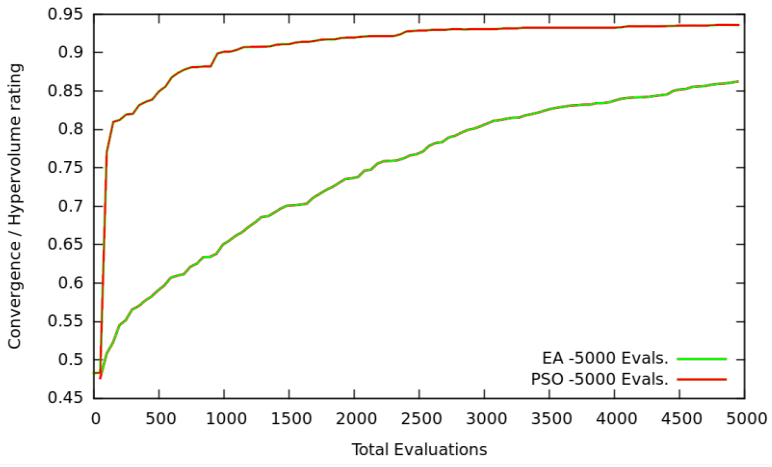


Fig. 5.5 Hyper-volume curve with regard to evaluations completed (ZDT-1 benchmark function), after 5000 total evaluations.

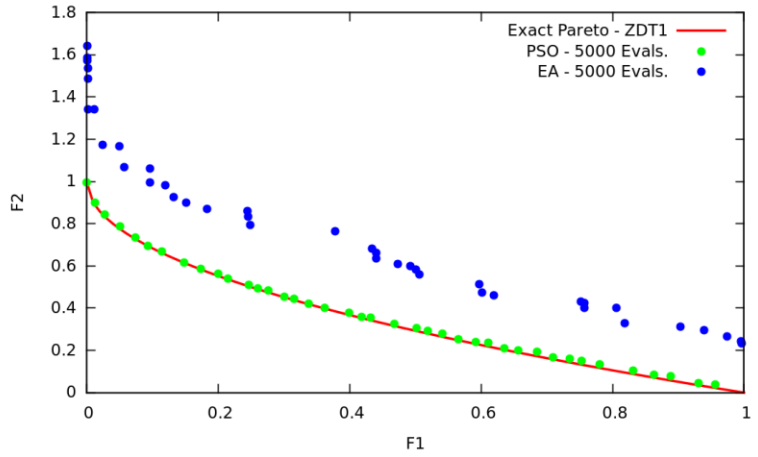


Fig. 5.6 Ultimate elite set achieved by the two optimizers (ZDT-1 benchmark function), after 5000 total evaluations. PSO has fully converged to the analytically calculated Pareto front, having already gone through a shuffle session to improve solution disparity.

5.3.2. ZDT-3

Similarly, both programs were limited to a maximum of **10000** total evaluations as the only termination criterion. The default set of options was applied to the PA, as in section **4.6**, except for *shuffle*, which was, again, only engaged once, at **70%** of total evaluations. The swarm population was set to **S=80** particles, offspring population **λ** and parents **μ** to **80** and **30** respectively. Therefore, both optimizers completed about **125** iterations, during which, the elite archive size could not exceed **80**.

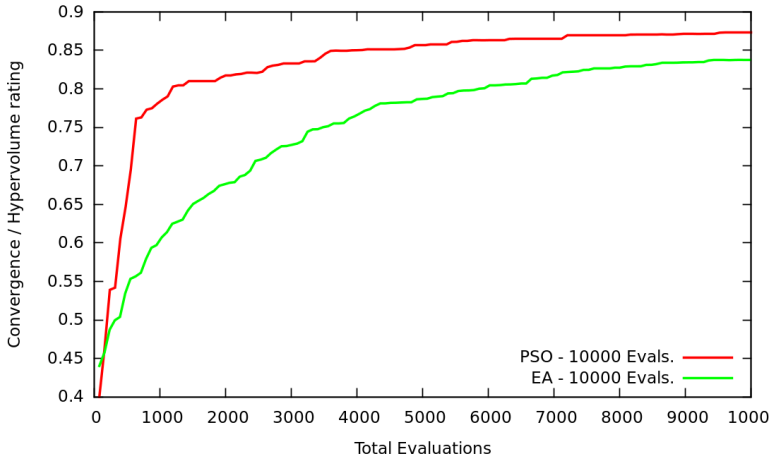


Fig. 5.7 Hyper-volume curve with regard to evaluations completed (ZDT-3 benchmark function), after 10000 total evaluations.

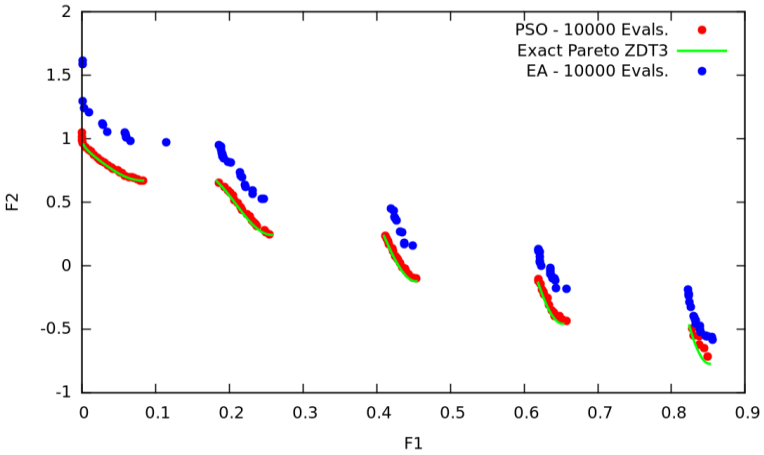


Fig. 5.8 Ultimate elite set achieved by the two optimizers (ZDT-3 benchmark function), after 10000 total evaluations. PSO has converged to the analytically calculated Pareto front, having already gone through a shuffle session to improve solution disparity.

5.3.3. Turbo-machinery application

This time, both programs were limited to a maximum of **1500** total evaluations, due to the high computational demands of this application. Again, the default set of options was applied to the PA, as in section **4.6**, except for *shuffle*, which was, again, only engaged once, at **70%** of total evaluations. The swarm population was set to **S=40** particles, offspring population **λ** and parents **μ** to **40** and **15** respectively. Therefore, both optimizers completed **~38** iterations, during which, the elite archive size could not exceed **20**.

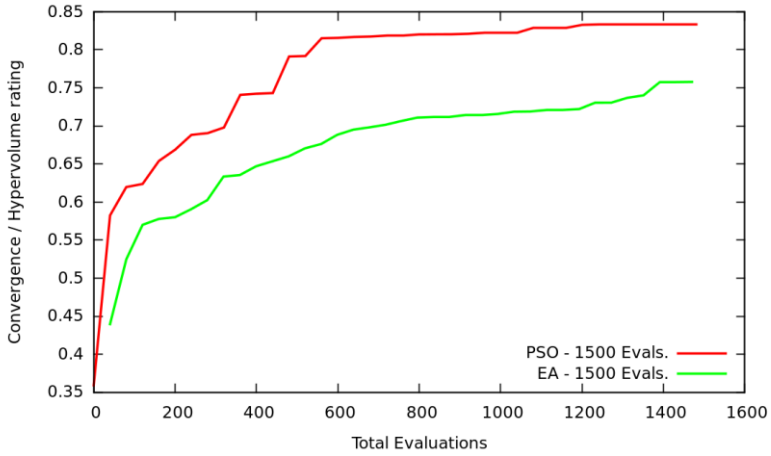


Fig. 5.9 Hyper-volume curve with regard to evaluations completed (Cascade compressor airfoil case), after 1500 total evaluations.

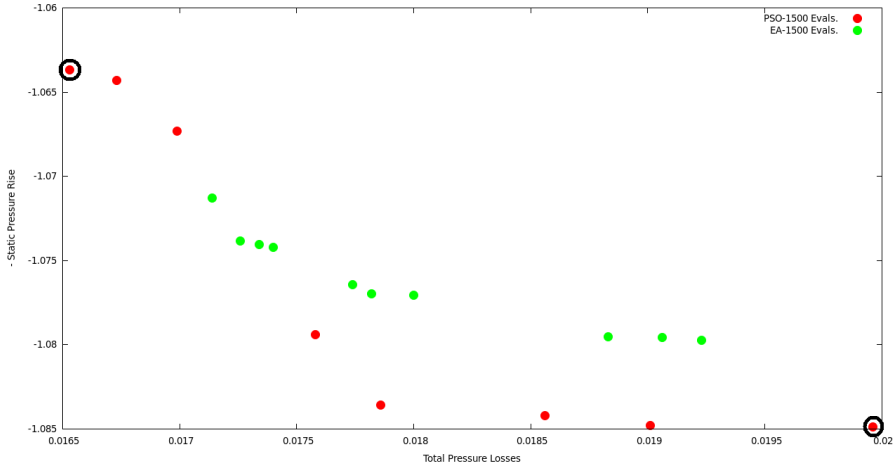


Fig. 5.10 Ultimate elite set achieved by the two optimizers (Cascade compressor airfoil case), after 1500 total evaluations. The 2 circled, extreme optima are the airfoil designs displayed below (fig 5.11). They were selected with optimality in each of the two objectives in mind.

The airfoil case is structured in such a way that it is heavily dependent upon the careful choice of starting variable ranges –which must also be relatively narrow. Else, the algorithm's flow is greatly hindered by frequent constraint violation. For the specified search space, both algorithms seem to have converged before the maximum allowed computational resource is depleted. Below (fig 5.11) are displayed two extreme solutions from the end result set, 'extreme' in the sense that they are picked out from the two opposite edges of the Pareto front, each featuring the best possible performance in one of the two objectives of the problem. Notice the -subtle- differences in geometry which accommodate good individual aerodynamic performance (minimum losses) in one case, maximized flow turning and static pressure rise in the other:

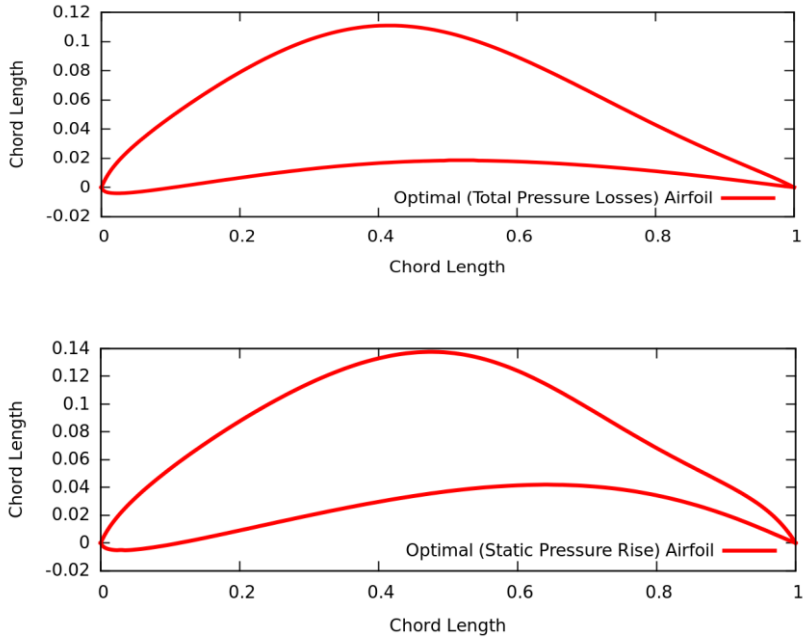


Fig. 5.11 Two sample designs (not in scale) from the non-dominated solution set achieved by the proposed optimizer after 1500 total evaluations. Each represents the optimal (within the framework set by the specified ranges of variables) with regard to each of the two objectives, i) Minimized losses, ii) Maximized static pressure rise.

Problem	*lower bound*	*upper bound*	**Objective function**
ZDT-1	0.0 0.0	1.0 7.0	$\mathbf{F_1}$ $\mathbf{F_2}$
ZDT-3	0.0 -0.8	1.0 6.0	$\mathbf{F_1}$ $\mathbf{F_2}$
Airfoil design	0.016 0.99	0.04 1.1	$\mathbf{F_1}$ **(**$\mathbf{P_t}$ **losses)** $\mathbf{\lvert F_2 \rvert}$ **(**$\mathbf{P_s}$ **rise)**

Fig. 5.12 Summary of the specified grid boundaries for the hyper-volume rating calculations in all 3 test cases. These exact boundaries apply for all runs that participate in the averaging procedure and have been fittingly picked beforehand to encompass the fitness vectors of all candidates generated during a run.

5.4. Discussion of results

Taking a separate at the outcome of each test, starting with ZDT-1, we witness a first indication of the PA's competitive performance: Halfway through the maximum allowed evaluations, it has already almost fully converged to the convex Pareto front, as is displayed by the hyper-volume curve (fig 5.5) that has flattened, in conjunction with the final non-dominated solution set in fig 5.6 which shows the final coincidence of the swarm's elites with the analytically computed optimum curve. The EA, on the other hand, having consumed all 5000 evaluations, has approached but not reached the Pareto front. Its progress rate is very steady throughout the optimization session, which is evidences the relatively straight-forward character of the 1st test benchmark function. The excellent disparity of the non-dominated set achieved by the PA is attributed to the single shuffle session that occurred after 3500 evaluations were completed. Figures 4.41 – 4.22 better illustrate the impact of a shuffle session in this test.

Continuing with ZDT-3, the picture only varies slightly. The PA, again, shows an excellent progress rate, especially during the first 1/5th of evaluations completed, after which it steadily converges to the sought solution fronts. The Pareto is reached before the expendable computational resource is depleted (fig. 5.7), accommodating the use of the shuffle operator to improve diversity and cover the entirety of the non-contiguous Pareto front (fig. 5.8). Shuffle, along with the linearly variable (with iterations) acceleration coefficients, account for the remarkable exploitation efficiency which, in theory, is PSO's greatest shortcoming. Similar experiments conducted by researchers [5, 18, 21], where various representative MO optimizers, either EA-based and/or PSO-based, are put through the same or very similar tests verify the **10000** evaluations threshold as an extremely strict one, validating both PA and EA utilized in this work, performance-wise. Indicatively, in [21], NSGA, SPEA and established PSO-MOO algorithms were allowed up to 25000 evaluations to achieve very similar results. In [5], where Coello et al propose 'MOPSO', they feature similar convex, non-contiguous benchmark functions, allowing 12000 evaluations, within which none of the competitors manages to cover the Pareto front.

Finally, the airfoil design case proves the efficiency of both the EA and the PA in coping with a complex MOO problem, when the –realistic- computational requirements are relatively high. The Drela solver is still considered a computationally cheap alternative to a generally very expensive CFD problem, but its demands in CPU time are much higher than a benchmark function's and closer to industry standards. The added difficulty posed by constraints complete a challenging test. The global optima, which obviously depend upon the specified design variable ranges, cannot be analytically obtained here, so we rely

on experience [39] in this application to specify the 1500 maximum evaluations checkpoint. In validation of this, the PA's hyper-volume rating curve (fig 5.9) has turned flat upon fulfillment of the above termination criterion. Both optimizers achieved solution sets of comparable quality, with the PA managing greater disparity in its non-dominated front, as well as a slightly wider range of ultimate optima (fig. 5.10). Observe that none of the duo managed to reach the elite archive size limit of 20, which is indicative of the complexity of the search process in this particular problem.

The demonstrated test results more than validate the proposed algorithm's functionality and efficiency in coping with varied MOO problems. They also solidify the speculation on distinctions between PSO- and EA- based algorithms, regarding both the end result and general behavior for the duration of the optimization session. The PA does demonstrate the early advantage of swift convergence to the general vicinity of the optima, as all three hyper-volume curves clearly show. Subsequently, its progress rate decreases, although that is in greater part due to the fact that it has actually reached the Pareto front (ZDT-1, ZDT-3). The very good level of diversity among the resulting solutions validates the functionality of the premature convergence counter-measures we introduced, such as the dynamically variant acceleration coefficients and inertia weight and the shuffle operator. Therefore, we can be confident that the speculated exploitation handicap has been appropriately addressed.

It should be stated at this point that none beyond a qualitative conclusion should be drawn from the differences in performance exhibited between the featured EA and the PA. As was outlined in section **5.3**, neither algorithm has been fine-tuned to each problem's specifications and unless the impact of the available settings is thoroughly investigated and the various parameters adjusted to facilitate maximized efficiency, we cannot accept these test results as a final comparative verdict. The featured EA is a proven, highly sophisticated, reliable optimization tool and its testing was intended to serve as a benchmark, successfully so: the PA held its own, even surpassing the EA in all 3 tests conducted, boasting a marginally faster *early* convergence rate and a slightly better end result for a given number of maximum evaluations. This consistency can at the very least establish the PA as an efficient and versatile optimization tool, and a decent foundation to build upon in future work.

6. Discussion - Conclusions - Suggestions for future work

6.1. Concluding summary

The main objective of this work has been the development and subsequent validation of a complete optimization tool based on the concept of the Particle Swarm. Particle Swarm Optimization (PSO), as which, the entirety of optimization-oriented applications of the Particle Swarm is referred to, is a subcategory of the great family of Swarm Intelligence techniques. As such, it introduces processes inspired from the collective activity of a swarm of insects, flock of birds and school of fish or similar to assess the search for optimal solutions to a variety of problems. The members of the swarm, or *particles*, are driven by two main forces: the particle's individual perception of search space, as it is shaped by its own progress thus far (cognitive influence), and its interaction with the rest of the swarm, its awareness of the progress of the swarm as a whole (social influence).

Swarm intelligence itself is a subcategory of the Stochastic Methods, which essentially encapsulate all optimization techniques that rely, to some extent, on randomized search within all specified variable ranges to locate the optima. The most popular and widely applied Stochastic Methods branch is that of Evolutionary Computation and Evolutionary Algorithms (EA's). It is the intention of this work to view all stochastic methods under a unified prism and in **chapter 2** and **3**, after EA's and PSO have been introduced in detail, a long discussion is conducted to highlight the similarities or equivalences between the two, as far as both their philosophical background and their practical application is concerned. The purpose of this is not only to determine the adjacencies between the various components and defining features of these two paradigms, but also to gain insight into possible improvements, either by borrowing principles from each other or by hybridizing. The prominent product of this analysis was that PSO has a relatively faster rate of progress through the earlier stages of a run, while EA's in general shine at a later stage, the phase of *exploitation*, namely the phase where search space has almost been exhausted and the optimizer focuses on refining the located solutions by searching in their immediate vicinity, thus slightly improving the end result.

In **section 3.4**, the various governing parameters of a generic PSO algorithm were discussed: Their impact was analyzed, relevant experiments and literature were surveyed and the various trends were reviewed. The choice of parameters for the proposed algorithm (PA) was elaborately justified, especially from the perspective of addressing the lacking exploitation capabilities of fundamental PSO. A scheme that dynamically alters these parameters was adopted; similar practices and their benefits in EA's were mentioned. No formal parametric analysis was conducted as part of this work, however.

In **chapter 4**, the main product of this thesis was presented: the entire algorithm was reviewed in various sections, each covering a single aspect and all relevant processes. The most effort behind building this program was centered round a few distinct points:

- Moderating PSO's inherent shortcomings: the premature convergence, the problematic behavior when in the general region of optima, which occurs near the conclusion of a session. In this direction, apart from the earlier mentioned adjustments to tuning parameters, the *shuffle* operator was introduced. This measure is intended to intervene and appropriately re-position the swarm and determine certain directions in which to search, directions with evident potential, thus revamping the optimization procedure and re-establishing a more efficient search.

- Emphasis was placed on multi-objective problems, namely problems where the optimality of a solution is judged on multiple criteria. As was explained, the multi-objective regime is completely different to the single-objective one and poses additional challenges, some of which are specific to PSO and pertain to the elevated roles of cognitive and social influence. The reader was introduced to the details of MOO and the current trends in dealing with such problems (the Pareto concept, non-dominated solution sorting methods etc.) in **chapter 2**. In **section 4.3** I specifically elaborated on the approaches adopted in the PA to facilitate a successful transition to MOO: A solution selection/sorting procedure determines the best solutions so far, wherein to invest. A solution *spacing* technique was incorporated to ensure the sought diversity among the various optimal solutions.

- A variety of alternatives in determining a fitting *Global Best* for each particle in a MMO problem was given special mention (**section 4.4**), as a critical step towards successfully extending the PA to MOO. I consider Global Best assignment the trickiest, and at the same time, the most vital factor in building a competitive, MOO-capable particle swarm optimizer. Hence, most novelty of this work and the greatest potential for improvement lie there.

- Other main points, like the Constraint Operator, responsible for administering candidate solutions in breach of any constraints imposed by the problem, and the Initialization phase were given the appropriate attention.

The PA was tested against two benchmark mathematical function cases, especially developed by optimization researchers for exactly this purpose: ZDT-1 and ZDT-3. The

latter, with its challenging non-contiguous Pareto front is a very popular experimental tool. One last test, of a more practical orientation, utilized the PA for the optimization of a cascade compressor's stator airfoil, with regard to individual aerodynamic efficiency and good static pressure rise qualities. This case featured strict constraints and a higher computational cost per examined candidate, thus, a more demanding problem. The PA was tested alongside EA-based optimization software of established competitiveness, serving as a point of reference. The demonstrated results not only granted the PA validation as a fully functional and competent optimizer but also showcased the previously discussed differences in behavior between EA's and PSO. They also highlighted the major contributions to the overall enhancement of PSO's exploitation ability by the addition of the shuffle and variant parameter schemes.

6.2. Future Work – Suggestions

The satisfactory outcome of the above verification justifies considering the PA as firm ground to build upon and improve, both in performance and in functionality/versatility. Improvements can be made through minor additions to the algorithm in its current form or by further developing/optimizing the existing features. Major leaps can also be made as a result of long-term effort, by following the various trends in modern Evolutionary Computation and the field of optimization in general.

Short-term suggestions include:

- ***Further enhancement of the Global Best assignment process.*** As has been emphasized repeatedly, the efficiency of said mechanism is of great importance and one should aim at perfecting it. This challenge goes hand-in-hand with the general problem of MOO, that of solution ranking based on multiple conflicting criteria. Therefore, ideas and new concepts can be borrowed from the advances in that area. A few planned adjustments are described in the respective section, **4.4**. Those are linked to the density scheme and are intended to implicate the pursuit of solution diversity in the selection process. Hybrids of the featured techniques are also possible. *Inexact pre-evaluation** could find ample use here, to give us a rough prediction of the outcome of each option.

- ***Grid computing compatible asynchronous search.*** As the tendency to involve multi-processor environments and grid computing in the procedure of solution evaluation gains momentum, it is ever more crucial for an optimization tool to be able to integrate with maximum efficiency in such a regime. A common

predicament, when a multitude of processing units of differing capacity are handling the evaluation of particles, is for a delay of one or more individuals' evaluations to postpone the transition to the next generation. It is therefore imperative to negate the concept of a 'generation', by making the process of a particle's relocation in search space completely independent from that of the rest of the swarm: the algorithm will no longer address the swarm as a population of solutions that must be updated simultaneously. Each solution, upon completion of its evaluation → 'returns' to the base of operations → immediately receives its new Global Best from within the elite archive, as it has been shaped up to that point → updates its Personal Best accordingly → if the solution is found to be non-dominated, the elite-related processes are spawned → immediately proceeds to update its velocity vector and be assigned its new position in space, according to the PSO core formulae (eqs. 4.1, 4.2) → it enters the evaluation phase once again.

- ***Adjustments to the Shuffle operator.*** This mechanism's positive contribution to the overall performance of the PA has been evidenced in **section 4.5.2**. However, in its current form, it remains a very 'violent' form of intervention and a very demanding one, in terms of swarm experience: the swarm must have already approached the general region of the perceived optima for it to work as intended. An approach where only a small percentage of the swarm is involved in a process launched multiple times and from an earlier stage of the search might be worth looking into. The goal is to gradually transform the shuffle scheme into a principal component of the optimizer, active throughout a run, similarly to mutation in EA's, as opposed to the occasionally intervening, radical measure it is now.

Long term additions worth considering are the following:

- ***A means of highly adaptive, dynamic control of the acceleration coefficients*** C_{cogn}, C_{soc}**.** The somewhat 'raw' linear variant adopted in this work is adequately functional and efficient but cannot match the likes of *strategic mutation* in EA's. The A.C.'s must be made to increase or decrease according to some highly sophisticated metric that will reflect the condition of the swarm, the current rate of progress and the potential for improvement.

- ***A formal, complete parametric analysis.*** Particular attention is due to the acceleration coefficients and inertia weight, or the range thereof, if a variant scheme is finally settled for, linear or otherwise. Second in priority are the various settings of the shuffle operator.

- ***Implementation of inexact pre-evaluation*** of some sort. Methods exist, that allow the approximation of a solution's fitness based on what knowledge we have of the previously exactly evaluated solutions in its vicinity [41, 42, 43]. Inexact pre-evaluation can occasionally substitute the computationally costly evaluation software, with self-explanatory gains. A popular such technique are the *Radial-Basis-Function Networks (RBFN)*. Another, *Kriging*, developed by *G. Krige*, provides additional statistical information: apart from the fitness value approximation, it also returns an estimation of the possible error of this approximation. I have some personal experience in the incorporation of Kriging to EA's for solving single-objective problems. Kriging's proper extension to MOO poses some very intriguing challenges, similar to those of PSO.

Bibliography

[1] Kennedy, J. and Eberhart, R.: Particle Swarm Optimization. IEEE Int. Conf. on Neural Networks (Perth, Australia), IEEE Service Center, Piscataway, NJ. (1995) 1942-1948.

[2] K. Deb, "Multi-objective genetic algorithms: problem difficulties and construction of test problems" Evol. Comput., vol. 7, pp. 205–230, Fall 1999.

[3] K. Deb, S. Agrawal, A. Pratab, and T. Meyarivan, "A fast elitist non- dominated sorting genetic algorithm for multi-objective optimization: NSGA-II," in Proc. Parallel Problem Solving From Nature VI Conf., 2000, pp. 849–858.

[4] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II," IEEE Trans. Evol. Comput., vol. 6, pp. 182–197, Apr. 2002.

[5] C. A. Coello and M. S. Lechuga, "MOPSO: A proposal for multiple objective particle swarm optimization," in Proc. Congr. Evolutionary Computation (CEC'2002), vol. 1, Honolulu, HI, May 2002, pp. 1051–1056.

[6] C.A. Coello, G.T. Pulido, M.S. Lechuga, Handling multiple objectives with particle swarm optimization, IEEE Transactions on Evolutionary Computation 8 (3) (2004) 256–279.

[7] X. Hu and R. Eberhart, "Multiobjective optimization using dynamic neighborhood particle swarm optimization," in Proc. Congr. Evolutionary Computation (CEC'2002), vol. 2, Honolulu, HI, May 2002, pp. 1677–1681.

[8] F. van den Bergh, "An analysis of particle swarm optimization," Ph.D. dissertation, Faculty of Natural and Agricultural Sci., Univ. Petoria, Pretoria, South Africa, Nov. 2002.

[9] Angeline, P. J.: Evolutionary Optimization versus Particle Swarm Optimization: Philosophy and Performance Difference. The 7th Annual Conference on Evolutionary Programming, San Diego, USA. (1998) 601 – 610.

[10] Eberhart, R.C., Shi, Y.: Particle Swarm Optimization: Developments, Applications and Resources. IEEE Int. Conference on Evolutionary Computation. (2001) 81 - 86.

[11] Y. Shi, R. C. Eberhart, "Empirical Study of Particle Swarm Optimization", In Proc. of IEEE Congress on Evolutionary Computation, 1999, pp. 1945 – 1950.

[12] R.C. Eberhart, Y. Shi, Comparing inertia weights and constriction factors in particle swarm optimization, in: Proceedings of the Congress of Evolutionary Computation, San Diego, USA, IEEE Service Center Piscataway, NJ, (2000), pp. 84–89.

[13] Y. Shi, R.C. Eberhart, Parameter selection in particle swarm optimization, in: Evolutionary Programming VII, Lecture Notes in Computer Science, vol. 1447, Springer, 1998, pp. 591–600.

[14] Y. Shi, R.C. Eberhart, A modified particle swarm optimizer, in: Proceedings of the IEEE International Conference on Evolutionary Computation, Anchorage, Alaska, May 4–9, 1998.

[15] J.E. Fieldsend, S.Singh, A multi-objective algorithm based upon particle swarm optimization, an efficient data structure and turbulence, in: Proceedings of UK Workshop on Computational Intelligence (UKCI'02), vol. 2–4, Birmingham, UK, September 2002, pp. 37–44.

[16] C.K. Monson, K.D. Seppi, Adaptive diversity in PSO, in: Proceedings of the 8th Annual Conference on Genetic and evolutionary Computation GECCO'2006, ACM Press, New York, NY, USA, 2006, pp. 59–66.

[17] Carlisle, G. Dozier, Adapting particle swarm optimization to dynamic environments, in: Proceedings of International Conference on Artificial Intelligence (ICAI 2000), Las Vegas, Nevada, USA, 2000, pp. 429–434.

[18] X. Li, A non-dominated sorting particle swarm optimizer for multi-objective optimization, in: Proceedings of the Genetic and Evolutionary Computation Conference (GECCO'2003), Lecture Notes in Computer Science, vol. 2723, Springer, 2003, pp. 37–48.

[19] S.C. Esquivel, C.A.C. Coello, On the use of particle swarm optimization with multimodal functions, in: The 2003 Congress on Evolutionary Computation, 2003. CEC '03., vol. 2, 2003, pp. 1130– 1136.

[20] Evolutionary Algorithm SYstem, NTUA, http://velos0.ltt.mech.ntua.gr/research/easy

[21] P. K. Tripathi, S. Bandyopadhyay, S. K. Pal , Multi-Objective Particle Swarm Optimization with time variant inertia and acceleration coefficients, Machine Intelligence Unit, Indian Statistical Institute , 23 June 2007

[22] K.E. Parsopoulos, M.N. Vrahatis, Particle swarm optimization method in multiobjective problems, in: Proceedings of the 2002 ACM Symposium on Applied Computing (SAC 2002), 2002, pp. 603–607.

[23] J.D. Knowles, D.W. Corne, Approximating the nondominated front using the Pareto archived evolution strategy, Evolutionary Computation 8 (2) (2000) 149–172.

[24] R.C. Eberhart, Y. Shi, Comparison between genetic algorithms and particle swarm optimization, in: Evolutionary Programming VII, Lecture Notes in Computer Science, vol. 1447, Springer, 1998 , pp. 611– 616.

[25] Radha Thangaraj, Millie Pant, Kusum Deep, Initializing PSO with Probability Distributions and Low-discrepancy Sequences: The Comparative Results, 2009 World Congress on Nature & Biologically Inspired Computing (NaBIC 2009)

[26] J. Kennedy, Small worlds and mega-minds: effects of neighborhood topology on particle swarm performance, in: Proceedings of the 1999, Congress of Evolutionary Computation, vol. 3, IEEE Press, 1999, pp.1931–1938.

[27] Ratnaweera, S.K. Halgamuge, Self-organizing hierarchical particle swarm optimizer with time-varying acceleration coefficient, IEEE Trans. Evol. Comput. vol. 8 (June (3)) (2004) 240–255.

[28] Kennedy J., Eberhart, R.C. (2001). Swarm Intelligence. Morgan Kaufmann.

[29] Reynolds, Craig (1987), "Flocks, herds and schools: A distributed behavioral model.", SIGGRAPH '87: Proceedings of the 14th annual conference on Computer graphics and interactive techniques (Association for Computing Machinery): 25–34.

[30] Clerc, M. (1999). The swarm and the queen: towards a deterministic and adaptive particle swarm optimization. Proc. I999 ICEC, Washington, DC, pp 195 1- 1957.

[31] M. Clerc, J. Kennedy, The particle swarm: explosion stability and convergence in a multi-dimensional complex space, IEEE Transactions on Evolutionary Computation 6 (2002) 58–73.

[32] Kennedy, J. (1997), The Particle Swarm: Social Adaptation of Knowledge, IEEE International Conference on Evolutionary Computation (Indianapolis, Indiana), IEEE Service Center, Piscataway, NJ, 303-308.

[33] Lotfi K. Gaafar *, Sherif A. Masoud, Ashraf O. Nassef (2005), A particle swarm-based genetic algorithm for scheduling in an agile environment.

[34] Chia-Feng Juang, A Hybrid of Genetic Algorithm and Particle Swarm Optimization for Recurrent Network Design, IEEE transactions on Systems, Man, and Cybernetics—part b: Cybernetics, vol. 34, no. 2, april 2004

[35] E.Zitzler, M.Laumanns, and L.Thiele: SPEA 2: Improving the strength pareto evolutionary algorithm. Technical Report 103, Computer Engineering and Networks Laboratory (TIK), Swiss Federal Institute of Technology (ETH), 2001.

[36] E. Zitzler, K. Deb, and L. Thiele, “Comparison of multiobjective evolutionary algorithms: Empirical results,” Evol. Comput. vol. 8, no. 2, pp.173–195, Summer 2000.

[37] M. Drela, M.B. Giles. Viscous-inviscid analysis of transonic and low Reynolds number airfoils. AIAA Journal, volume 25(10), pages 1347–1355, 1987.

[38] Ι. Καμπόλης, Πολύ-επίπεδοι, πολυ-επεξεργαστικοί αεροδυναμικής βελτιστοποίησης στις στροβιλομηχανές, Διδακτορική Διατριβή, Εργαστήριο Θερμικών Στροβιλομηχανών, ΕΜΠ, 2009.

[39] Χ. Γεωργοπούλου, Τεχνικές Βελτιστοποίησης για τον προγραμματισμό λειτουργείας αεριο/ατμοστροβιλικών μονάδων και το σχεδιαμό συνιστωσών τους, Διδακτορική Διατριβή, Εργαστήριο Θερμικών Στροβιλομηχανών, ΕΜΠ, 2009.

[40] K.C. Giannakoglou: 'Design of Optimal Aerodynamic Shapes using Stochastic Optimization Methods and Computational Intelligence', International Review Journal Progress in Aerospace Sciences, Vol. 38, pp. 43-76, 2002

[41] M. Karakasis and K.C. Giannakoglou: 'On the Use of Metamodel-Assisted Multi-Objective Evolutionary Algorithms', Engineering Optimisation, Vol. 38(8), pp. 941-957, 2006.

[42] K.C. Giannakoglou: 'Cost-Effective Metamodel-Assisted Evolutionary Algorithms', in 'Evolutionary Algorithms and Intelligent Tools in Engineering Optimization', CIMNE Series of Handbooks, 2005 (ISBN 1-84564-038-1).

[43] K.C. Giannakoglou and I.C. Kampolis: 'Multilevel Optimization Algorithms based on Metamodel- and Fitness Inheritance-Assisted Evolutionary Algorithms', in 'Computational Intelligence in Expensive Optimization Problems', Springer-Verlag Series in Evolutionary Learning and Optimization, 2009

[44] David, M. (1988) Handbook of Applied Advanced Geostatistical Ore Reserve Estimation, Elsevier Scientific Publishing.

[45] Michalewicz, Zbigniew, Fogel, David B., How to Solve It: Modern Heuristics, 2nd ed. Revised and Extended, 2004, ISBN: 978-3-540-22494-5

[46] John Henry Holland. Adaptation in Natural and Artificial Systems: 2nd edition, MIT Press, 1992

[47] Charles C. Darwin, On the Origin of Species by Means of Natural Selection, or the Preservation of Favoured Races in the Struggle for Life, 1859