ΠΑΝΕΠΙΣΤΗΜΙΟ ΜΑΚΕΔΟΝΙΑΣ
ΤΜΗΜΑ ΕΦΑΡΜΟΣΜΕΝΗΣ ΠΛΗΡΟΦΟΡΙΚΗΣ

ΕΡΓΑΣΤΗΡΙΟ ΠΑΡΑΛΛΗΛΗΣ & ΚΑΤΑΝΕΜΗΜΕΝΗΣ ΕΠΕΞΕΡΓΑΣΙΑΣ

ΠΑΡΑΛΛΗΛΟΙ ΜΙΜΗΤΙΚΟΙ ΑΛΓΟΡΙΘΜΟΙ

Παράλληλοι εξελικτικοί αλγόριθμοι και άλλες τεχνικές

ΔΙΔΑΚΤΟΡΙΚΗ ΔΙΑΤΡΙΒΗ

του

Ιάσονος Γ. Διγαλάκη

Θεσσαλονίκη, Οκτώβριος 2005

ΠΑΝΕΠΙΣΤΗΜΙΟ ΜΑΚΕΔΟΝΙΑΣ
ΤΜΗΜΑ ΕΦΑΡΜΟΣΜΕΝΗΣ ΠΛΗΡΟΦΟΡΙΚΗΣ
ΕΡΓΑΣΤΗΡΙΟ ΠΑΡΑΛΛΗΛΗΣ & ΚΑΤΑΝΕΜ. ΕΠΕΞΕΡΓΑΣΙΑΣ

ΠΑΡΑΛΛΗΛΟΙ ΜΙΜΗΤΙΚΟΙ ΑΛΓΟΡΙΘΜΟΙ

Παράλληλοι εξελικτικοί αλγόριθμοι και άλλες τεχνικές

ΔΙΔΑΚΤΟΡΙΚΗ ΔΙΑΤΡΙΒΗ

του

Ιάσονος Γ. Διγαλάκη

Συμβουλευτική Επιτροπή: Κ. Μαργαρίτης, Καθηγητής (επιβλέπων)
Ι. Βλαχάβας, Καθηγητής
Ε. Ρουμελιώτης, Επ. Καθηγητής

Εγκρίθηκε από την επταμελή εξεταστική επιτροπή την 26η Σεπτεμβρίου 2005.

.....	.....	.....
Κών/νος Μαργαρίτης	Ιωάννης Βλαχάβας	Εμμανουήλ Ρουμελιώτης
Καθηγητής	Καθηγητής	Επ. Καθηγητής
.....	.....	.....
Αναστάσιος Κάτος	Κών/νος Παπαρρίζος	Δημήτριος Παπαναστασίου
Καθηγητής	Καθηγητής	Αν. Καθηγητής
	.....	
	Μαρία Σατρατζέμη	
	Αν. Καθηγήτρια	

Θεσσαλονίκη, (Οκτώβριος 2005).

.....
(Ιάσων Γ. Διγαλάκης)
Υπ. Διδάκτωρ Εφ. Πληροφορικής Πανεπιστημίου Μακεδονίας.

Ἐν ἀρχῇ ἦν ὁ λόγος,
καὶ ὁ λόγος ἦν πρὸς τὸν θεόν,
καὶ θεὸς ἦν ὁ λόγος.
οὗτος ἦν ἐν ἀρχῇ
πρὸς τὸν θεόν.
πάντα δι᾽ αὐτοῦ ἐγένετο,
καὶ χωρὶς αὐτοῦ ἐγένετο οὐδὲ ἕν,
ὃ γέγονεν ἐν αὐτῷ ζωὴ ἦν,
καὶ ἡ ζωὴ ἦν τὸ φῶς τῶν ἀνθρώπων
καὶ τὸ φῶς ἐν τῇ σκοτίᾳ φαίνει,
καὶ ἡ σκοτία αὐτὸ οὐ κατέλαβεν[1].

[1] (Κατά Ιωάννη 1.1-1.5)

Ευχαριστίες

Θέλω να ευχαριστήσω ιδιαίτερα τον επιβλέποντα καθηγητή μου κ. Μαργαρίτη Κωνσταντίνο για την πολύτιμη συνεισφορά του στην εκπόνηση της διατριβής. Η αρχική θεωρητική κατάρτιση που μου έδωσε, όπως και οι συνεχείς παρατηρήσεις του, απεδείχθηκαν απαραιτητοι σύμβουλοι για την επιστημονική ορθότητα της εργασίας. Ιδίως δε για την αμέριστη αρωγή του, μέσω προτροπών και επισημάνσεων στο θέμα της υλοποίησης των προβλημάτων μας, οι οποίες καθόρισαν ουσιαστικά την πορεία περάτωσης όσο και την συντακτική μορφή του τελικού κειμένου.

Στο σημείο αυτό να ευχαριστήσω ιδιαίτερα τα μέλη της συμβουλευτικής επιτροπής που στήριξαν στα χρόνια που πέρασαν αυτή την προσπάθεια.

Ολοκληρώνοντας να τονίσω ότι σημαντική ήταν και η βοήθεια που μου προσέφερε ο ερευνητής του πανεπιστημίου του Newcastle της Αυστραλίας Δρ. Pablo Alberto Moscato παρέχοντας μου πλούσιο βιβλιογραφικό υλικό και δίνοντας μου χρήσιμες πληροφορίες, κυρίως πάνω σε θέματα μιμητικών αλγορίθμων.

Περιεχόμενα

Κατάλογος Πινάκων

Κατάλογος Σχημάτων

Περίληψη

Τις τρείς τελευταίες δεκαετίες υπάρχει μια έντονη δραστηριότητα στων χώρο των αλγορίθμων που βασίζονται στη μοντελοποίηση διαφόρων φυσικών, βιολογικών ή κοινωνικών φαινομένων με σκοπό τη δημιουργία ευρετικών αλγορίθμων δια την εύρεση του ακροτάτου συνεχών συναρτήσεων ή προβλημάτων αλλά και για την επίλυση δύσκολων προβλημάτων συνδυαστικής βελτιστοποίησης.

Από τους πιο πρόσφατους αλγορίθμους που αναπτύχθηκαν στα πλαίσια της παραπάνω έρευνας είναι οι Μιμητικοί Αλγόριθμοι οι οποίοι είναι σύνθεση διαφόρων τύπων ευρετικών αλγορίθμων που περιέχουν και στοιχεία μίμησης βιολογικών και κοινωνικών διαδικασιών. Ο χρόνος εκτέλεσης τους είναι εν γένει μεγαλύτερος από αυτόν κάποιου αντίστοιχου ευρετικού αλλά σε αντιστάθμισμα αναμένουμε λύσεις καλύτερης ποιότητας από αυτές που παράγει ο αντίστοιχος ευρετικός ή εξελικτικός.

Τα αποτελέσματα από τη χρήση των μιμητικών αλγορίθμων σε δύσκολα προβλήματα βελτιστοποίησης είναι ενθαρρυντικά και συγκρίσιμα με μεθόδους που έχουν ερευνηθεί για πολύ μεγαλύτερο χρονικό διάστημα όπως των Εξελικτικών Αλγορίθμων, της Προσομοιούμενης Ανόπτησης και της Αποτρεπτικής Αναζήτησης.

Η διερεύνηση μεγάλων χώρων αναζήτησης με πολύπλοκες συναρτήσεις ποιότητας και η χρήση

πολυάριθμων μεταβλητών επιβάλλουν την ανάγκη γρήγορων υπολογιστικά ισχυρών υλοποιήσεων. Η διαθεσιμότητα γρήγορων και φθηνών παράλληλων Η/Υ, κάνει δυνατή την εφαρμογή μιμητικών και άλλων μεταευρετικών μεθόδων σε δύσκολα και σύνθετα προβλήματα στα οποία γενικώς απαιτείται η χρήση μεγάλων πληθυσμών.

Στα πλαίσια της παρούσης διδακτορικής διατριβής αρχικά γίνεται μια εισαγωγή στις βασικές έννοιες της θεωρίας βελτιστοποίησης, των εξελικτικών αλγορίθμων και άλλων συναφών τεχνικών. Κατόπιν δίνονται οι βασικές έννοιες του παράλληλου προγραμματισμού και της βιβλιοθήκης MPI.

Ακολούθως παρουσιάζεται η βασική διάταξη μιμητικών αλγορίθμων και μερικά παραδείγματα χρήσης. Οι αλγόριθμοι αυτοί ενσωματώνουν με αποτελεσματικό τρόπο τις αρχές των εξελικτικών τεχνικών εφαρμόζοντας άλλες ευρετικές στρατηγικές με στόχο την αποφυγή εγκλωβισμού σε τοπικά ακρότατα.

Στη συνέχεια γίνεται εκτενής μελέτη της απόδοσης μιμητικών αλγορίθμων καθώς και άλλων εξελικτικών και μεταευρετικών τεχνικών, προτείνεται ένα σύνολο δεκατεσσάρων προβλημάτων για την αξιολόγηση μεταευρετικών αλγορίθμων και μετά παρουσιάζεται μια παράλληλη υλοποίηση τριών αντιπροσωπευτικών Μιμητικών Αλγορίθμων PMA1, PMA2, PMA3 στην πειραματική συστοιχία του Πανεπιστημίου Μακεδονίας. Προτείνεται δε σαν αποδοτικότερη και αποτελεσματικότερη μέθοδος μια υβριδική των τριών παραλλήλων υλοποιήσεων PMA4.

Το πρόβλημα του περιοδεύοντος πωλητή TSP χρησιμοποιήθηκε για να επιβεβαιωθεί η υπεροχή της PMA4 έναντι των υπολοίπων, αλλά και για να δοκιμαστεί η απόδοση της PMA4 σε συνδυαστικά προβλήματα. Για την αξιολόγηση των μεθοδολογιών μας, χρησιμοποιήθηκαν και πραγματικά προβλήματα. Τα προβλήματα ελήφθησαν από τη βιβλιογραφία και αναφέρονται σε

εφαρμογές από το πεδίο του χρονοπρογραμματισμού συστημάτων παραγωγής και του αυτοματοποιημένου ωρολογίου προγράμματος. Από την ανάλυση που πραγματοποιήθηκε προέκυψε ότι την καλύτερη επίδοση, τόσο από πλευράς ακρίβειας εντοπισμού του ολικού βέλτιστου όσο και από πλευράς ταχύτητας, παρουσιάσε η χρήση στις διάφορες νησίδες επεξεργαστών ετερογενών μιμητικών αλγορίθμων.

Κεφάλαιο 0

Εισαγωγή

0.1 Αντικείμενο της εργασίας

Αντικείμενο αυτής της εργασίας είναι η μελέτη της απόδοσης εξελικτικών και άλλων τεχνικών κατά την διερεύνηση μεγάλων χώρων αναζήτησης με πολύπλοκες συναρτήσεις ποιότητας και χρήση πολυάριθμων πληθυσμών, εστιάζοντας στην μελέτη υβριδικών αλγορίθμων με πιο πολύπλοκη δομή από ένα εξελικτικό ή ένα απλό ευρετικό αλγόριθμο. Ο χρόνος εκτέλεσης τους είναι εν γενει μεγαλύτερος από αυτόν κάποιου αντίστοιχου εξελικτικού και πολλές φορές απαγορευτικός σε ένα συμβατικό υπολογιστή. Σε αντιστάθμισμα αναμένουμε λύσεις καλύτερης ποιότητας από αυτές που παράγει ένας απλός ευρετικός ή ένας απλός εξελικτικός αλγόριθμος. Η εργασία αυτή μπορεί να χωριστεί σε τρία σκέλη. Το πρώτος σκέλος περιλαμβάνει μια μελέτη της απόδοσης των μιμητικών καθώς και αλλων εξελικτικών και ευρετικών αλγορίθμων, η ανάπτυξη των οποίων είναι ραγδαία τα τελευταία χρόνια και συμβαδίζει με την εντυπωσιακή

βελτίωση των δυνατοτήτων των ηλεκτρονικών υπολογιστών.

Το δεύτερο σκέλος αναφέρεται στην παράλληλη υλοποίηση σε μια παράλληλη αρχιτεκτονική γενικού σκοπού, όπως η συστοιχία από σταθμούς εργασίας. Στα πλαίσια αυτής της προσπάθειας οι αλγόριθμοι αυτοί να γίνουν πιο γρήγοροι, αποτελεσματικοί και ευέλικτοι εντάσσεται και η υλοποίηση ετερογενών και ομογενών αλγορίθμων στις διάφορες νησίδες επεξεργαστών της πειραματικής συστοιχίας του Πανεπιστημίου Μακεδονίας. Οι υλοποιήσεις των αλγορίθμων σε αυτήν την αρχιτεκτονική γίνεται με την βοήθεια της βιβλιοθήκης παράλληλου προγραμματισμού MPI (Message Passing Interface). Η παράλληλη υλοποίηση που αναπτυχθηκε στα πλαίσια αυτης της διατριβής συνδυάζει με αποτελεσματικό τρόπο τις υφιστάμενες μεθοδολογίες σε ένα πρωτότυπο μοντέλο.

Το τρίτο και τελευταίο σκέλος της εργασίας αυτής έχει στόχο την εμπειρική αξιολόγηση των κυριότερων μεθόδων που διερευνήθηκαν, βάσει πραγματικών προβλημάτων. Η έρευνα που έγινε κατάδειξε οτι οι μεθοδολογίες που αναπτύχθηκαν μπορεί να θεωρηθούν πολύ καλύτερες μέθοδοι των καθιερωμένων μεθόδων βελτιστοποίησης, καθώς αντιμετωπίζουν με μεγαλύτερη επιτυχία το σύνολο των προβλημάτων που εξετάστηκαν. Στην παράλληλη εκδοχή παρουσιάζουμε πολύ καλύτερους χρόνους από τις αντίστοιχες παράλληλες εκδοχές άλλων μεθόδων.

0.2 Συμβολή της διατριβής

Η συστηματική μελέτη της απόδοσης των μιμητικών αλγορίθμων ενός γνωστικού πεδίου με μικρό χρόνο ζωής, μεγάλο εύρος εφαρμογής και ταχέως εξελισσόμενου αποτελεί μια πρώτη συμβολή της παρούσας εργασίας. Συγκεκριμένα:

α) καλύφθηκε όλο το φάσμα της σχετικής με το αντικείμενο των Μιμητικών Αλγορίθμων βιβλιογραφίας

β) συστηματοποιήθηκαν και αναλύθηκαν τα χαρακτηριστικά των Μιμητικών Αλγορίθμων.

Η παράλληλη υλοποίηση ενός πρωτότυπου σχήματος, δηλαδή των Μιμητικών Αλγορίθμων συνιστά τη δεύτερη και βασικότερη συμβολή της εργασίας. Είναι αλήθεια ότι η ιδέα για την ανάπτυξη του υβριδικού μοντέλου ετερογενών μιμητικών αλγορίθμων σε ετερογενείς σταθμούς εργασίας ήρθε μαλλον τυχαία, κατά τη διάρκεια πειραματισμών με διάφορα σχήματα προσομοιούμενης ανόπτησης, εξελικτικών αλγορίθμων και αποτρεπτικής αναζητησης με χρήση της βιβλιοθήκης διεπαφής περάσματος μηνυμάτων (MPI) και της βιβλιοθήκης PARAMENOAS. Βέβαια για να φτάσει το μοντέλο μας σε ένα επιθυμητό επίπεδο ποιότητας, χρειάστηκε να αφιερωθεί αρκετός υπολογιστικός χρόνος, ωστόσο τα ιδιαίτερα ενθαρρυντικά συμπεράσματα καταδεικνύουν ότι το πείραμα πέτυχε.

Η μεθοδολογία αξιολόγησης των διαφόρων αλγορίθμων αποτελεί την τρίτη συμβολή της διατριβής. Επιλέχθηκαν αρχικά δεκατέσσερα θεωρητικά προβλήματα βελτιστοποίησης με βάση το πλήθος των μεταβλητών ελέγχου, το πλήθος των ακροτάτων, τη γνώση της θέσης του ολικού ακροτάτου, τη γεωμετρία της επιφάνειας απόκρισης και την ύπαρξη θορύβου ή ασυνεχειών στην αντικειμενική συνάρτηση. Τα πραγματικά προβλήματα που επιλυσαμε σε αντίθεση με τα θεωρητικά μπορούν να χαρακτηριστούν αντιπροσωπευτικά προβλήματα βελτιστοποίησης με περιορισμούς. Τα προβλήματα αυτά αφορούσαν τον χρονοπρογραμματισμό παραγωγής από μονάδες ηλεκτρικού ρεύματος και το εξαμηνιαίο πανεπιστημιακό ωρολόγιο πρόγραμμα.

Τέλος στα πλαίσια της εργασίας αυτής καταβλήθηκε προσπάθεια απόδοσης πολλών ξένων όρων, δεδομένου μάλιστα ότι η ελληνική βιβλιογραφία πάνω στο αντικείμενο είναι περιορισμένη . Οι

όροι αυτοί απαντώνται στο κείμενο εντος παρενθέσεως δια διευκόλυνση του αναγνώστη και εχουν καταχωρηθεί στο ευρετήριο της διατριβής.

0.3 Διάρθρωση της εργασίας

Η εργασία αυτή περιλαμβάνει, εκτός από την παρούσα εισαγωγή (Κεφάλαιο 0), επτά Κεφάλαια, τα συμπεράσματα - προτάσεις (Κεφάλαιο 8), δύο παραρτήματα, την βιβλιογραφία και ένα ευρετήριο ελληνοαγγλικών όρων.

Στο Κεφάλαιο 1 παρουσιάζεται μια εισαγωγή στις βασικές έννοιες της θεωρίας βελτιστοποίησης, των εξελικτικών αλγορίθμων και άλλων συναφών υβριδικών τεχνικών.

Στο Κεφάλαιο 2 σκοπός είναι να φερει τον αναγνώστη σε επαφή με τις βασικές έννοιες του παράλληλου προγραμματισμού και της βιβλιοθήκης διεπαφής περάσματος μηνυμάτων.

Στο Κεφάλαιο 3 αναφέρονται τα κύρια στοιχεία που χαρακτηρίζουν τους μιμητικούς αλγορίθμους και παρουσιάζεται η βασική διάταξη ενός MA.

Στο Κεφάλαιο 4 επιχειρείται συγκριτική αξιολογηση πέντε αντιπροσωπευτικών αλγορίθμων σε προβλήματα αναλυτικών αντικειμενικών συναρτήσεων και προτείνεται ενα σύνολο δεκατεσσάρων θεωρητικών μαθηματικών προβλημάτων για μελέτη της απόδοσης των εξελικτικων αλγορίθμων.

Στο Κεφάλαιο 5 παρουσιάζονται τέσσερις υλοποιήσεις μιμητικών αλγορίθμων σε μια συστοιχία ετερογενών σταθμών εργασιών με σκοπό να κερδίσουμε χρόνο σε χαμηλό κόστος και παρουσιάζονται τα αποτελέσματα που ειχαμε για τα προβλήματα που περιγράφονται στο κεφάλαιο 4.

Στο Κεφάλαιο 6 παρουσιάζεται ένα αιτιοκρατικό πρόβλημα χρονοπρογραμματισμου αυτό του χρονοπρογραμματισμού παραγωγής σταθμών ηλεκτρικού ρεύματος και τα αποτελέσματα που

είχαμε απο τις προτεινόμενες μεθοδολογίες του κεφαλαίου 5.

Στο Κεφάλαιο 7 παρουσιάζεται το πρόβλημα του πανεπιστημιακού ωρολογιου προγράμματος και τα αποτελέσματα που είχαμε από τις προτεινόμενες μεθοδολογίες του κεφαλαίου 5.

Στο Παράρτημα Α έχουμε ένα εγχειρίδιο εγκατάστασης και χρήσης της βιβλιοθήκης PARAMENOAS. Το Παράρτημα Β περιέχει τους κώδικες για τα προβλήματα του κεφαλαίου 6 και 7 καθώς και τον κώδικα για το πρόβλημα του περιοδεύοντος πωλητή TSP που χρησιμοποιείται στο κεφάλαιο 5.

0.4 Δημοσιεύσεις

Οι δημοσιεύσεις που έγιναν στα πλαίσια αυτής της διατριβής ήσαν οι παρακάτω:

Αρθρα σε διεθνή περιοδικά

- Digalakis Jason, Margaritis Konstantinos, Parallel Evolutionary Algorithms on Message Passing Cluster, Parallel Processing Letters, accepted for publication 2005-2006.

- Digalakis Jason, Margaritis Konstantinos, Performance Comparison of Memetic Algorithms, Journal of Applied Mathematics and Computation, Elsevier Science Volume 158, 25 October 2004, Pages 237-252.

- Digalakis Jason, Margaritis Kostantinos, A parallel memetic algorithm for optimization problem, vol. I, pages 121-130, MIC Book, Kluwer. 2003.

- Digalakis Jason, Margaritis Konstantinos, An Experimental study of Benchmarking

Functions for Evolutionary Algorithms, International Journal of Computer Mathemathics, Vol. 79, pages 403-416, April 2002.

- Digalakis Jason, Margaritis Konstantinos, A parallel cultural algorithm for the electrical generator scheduling problem, IMACS Journal : "Mathematics and Computers in Simulation" Elsevier Sience , Vol 60, Issues 3-5, 30 September 2002, Pages 293-301

- Digalakis Jason, Margaritis Konstantinos, Benchmarking Functions for Genetic Algorithms, International Journal of Computer Mathemathics, Vol. 77, Number 4, pages 481-506, 2001.

Διεθνή συνέδρια

- Digalakis Jason, Margaritis Konstantinos, Parallel Evolutionary Algorithms on Message Passing Clusters, Parallel Computing Conference PARCO, Germany, 2003.

- Digalakis Jason, A parallel memetic Library for non-linear Optimization Problems,3th meeting of PAREO Euro working group on Parallel Processing in Operations Research, France, 2002.

- Digalakis Jason, Adriazola Valenzuela Cecilia Alejandra, Margaritis Konstantinos, A parallel memetic environment for non-linear and other unconstrained optimisation problems, XI Congreso Latino-Iberoamericano de Investigation de Operaciones, Vol 1, pages 60-65, Concepcion-Chile, 2002.

- Digalakis Jason, Margaritis Konstantinos, A parallel memetic algorithm for solving

optimization problems, In Proceedings of 4th Meta-heuristics International Conference,vol. I, pages 121-125, July 16-20, Portugal, 2001.

- Digalakis Jason, Margaritis Konstantinos, A Parallel Memetic Algorithm for the maintenance scheduling problem, In Proceedings of 5th World Multiconference on Systemics, Cybernetics and Informatics (SCI 2001) and the 7th International Conference on Information Systems Analysis and Synthesis (ISAS 2001),vol VII, pages 343-347, Orlando-Florida, USA, 2001.

- Digalakis Jason, Margaritis Konstantinos, A Parallel Hybrid Evolutionary Algorithm for electrical generator scheduling problem, In Proceedings of 5th World Multiconference on Systemics, Cybernetics and Informatics (SCI 2001) and the 7th International Conference on Information Systems Analysis and Synthesis (ISAS 2001), vol. XVI, pages 397-402, Orlando-Florida, USA, 2001.

- Digalakis Jason, Margaritis Konstantinos, A parallel environment for Optimization Problems, First Cracow Grid Workshop, November 5-7, Cracow, Poland, 2001.

- Digalakis Jason, Margaritis Konstantinos, An Experimental study of benchmarking functions for Genetic Algorithms, In Proceedings of IEEE Conference on Transactions, Systems, Man and Cybernetics, vol. V, pages 3810-3815, Nashville-Tennessee, USA, 2000.

Εθνικά συνέδρια

- Digalakis Jason, Margaritis Konstantinos, A parallel memetic library for optimization

problems, 4th GRACM Congress on Computational Mechanincs , Patra, 2002.

- Digalakis Jason, Margaritis Konstantinos, A multipopulation memetic model for the maintenance scheduling problem, 5th Hellenic European Conference on Computer Mathematics and its Applications(HERCMA 2001), pages 241-242, Athens, 2001.

- Digalakis Jason, Margaritis Konstantinos, A Multipopulation Cultural Algorithm for the Electrical Generator Scheduling problem, In Proceeding of Panhellenic Symposium on Automation, Robotics and Industrial Production. pages 85-90,June 28-30, Santorini, 2001.

- Digalakis Jason, Margaritis Konstantinos, An Experimental Study of Genetic Algorithms using PGAPack,In Proceedings of 7th Hellenic Conference on Informatics, pages v34-v40, University of Ioannina, 1999.

0.5 Επιλεγμένες βιβλιογραφικές αναφορές άλλων ερευνητών

Ορισμένες εργασίες άλλων ερευνητών στις οποίες αναφέρουν τμήματα της εργασίας αυτής είναι οι ακόλουθες:

- Jonathan Gomez, Self Adaptation of Operator Rates in Evolutionary Algorithms, K. Deb et al. (Eds.): GECCO 2004, LNCS 3102, pages 1162-1173, Springer-Verlag, Berlin Heidelberg 2004.

- Krishnaiyer Krishnan and Hossein Cherarghi, Minimizing Total Completion on a Single Machine with Tool changes: An Ant Approach, Proceedings of the Industrial Engineering Research Conference (IERC), May 2004.

- R. Weber, Knowledge Management for Computational Intelligence Systems,College of Information Science Technology Drexel University, Eight IEEE International Symposium on High Assurance Systems Engineering, 25-26 March 2004.

- Gunduz-Oguducu S., Etaner-Uyar Sima, A Graph Based Clustering Method using a Hybrid Evolutionary Algorithm, WSEAS Transactions on Mathematics, Vol. 3, Issue 3, pages 731-736, WSEAS, 2004.

- H. Mayer, M. Spitzlinger, Multi-Chromosomal Representations and Chromosome Shuffling in Evolutionary Algorithms, 2003 Congress on Evolutionary Computation, Australia, 2003.

- Salem Andra,Optimisation Techniques for GAs Turbine Engine Control System, MS Thesis in Advanced Software Engineering, University of Sheffield, Department of Computer Science, August 2003,UK.

- Stephen Shervais and Martin Zwick, Ordering Genetic Algorithm Genomes with reconstrability analysis, International Journal of General Systems, Taylor Francis, Volume 32, Number 5, pages 491 - 502, September 2003.

- Krishnaiyer, Krishnan, S. Hossein Cheraghi, Ant Algorithms: Review and Future Applications, Proceedings of the Industrial Engineering Research Conference (IERC),

May, 2002.

- Luiz Antonio Nogueira, Alexandre C. M. de Oliveira, Real-Coded Evolutionary Approaches to Unconstrained Numerical Optimization, 3rd Congress of Logic Applied to Technology LAPTEC 2002, Book Series Frontiers in Artificial Intelligence and its Applications, Editors J.M. Abe and J.I. da Silva Filho , IOS Press, 2002.

- Alexandre C. M. de Oliveira, Treinamento Populacional em Heuristicas: Aplicacoes em Otimizacao, (PhD Thesis in Computer Science) Proposta de Tese apresentada como parte dos requisitos para a obtencao do titulo de Doutor em Computacao Aplicada no Instituto Nacional de Pesquisas Espaciais Orientador: Prof. Dr. Luiz Antonio de Nogueira Lorena Sao Jose dos Campos/SP Maio/2002.

- Jonatan Gomez, Dipankar Dasgupta, Fabio Gonzalez, Using Adaptive Operators in Genetic Search, In Proceedings of the Genetic and Evolutionary Computation Conference (GECCO), Lecture Notes in Computer Science, Vol. 2724, pages 1577-1578, Springer-Verlag, July 2003.

- Sergio Luciano Avila, Algoritmos Geneticos Aplicados na Otimizacao de Antenas Refletoras, (MS Thesis in Electrical Engineering) Dissertacao submetida a Universidade Federal de Santa Catarina como parte dos requisitos para a obtencao do grau de Mestre em Engenharia Eletrica Florianopolis, Novembro de 2002.

Κεφάλαιο 1

Εξελικτικές και άλλες Ευρετικές Προσεγγίσεις

Αυτό το κεφάλαιο παρουσιάζει μια εισαγωγή στις βασικές έννοιες της βελτιστοποίησης, των εξελικτικών αλγορίθμων και άλλων συναφών τεχνικών. Αρχικά παρουσιάζονται εξελικτικές προσεγγίσεις βελτιστοποίησης, στην συνέχεια άλλες ευρετικές μέθοδοι και στο τέλος υβριδικές προσεγγίσεις βελτιστοποίησης.

1.1 Προβλήματα Βελτιστοποίησης

1.1.1 Ορισμοί

Έστω το μέτρο επίδοσης ενός φυσικού ή μαθηματικού συστήματος:

$$P = f(x_1, x_2, ..., x_n)$$

όπου $f(x_1, x_2, ..., x_n)$ μια πραγματική συνάρτηση ορισμένη στο $D \subseteq R^n$ και $x = [x_1, x_2, ..., x_n]^T$ διάνυσμα παραμέτρων. Η f καλείται αντικειμενική συνάρτηση (objective function), ενώ οι παράμετροι x_i ονομάζονται μεταβλητές ελέγχου (control variables) ή μεταβλητές απόφασης (decision variables) ή απλά παράμετροι του συστήματος. Η γεωμετρική απεικόνιση της αντικειμενικής συνάρτησης f ονομάζεται επιφάνεια απόκρισης (response surface), ενώ το πεδίο ορισμού της D καλείται εφικτή περιοχή (feasible region) ή εφικτός χώρος (feasible space) ή χώρος πολιτικής (policy domain) . Στην γενική περίπτωση ο χώρος D ορίζεται από ένα σύνολο m μαθηματικών σχέσεων της μορφής:

$$g_i(x_1, x_2, ..., x_n) \leqslant, =, \geqslant 0$$

Ειδικότερα, αν η αντικειμενική συνάρτηση f είναι της μορφής:

$$f(x_1, x_2, ..., x_n) = c_1x_1 + c_2x_2, + ... + c_nx_n$$

και όλοι οι περιορισμοί $g_i(i = 1, 2, ..., m)$ είναι της μορφής:

$$\alpha_{i1}x_1 + \alpha_{i2}x_2 + ... + \alpha_{in}x_n \leqslant, =, \geqslant 0$$

τότε ορίζεται ένα πρόβλημα γραμμικού προγραμματισμού (linear programming).

Μια πολύ σημαντική ιδιότητα των συνόλων είναι η κυρτότητα (convexity). Ένα σύνολο D είναι κυρτό όταν όλα τα σημεία που βρίσκονται πάνω στο ευθύγραμμο τμήμα που ενώνει δύο σημεία του x, y ανήκουν επίσης στο D, δηλαδή για κάθε $\lambda \in [0, 1]$ ισχύει :

$$\lambda f(x) + (1 - \lambda)f(y) \geqslant f[\lambda x + (1 - \lambda)y]$$

Μια συνάρτηση f παρουσιάζει τοπικό ελάχιστο (local minimum) στο σημείο $x^* \in D$ όταν υπάρχει περιοχή $U \subset D$ του x^* τέτοια ώστε για κάθε $x \in U$ να ισχύει:

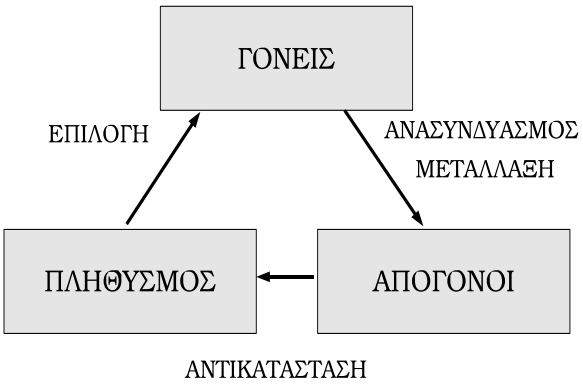
$$f(x^*) \leqslant f(x)$$

Αντίστοιχος είναι οι ορισμός για το τοπικό μέγιστο (local maximum). Τα σημεία τοπικού ελαχίστου και τοπικού μεγίστου καλούνται τοπικά ακρότατα (local extremum) . Αν $U \equiv D$, το ακρότατο ονομάζεται ολικό (global). Η διαδικασία αναζήτησης του ολικού ακρότατου (μεγίστου ή ελαχίστου) μιας συνάρτησης f ορισμένης στο D είναι γνωστή ως ολική βελτιστοποίηση (global optimization). Αν $D = R^n$, το πρόβλημα βελτιστοποίησης είναι χωρίς περιορισμούς (unconstrained optimization), ενώ αν $D \subseteq R^n$, πρόκειται για πρόβλημα βελτιστοποίησης με περιορισμούς (constrained optimization).

Οι μέθοδοι βελτιστοποίησης αξιολογουνται ως προς δυο βασικά τους χαρακτηριστικά, την αποτελεσματικότητα (effectiveness), η οποία σχετίζεται με την ακρίβεια του ολικού βέλτιστου, και την αποδοτικότητα (efficiency), η οποία σχετίζεται με τον υπολογιστικό φόρτο. Στην παρούσα εργασία θεωρείται συμβατικά ότι η βελτιστοποίηση μιας συνάρτησης έγκειται στην εύρεση του ολικού ελαχίστου αυτής ενώ όταν χρησιμοποιείται ο όρος βέλτιστη λύση αναφερόμαστε στη τρέχουσα βέλτιστη λύση.

1.2 Εξελικτικοί Αλγόριθμοι

Ένας εξελικτικός αλγόριθμος (evolutionary algorithm) EA ορίζεται σαν μια αλγοριθμική διαδικασία που διατηρεί ένα πληθυσμό ατόμων (population of individuals), τον οποίο εξελίσσει σύμφωνα με κάποιους κανόνες επιλογής (selection rules) και κάποιους τελεστές (operators)όπως ο ανασυνδυασμός (recombination) και η μετάλλαξη (mutation). Αυτό σημαίνει ότι σε κάθε εκτέλεση έχουμε ένα σύνολο λύσεων σε αντίθεση με άλλους ευρετικούς αλγορίθμους όπου συνήθως σε κάθε εκτέλεση έχουμε μια ενιαία λύση ή ένα μέρος της τελικής λύσης που επιδιώκεται.



Σχήμα 1.1: Εξελικτικός κύκλος

Το μεγαλύτερο μέρος των ΕΑ εμπνέεται από τη βιολογία και τους φυσικούς ή κοινωνικούς μηχανισμούς εξέλιξης (πχ. εξέλιξη των ειδών, κοινωνική εξέλιξη , πολιτισμική κλπ.). Για παράδειγμα μια υποψήφια λύση καλείται άτομο, η κωδικοποίηση της καλείται γενότυπος, η αναπαράσταση (τα διάφορα χαρακτηριστικά) καλούνται φαινότυπος.

Οι απόγονοι δημιουργούνται μέσω του ανασυνδυασμού των γονέων δηλαδή με την ανταλλαγή πληροφορίας μεταξύ των γονέων και μέσω της μετάλλαξης η οποία διαταράσσει περαιτέρω τους απογόνους. Ακολουθεί η χρήση της συνάντησης ποιότητας (fitness function) για την αξιολόγηση (evaluation) των απογόνων και τελικά η επιλογή των ατόμων του πληθυσμού που θα επιβιώσουν στην επόμενη γενιά. Η διαδικασία αυτή ονομάζεται εξελικτικός κύκλος (evolutionary cycle). Στο σχήμα 1.1 παρουσιάζεται ο συνηθέστερος εξελικτικός κύκλος. Το κεντρικό σημείο της έρευνας στους ΕΑ υπήρξε η ευρωστία, η ισορροπία δηλαδή ανάμεσα στην ικανότητα επίλυσης συγκεκριμένων προβλημάτων από την μια μεριά και στην αποτελεσματικότητα που απαιτείται για την επιβίωση σε πολλά διαφορετικά περιβάλλοντα από την άλλη. Οι ΕΑ

```
Evolutionary Algorithm Pseudocode
t:=0;
initialize and evaluate [pop(t)];
while not stopcondition do
pop'(t):=variation [pop(t)];
evaluate [pop'(t)];
pop(t+1):=select [pop'(t),pop(t)];
t:=t+1;
end while;
```

Σχήμα 1.2: Ψευδοκώδικας Εξελικτικού Αλγορίθμου

διατηρούν ένα πληθυσμό ατόμων το οποίο εξελίσσουν σύμφωνα με κάποιους κανόνες επιλογής και κάποιους τελεστές, όπως ο ανασυνδυασμός και η μετάλλαξη. Κάθε άτομο του πληθυσμού αντιπροσωπεύει ένα σημείο του συνόλου των πιθανών λύσεων ενός συγκεκριμένου προβλήματος. Επίσης περιέχει και κάποια γνώση για τους κανόνες του περιβάλλοντος του προβλήματος. Σε κάθε άτομο του πληθυσμού αντιστοιχείται ένα μέτρο της ποιότητας που διαθέτει στο συγκεκριμένο περιβάλλον του προβλήματος το οποίο αντιμετωπίζεται και το οποίο κωδικοποιείται μέσω κάποιας συνάρτησης ποιότητας. Ένας απλός ΕΑ παρουσιάζεται στο σχήμα 1.2 Κατά την επιλογή η προσοχή εστιάζεται σε άτομα υψηλής ποιότητας αξιοποιώντας την διαθέσιμη πληροφορία μέσω της ποιότητας των ατόμων. Ο ανασυδνυασμός και η μετάλλαξη διαταράσσουν την δομή των ατόμων παρέχοντας δυνατότητες διερεύνησης του χώρου.

Ένα σημαντικό χαρακτηριστικό των ΕΑ είναι η δυνατότητα να εξερευνήσουν τις διαφορετικές περιοχές του χώρου αναζήτησης ταυτόχρονα. Η εξερεύνηση αυτή είναι συνήθως συνυφασμένη με την εκμετάλλευση των υποψηφίων λύσεων. Ο στόχος είναι η βέλτιστη εκμετάλλευση των

πληροφοριών που βρίσκονται στους υποψηφίους. Ο αρχικός πληθυσμός ενός ΕΑ συνήθως αρχικοποιείται σε τυχαίες τιμές και εξελίσσεται προς διαδοχικά καλύτερες περιοχές του χώρου αναζήτησης μέσω τυχαίων διαδικασιών της επιλογής, του ανασυνδυασμού και της μετάλλαξης. Το περιβάλλον αποδίδει πληροφορίες σχετικά με την ποιότητα των νέων σημείων αναζήτησης και η διαδικασία επιλογής ευνοεί τα άτομα με την καλύτερη ποιότητα να αναπαράγονται συχνότερα από τα άλλα άτομα του πληθυσμού.

Για την καλύτερη διερεύνηση του χώρου αναζήτησης χρησιμοποιούνται σε πολλές περιπτώσεις ΕΑ στρατηγικές διαποίκιλσης (diversification strategies) [135] με κύριο στόχο να αποφευχθεί η γρήγορη σύγκλιση ολόκληρου του πληθυσμού σε μια περιοχή του χώρου αναζήτησης. Η έννοια της διαποίκιλσης είναι αντίθετη με την έννοια τυχαιότητας (randomization) και επιδιώκει τον χωρισμό και άλλο των συνόλων λύσεων που παρήχθεισαν προηγουμένως σε αντιδιαστολή με τις στρατηγικές ενδυνάμωσης (intensification strategies) ή τυχαιότητας που χρησιμοποιούνται για να ενεργήσουν προς μια βελτίωση των καλύτερα ποιοτικά υποψηφίων.

Το αποτέλεσμα είναι η λύση που διαμορφώνεται από το άτομο με την καλύτερη ποιότητα στον πληθυσμό κατά την διάρκεια ολόκληρης της εξελικτικής διαδικασίας. Αυτό το άτομο δεν είναι στοιχείο του πληθυσμού σε όλες τις φάσεις της εξέλιξης. Επομένως πρέπει να απομνημονευθεί κατά τη διάρκεια της εξέλιξης. Στις περιπτώσεις όμως εκείνες που το καλύτερο ποιοτικά άτομο είναι σίγουρο ότι θα συμμετέχει στην εξέλιξη του πληθυσμού τότε λέμε ότι το άτομο είναι ελιτιστικό.

Είναι γενικά αποδεκτό ότι οποιοσδήποτε ΕΑ πρέπει να διαθέτει πέντε βασικά στοιχεία

- κάποια μέθοδο αναπαράστασης των λύσεων ενός προβλήματος

- κάποια μέθοδο δημιουργίας του αρχικού πληθυσμού
- κάποια μέθοδο (συνάρτηση) υπολογισμού της ποιότητας των λύσεων.
- ορισμένους τελεστές οι οποίοι μεταβάλλουν τον τρέχοντα πληθυσμό δημιουργώντας την νέα γενιά κατά την διάρκεια της αναπαραγωγής και
- τιμές για τις παραμέτρους (μέγεθος πληθυσμού, πιθανότητα ανασυνδυασμού).

Οι γνωστότερες μεθοδολογίες ΕΑ είναι οι:

- Γενετικοί Αλγόριθμοι (Genetic Algorithms - GA) , οι οποίοι προτάθηκαν από τον Holland [109]. Συνήθως χρησιμοποιούν δυαδική αναπαράσταση η οποία είναι σχετικά ανεξάρτητη από το πρόβλημα. Η μετάλλαξη αναστρέφει τα δυαδικά ψηφία με κάποια πιθανότητα και θεωρείται πολύ σημαντικός τελεστής.
- Εξελικτικές Στρατηγικές (Evolutionary Strategies - ES), οι οποίες προτάθηκαν από τον Rechenberg[179], με επιλογή, μετάλλαξη και μέγεθος πληθυσμού ένα. Ο Schwefel[192] εισήγαγε τον ανασυνδυασμό και πληθυσμούς με περισσότερα του ενός μέλη.

  Τυπικά, οι ES χρησιμοποιούν κωδικοποίηση με πραγματικές τιμές. Η μετάλλαξη είναι πολύ σημαντική. Ο αριθμός των απογόνων που αναπαράγονται είναι μεγαλύτερος του πληθυσμού των γονέων και γίνεται επιλογή των απογόνων που θα επιβιώσουν.
- Εξελικτικός Προγραμματισμός (Evolutionary Programming - EP), που αναπτύχθηκε από τον Fogel. Χρησιμοποιεί κωδικοποίηση προσαρμοσμένη στο πρόβλημα. Η μετάλλαξη είναι ο κύριος γενετικός τελεστής. Ο ανασυνδυασμός χρησιμοποιείται σπάνια.

Υπάρχουν πάρα πολλές υβριδικές τεχνικές οι οποίες συμπεριλαμβάνουν στοιχεία των παραπάνω μεθοδολογιών. Αν και το εννοιολογικό πλαίσιο όλων των ΕΑ είναι ίδιο οι υλοποιήσεις τους διαφέρουν σε πολλά σημεία. Για παράδειγμα υπάρχει μια μεγάλη ποικιλία μεθόδων επιλογής. Επίσης, η αναπαράσταση των ατόμων ποικίλει από δυαδικές συμβολοσειρές μέχρι διανύσματα πραγματικών αριθμών. Τέλος η βαρύτητα των δυο βασικών τελεστών (του ανασυνδυασμού και της μετάλλαξης) όπως επίσης και οι υλοποιήσεις τους διαφέρουν κατά πολύ μεταξύ των διαφορετικών μοντέλων ΕΑ.

1.2.1 Γενετικοί Αλγόριθμοι

Η ανάπτυξη τους ξεκίνησε την δεκαετία του 1960 από τον Holland και τους συνεργάτες του. Οι γενετικοί αλγόριθμοι GA χρησιμοποιούν συνήθως μια αναπαράσταση η οποία είναι ανεξάρτητη από το πρόβλημα, δηλαδή συμβολοσειρές δυαδικών ψηφίων[108].

Αρχικά ένας πληθυσμός παράγεται τυχαία από ένα σύνολο δυαδικών συμβολοσειρών. Μετά την αρχικοποίηση επιλέγονται οι γονείς (parents) σύμφωνα με μια συνάρτηση πιθανότητας η οποία βασίζεται στην σχετική ποιότητα των ατόμων του πληθυσμού. Δημιουργείται δηλαδή με βάση τη σχετική ποιότητα των ατόμων ένας ενδιάμεσος πληθυσμός (intermediate population). Όσο καλύτερη είναι η ποιότητα ενός ατόμου, τόσο αυξάνονται οι πιθανότητες να επιλεγεί περισσότερες φορές σαν γονέας για την αναπαραγωγή απογόνων (offspring).

Γενικά, από N γονείς αναπαράγονται N παιδιά μέσω διασταύρωσης (crossover) όπως ονομάζεται ο ανασυνδυασμός στην περίπτωση των GA. Σε κάθε ζευγάρι γονέων ο τελεστής διασταύρωσης εφαρμόζεται με μια πιθανότητα p_c. Για παράδειγμα στη διασταύρωση ενός σημείου τα άτομα κό-

Genetic Algorithm Pseudocode

```
t:=0;
initialize and evaluate [pop(t)];
while not stopcondition do
pop'(t):=variation [pop(t)];
evaluate [pop'(t)];
Reproduction[pop'(t),pop(t)];
Crossover[pop'(t),pop(t)];
Mutation[pop(t+1)];
evaluate [pop(t+1)];
t:=t+1;
end while;
```

Σχήμα 1.3: Ψευδοκώδικας Γενετικού Αλγορίθμου

βονται σε ένα σημείο και δημιουργούνται δυο νέες συμβολοσειρές οι οποίες και ανασυνδυάζονται με την ανταλλαγή των άκρων τους. Έτσι παράγονται δυο νέες συμβολοσειρές που καλούνται απόγονοι. Ο ψευδοκώδικας ενός απλού GA παρουσιάζεται στο σχήμα 1.3.

Οι τελεστές της επιλογής και της διασταύρωσης μπορουν να θεωρηθούν σαν ένα βήμα που ονομάζεται βήμα συνεργασίας του GA (cooperation step). Υπάρχουν αρκετά είδη διασταυρώσεων που χρησιμοποιούνται. Στη συνέχεια θα περιγράψουμε σε συντομία χρησιμοποιώντας τον γενικότερο όρο του ανασυνδυασμού, μερικά από αυτά:

- Ανασυνδυασμός δειγμάτων: Στον ανασυνδυασμό δειγμάτων (segmented recombination) ο αριθμός των σημείων ανασυνδυασμού δεν είναι καθορισμένος αλλά μπορεί να ποικίλει γύρω από μια αναμενόμενη τιμή. Αυτό επιτυγχάνεται από ένα ποσοστό αλλαγής του δείγματος το οποίο καθορίζει την πιθανότητα κατάληξης του δείγματος σε οποιοδήποτε σημείο της συμβολοσειράς. Εάν το p_c είναι το ποσοστό αλλαγής τότε είναι αναμενόμενες

περίπου 1^*p_c μονάδες ανασυνδυασμού.

- Ανασυνδυασμός Μετατόπισης: Η τεχνική αυτή μπορεί να χρησιμοποιηθεί σε συνδυασμό με οποιοδήποτε ανασυνδυασμό πολλαπλών σημείων. Σύμφωνα με τον ανασυνδυασμό μετατόπισης (shuffle recombination) μετατοπίζονται τυχαία οι θέσεις των bits και των δύο γονέων.

- Ανασυνδυασμός Στίξης: Ο ανασυνδυασμός αυτός παρουσιάζει ιδιαίτερο ενδιαφέρον μιας και είναι η μόνη γνωστή προσπάθεια αυτοπροσαρμογής τόσο των αριθμών όσο και των θέσεων των σημείων ανασυνδυασμού σε ένα ανασυνδυασμό πολλαπλών σημείων. Τούτο επιτυγχάνεται προσθέτοντας ένα bit στην υπόλοιπη αλφαριθμοσειρά ενός ατόμου. Τα bits ανασυνδυασμού στίξεως (punctuated recombination) και τα υπόλοιπα ανταλλάσσονται ανάμεσα στους γονείς ή παραμένουν αμετάβλητα[16].

Τα άτομα που προκύπτουν στον ενδιάμεσο πληθυσμό από το πρώτο βήμα αντικαθιστούν όλα ή ένα μέρος των ατόμων στον αρχικό πληθυσμό. Υπάρχουν δυο μοντέλα αντικατάστασης το μοντέλο ολικής γενεαλογικής αντικατάστασης (generational replacement model) και σταθερής αντικατάστασης (state steady replacement model). Στο μοντέλο ολικής αντικατάστασης ο ενδιάμεσος πληθυσμός έχει το ίδιο μέγεθος με τον αρχικό πληθυσμό, που ανανεώνει ολόκληρο τον πληθυσμό σε μια γενεά ενώ στο μοντέλο σταθερής αντικατάστασης ο ενδιάμεσος πληθυσμός έχει πολύ μικρότερο μέγεθος από τον αρχικό πληθυσμό. Στο μοντέλο σταθερής αντικατάστασης ο απόγονος δεν αντικαθιστά απαραίτητα τους γονείς του αλλά μπορεί να πάρει τη θέση οποιονδήποτε άλλων ατόμων. Η εκτέλεση ολοκληρώνει μετά από ένα προκαθορισμένο αριθμό γενεών. Πιο λεπτομερειακή περιγραφή των GA μπορεί να βρει ο αναγνώστης στα

[99, 31].

1.2.2 Εξελικτικές Στρατηγικές

Η ανάπτυξη των Εξελικτικών Στρατηγικών ES ξεκίνησε από τον Rechenberg στα 1960 για την επίλυση υδροδυναμικών προβλημάτων. Οι πρώτες εργασίες στον χώρο των ES ήταν περισσότερο τεχνικές που ταίριαζαν πιο πολύ στην τεχνική της προσομοιούμενης ανόπτησης παρά στους εξελικτικούς αλγορίθμους με δεδομένο ότι είχαν την δυνατότητα να χειριστούν μόνο δύο άτομα. Η πρώτες ES που βασίζονταν σε πληθυσμούς ατόμων εμφανίστηκαν την δεκαετία του 70 [179, 192] με επικρατέστερη την (λ,μ)-στρατηγική που κυριάρχησε από το 1977. Ο απόγονος που δημιουργείται μετά από ανασυνδυασμό και μετάλλαξη των γονέων και σε κάθε γενιά τα μ καλύτερα παιδιά αντικαθιστούν τον πατρικό πληθυσμό. Σε μια παραλλαγή αυτής της στρατηγικής τα μ καλύτερα άτομα γίνονται και οι νέοι γονείς στην νέα γενιά.

Οι ES δουλεύουν με πίνακες πραγματικών διανυσμάτων. Ο απόγονος δημιουργείται εφαρμόζοντας διωνυμικές κατανομές (με αναμενόμενη τιμή μηδέν και διασπορά σ^2 στον γονέα και είτε ο απόγονος γίνεται ο γονέας της επόμενης γενιάς (εάν είναι καλύτερος του γονέα) είτε ο γονέας επιβιώνει. Στην περίπτωση των διμελών ES (χρήση μόνο δύο ατόμων) ένα άτομο δημιουργείται από ένα μόνο γονέα μέσω της πρόσθεσης κανονικά κατανεμημένων τυχαίων διανυσμάτων με αναμενόμενη τιμή μηδέν και τυπική απόκλιση σ (το ίδιο σ χρησιμοποιείται για όλα τα στοιχεία του διανύσματος. Ο αλγόριθμος του σχήματος 1.4 είναι ένα απλό παράδειγμα ES.

```
Evolutionary Strategy
Q=pop_i for a (λ + μ) strategy, Q=0 for a (μ, λ)-strategy with λ > μ ≥ 1
determine an initial set pop(0) of size μ
i := 0
repeat
generate pop'(i) of size λ by combining and mutating individuals of pop(i)
select μ individuals in pop'(i) ∪ Q to put in pop(i+1)
i := i + 1
until termination condition is met
```

Σχήμα 1.4: Ψευδοκώδικας Εξελικτικής Στρατηγικής

1.2.3 Εξελικτικός Προγραμματισμός

Ο εξελικτικός προγραμματισμός EP είναι παρόμοιος με τις ES και αναπτύχθηκε στην δεκαετία του 60 από τον Fogel [87]. Οι κυριότερες διαφορές ανάμεσα στις δυο μεθόδους είναι στη φάση της επιλογής που στην περίπτωση του EP γίνεται στοχαστικά μέσω αγώνων (tournament) και οι χειρότερες λύσεις αφαιρούνται αιτιοκρατικά (deterministic) . Πρέπει να σημειωθεί ότι δεν χρησιμοποιείται κανένας μηχανισμός επανασυνδυασμού στον EP ενώ στις ES μπορούμε να χρησιμοποιήσουμε μηχανισμούς επανασυνδυασμού[103].

1.2.4 Πολιτισμικοί Αλγόριθμοι

Οι πολιτισμικοί αλγόριθμοι (Cultural Algorithms - CA) είναι μια κατηγορία EA [181, 49] στους οποίους αρχικά ένας πληθυσμός ατόμων που αντιπροσωπεύουν τον χώρο λύσεων παράγεται τυχαία για να δημιουργήσει την πρώτη γενεά. Το αρχικό διάστημα πεποίθησεων (χώρος αποδεκτών λύσεων) (belief space) είναι κενό. Σε κάθε γενεά ο πολιτισμικός αλγόριθμος εξελίσσει ένα πληθυσμό ατόμων με βάση το πλαίσιο Vote - Inherit - Promote (VIP) (ψηφοφορία

- κληρονομιά - προαγωγή). Κατά τη διάρκεια της διαδικασίας ψηφοφορία τα άτομα των πληθυσμών αξιολογούνται για τη συμβολή τους στο διάστημα πεποίθησης χρησιμοποιώντας τη λειτουργία της αποδοχής. Εκείνες οι πεποιθήσεις που συμβάλλουν πιο πολύ στη λύση προβλήματος επιλέγονται ή ψηφίζονται για να συμβάλλουν στο τρέχον διάστημα πεποίθησης. Το διάστημα ή αλλιώς χώρος πεποιθήσεων τροποποιείται όταν συνδυάζονται οι κληρονομημένες πεποιθήσεις με τις πεποιθήσεις που έχουν προστεθεί κατά την τρέχουσα γενιά. Στην συνέχεια ο ενημερωμένος χώρος πεποιθήσεων χρησιμοποιείται για να επηρεάσει την εξέλιξη του πληθυσμού. Κατά τη διάρκεια της τελευταίας φάσης ένας νέος πληθυσμός αναπαράγεται χρησιμοποιώντας ένα βασικό σύνολο εξελικτικών τελεστών. Ο κύκλος VIP τελειώνει με κάποιες συνθήκες τερματισμού. Συνήθως η διαδικασία τερματίζεται όταν ανιχνεύεται μηδαμινή ή ελάχιστη διαφορά ανάμεσα στους πληθυσμούς ενός πεπερασμένου πλήθους γενεών ή όταν προκύψει κάποια γνώση στο διάστημα πεποιθήσεων. Ο βασικός πολιτισμικός αλγόριθμος περιγράφεται από τον κώδικα του σχήματος 1.5.

1.3 Άλλες μέθοδοι

1.3.1 Προσομοιούμενη Ανόπτηση

Η προσομοιούμενη ανόπτηση (Simulated Annealling - SA) είναι μια τεχνική βελτιστοποίησης, η οποία βασίζεται στις αρχές της στατιστικής μηχανικής. Η πρωτοτυπία της μεθόδου έγκειται στην αποφυγή των τοπικών ακρότατων, μέσω πραγματοποίησης περιορισμένου αριθμού μη βέλτιστων βημάτων με βάση πιθανοτικά κριτήρια. Η προσομοιωμένη ανόπτηση βρήκε εφαρμογή

```
Cultural Algorithm Pseudocode
t:=0;
Initialize pop(t);
Initialize blf(t);
repeat
pop(t);
vote[blf(t), accept(pop(t))];
adjust(blf(t));
evolve[pop(t), influence(blf(t))];
t:=t+1;
select pop(t) from pop(t-1);
until (termination condition achieved)
end
```

Σχήμα 1.5: Ψευδοκώδικας Πολιτισμικού Αλγορίθμου

κυρίως σε μεγάλης κλίμακας προβλήματα συνδυαστικής βελτιστοποίησης (combinatorial optimization). Στην κατηγορία αυτή περιλαμβάνονται προβλήματα στα οποία ο χώρος πολιτικής είναι διακριτός, περιέχει δηλαδή πεπερασμένο αριθμό εφικτών λύσεων, ο οποίος αυξάνει εκθετικά με τον αριθμό των μεταβλητών ελέγχου. Αντίθετα, σχετικά περιορισμένο είναι ως τώρα το πεδίο εφαρμογής της μεθόδου σε προβλήματα συνεχών μεταβλητών, στα οποία επικεντρώνεται το ενδιαφέρον της παρούσας εργασίας.

Έχει προσελκύσει το ενδιαφέρον αρκετών επιστημόνων από διάφορους κλάδους. Σαν τεχνική θα λέγαμε ότι είναι ένας στατιστικός μηχανισμός που προσομοιώνει την φυσική διαδικασία ανόπτησης που χρησιμοποιείται για την ψύξη ενός στερεού σώματος έτσι ώστε αυτό να γίνει ένας τέλειος κρύσταλλος. Ο Metropolis και οι συνεργάτες τους [149] ήταν οι πρώτοι που περιέγραψαν έναν αλγόριθμο που προσομοίωνε την διαδικασία της ανόπτησης. Ο Kirkpatrick [120] και οι συνεργάτες τους πρότειναν τη χρήση αυτού του αλγορίθμου για την επίλυση ενός προβλήματος βελτιστοποίησης. Η προσομοιούμενη ανόπτηση είναι μια τυχαία μέθοδος με την

οποία μειώνεται η πιθανότητα να παγιδευτουμε σε τοπικά ελάχιστα και να οδηγηθούμε σε χειρότερες λύσεις. Λαμβάνοντας υπόψη την γειτονιά $N(s)$ ενός προβλήματος βελτιστοποίησης οι επιλογές μέσα σε αυτό το σύνολο λύσεων γίνονται τυχαία. Η μετακίνηση από μια λύση s σε μια λύση s' γίνεται αποδεκτή μόνο αν:

- s' είναι καλύτερο από το s
- s' είναι χειρότερο από το s αλλά $e^{-(f(s)-f(s'))r} > R$

όπου T είναι μια παράμετρος ελέγχου η οποία καλείται θερμοκρασία και $R \in [0,1]$ είναι μια ομοιόμορφη κατανομή τυχαίων αριθμών. Ο αλγόριθμος αποφεύγει την προσκόλληση σε τοπικά βέλτιστα μέχρι τα τελευταία στάδια της αναζήτησης κατά τα οποία η θερμοκρασία είναι πολύ χαμηλή και ο αλγόριθμος έχει ήδη μια καλή λύση.

Η εφαρμογή της προσομοιωμένης ανόπτησης σε προβλήματα βελτιστοποίησης προϋποθέτει την ύπαρξη μιας γεννήτριας διανυσμάτων, η οποία παράγει νέα σημεία στην γειτονιά της εκάστοτε λύσης. Ως γεννήτρια μπορεί να χρησιμοποιηθεί είτε ένας προσδιοριστικός αλγόριθμος τοπικής αναζήτησης είτε ένας αλγόριθμος παραγωγής τυχαίων διανυσμάτων. Στην πρώτη περίπτωση η κατεύθυνση της αναζήτησης είναι μονοσήμαντη, αφού επιλέγονται πάντοτε ολοένα και καλύτερες λύσεις, καταλήγοντας τελικά στην περιοχή κάποιου τοπικού ακρότατου. Αυτό δεν είναι επιθυμητό, αφού η στρατηγική ανόπτησης προϋποθέτει τη δυνατότητα εκτέλεσης μη βέλτιστων βημάτων. Από την άλλη πλευρά, η εφαρμογή τυχαίων βημάτων επιτρέπει τη διαφυγή από τοπικά ακρότατα αλλά συνεπάγεται μεγάλο υπολογιστικό φόρτο, αφού κατά κανόνα απαιτείται μεγάλος αριθμός δοκιμών μέχρι να επιτευχθεί σύγκλιση. Μεγάλη σημασία για την επιτυχή λειτουργία του αλγορίθμου έχει το χρονοδιάγραμμα ανόπτησης (annealing schedule) του προσομοιωμένου

Simulated Annealing Pseudocode

```
t:=T(0), n:=1;
s_best := s;
repeat
Generate neighboring solution s' ∈ N(s);
△f := f(s) − f(s') ;
if △f ≤ 0 or exp(−△f/t) > random[0,1) then s := s';
if f(s) > f(s_best) then s_best := s;
t:=T(n);
n:=n+1;
until termination criterion fulfilled;
return s_best
end;
```

Σχήμα 1.6: Ψευδοκώδικας Προσομοιούμενης Ανόπτησης

θερμοδυναμικού συστήματος. Το χρονοδιάγραμμα ανόπτησης συνίσταται από τους ακόλουθους παράγοντες:

- την αρχική θερμοκρασία
- τη συνάρτηση μείωσης της θερμοκρασίας
- το μήκος των κύκλων θερμικής ισορροπίας
- τη συνθήκη τερματισμού του αλγορίθμου

Στο σχήμα 1.6 έχουμε τον ψευδοκώδικα για την προσομοιούμενη ανόπτηση.

1.3.2 Αποτρεπτική Αναζήτηση

Η αποτρεπτική αναζήτηση (Tabu Search (TS)) αναπτύχθηκε από τον Glover[97]. Η ευρετική αυτή μέθοδος συνδυάζει μια τοπική διαδικασία αναζήτησης με διάφορους κανόνες αντιανακύκλωσης όπως αποκαλούνται (anti-cycling) που αποτρέπουν την προσκόλληση σε τοπικά ελάχιστα κατά την διάρκεια της αναζήτησης. Από το 1986 μέχρι σήμερα η μέθοδος έχει τροποποιηθεί αρκετά και έχουν ενσωματωθεί αρκετά νέα στοιχεία που έχουν βελτιώσει κατά πολύ την απόδοση της.

Ο βασικός αλγόριθμος της αποτρεπτικής αναζήτησης περιγράφεται από τον Glover ως εξής. Έχουμε μια συνάρτηση $f(x)$ η οποία βελτιστοποιείται σε ένα σύνολο X. Ο αλγόριθμος ξεκινάει όπως ένας οποιοδήποτε συνηθισμένος αλγόριθμος τοπικής αναζήτησης και προχωράει σταδιακά από ένα σημείο (λύση) σε ένα άλλο μέχρι να ικανοποιηθούν τα κριτήρια τερματισμού. Για κάθε $x \in X$ υπάρχει γειτονιά $N(x) \subset X$.

Η αποτρεπτική αναζήτηση υπερέχει μιας απλής μεθόδου τοπικής αναζήτησης στο ότι περιέχει μια στρατηγική κατά την διερεύνηση του χώρου $N(x)$ σύμφωνα με την οποία αντικαθίσταται αυτή η γειτονιά λύσεων από μια άλλη $N^*(x)$). Εκείνο που χαρακτηρίζεται την συγκεκριμένη μεθοδολογία είναι οι απαιτήσεις σε μνήμη προκείμενου να καθοριστεί η γειτονιά $N^*(x)$) και ο τρόπος με τον οποίο θα διερευνηθεί καλύτερα ο χώρος αναζήτησης. Ο ψευδοκώδικας της αναζήτησης με απαγορεύσεις δίνεται στο σχήμα 1.7. Η μέθοδος αποτρεπτικής αναζήτησης , προσομοιώνει τις διεργασίες της ανθρώπινης μνήμης. Η βασική αρχή της μεθόδου συνίσταται στη διατήρηση μιας απαγορευμένης λίστας (tabu list), στην οποία αποθηκεύονται όλες οι πρόσφατες μετακινήσεις που πραγματοποιούνται κατά τη διαδικασία βελτιστοποίησης. Όπως

Tabu Search Pseudocode
$t := 0$;
$s_{best} := s$;
repeat
Find best solution $s' \in N(s)$ **with** $s' \notin t$;
$s := s'$;
$t := t \cup s$;
if $f(s) > f(s_{best})$ **then** $s_{best} := s$;
until termination criterion fulfilled;
return s_{best};
end;

Σχήμα 1.7: Ψευδοκώδικας Αποτρεπτικής Αναζήτησης

προδικάζει η ονομασία της, η λίστα αυτή χρησιμοποιείται για να εμποδίζει την αναζήτηση σε περιοχές που έχουν ήδη εξερευνηθεί. Προτού επιλεγεί μια υποψήφια λύση, ελέγχεται αν αυτή είναι καταχωρημένη ή όχι στη λίστα και στην περίπτωση που είναι, δεν γίνεται αποδεκτή. Με τον τρόπο αυτό επιτυγχάνεται η διαφυγή από τοπικά ακρότατα, αφού θεωρητικά διερευνάται όσο το δυνατό μεγαλύτερο εύρος του εφικτού χώρου.

1.3.3 Καθοδηγούμενη Τοπική Αναζήτηση

Η καθοδηγούμενη τοπική αναζήτηση (Guided Local Search - GLS) είναι μια ευρετική τεχνική που καλύπτει ένα ευρύ φάσμα προβλημάτων βελτιστοποίησης. Η καθοδηγούμενη τοπική αναζήτηση εκμεταλεύεται διάφορες πληροφορίες σχετικά με τον χώρο λύσεων και καθοδηγεί την τοπική αναζήτησης στις πιο καλές περιοχές του χώρου αναζήτησης. Τούτο γίνεται εφικτό με την μεγιστοποίηση της συνάρτησης κόστους του προβλήματος που περιέχει αρκετούς όρους ποινής(penalty) . Η τοπική αναζήτηση περιορίζεται από τους όρους ποινής και στρέφεται στις

```
Guided Local Search Pseudocode
k ← 0;
s(0) := random or heuristically generated solution in S;
for i := 1 until M do /* set all penalties to 0*/
pop(i) := 0 ;
while StoppingCriterion do
begin
h := g + λ * ∑ pop(i) * I(i);
s(k + 1) := LocalSearch(s(k), h);
for i := 1 until M do
util(i) := I(i)(s(k + 1)) * c(i)/(1 + pop(i)) ;
for each i such that util(i) is maximum do
pop(i) := pop(i + 1);
k := k + 1;
end
s* := best solution found with respect to cost function g;
return s*;
end
```

Σχήμα 1.8: Ψευδοκώδικας Κατευθυνόμενης Τοπικής Αναζήτησης

περιοχές του χώρου αναζήτησης που υπόσχονται περισσότερα. Ο κώδικας αυτής της τεχνικής δίνεται στο σχήμα 1.8 ενώ μια αναλυτική περιγραφή των χαρακτηριστικών αυτής της τεχνικής μπορεί ο αναγνώστης μας να δεί στην διατριβή του Βουδούρη [205].

1.4 Υβριδικές προσεγγίσεις

Πολλές μελέτες έχουν γίνει μέχρι σήμερα για την βελτίωση της ποιότητας των αποτελεσμάτων που επιτυγχάνονται με EAs και κυρίως των GA [202]. Μια τέτοια προσέγγιση είναι και η δημιουργία ενός μοντέλου στο οποίο διάφοροι αλγόριθμοι καλούνται να συνλειτουργήσουν

Memetic Algorithm Pseudocode

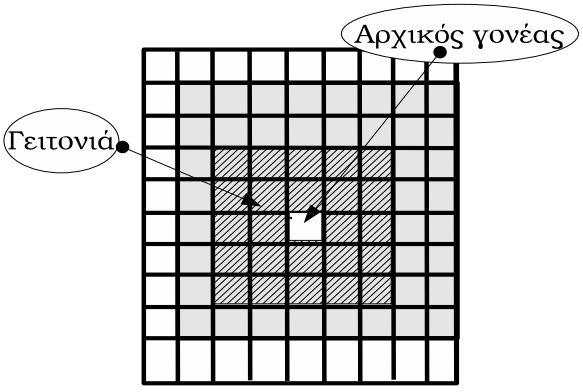
```
t:=0; Initialise pop(t) randomly;
For i=1 to m (m=pop size)
Perform local search in the neighbourhood of i
(i being the current individual);
Evaluate fitness of i and its neighbourhood explored;
Make i the best individual found;
End for
Repeat
t:=t+1; Select parents from pop(t);
Generate offspring applying recombination
to the parents selected ;
If an individual is selected to undergo mutation,
then apply local search;
Evaluate fitness of current individual and its neighbours;
Adopt best individual ;
Until stopping condition is reached;
```

Σχήμα 1.9: Ψευδοκώδικας Μιμητικού Αλγορίθμου

προκειμένου να επωφεληθούμε κατα τη διερεύνηση του χώρου λύσεων από τα καλύτερα χαρακτηριστικά της κάθε μεθόδου από αυτές. Το μοντέλο αυτό καλείται υβριδικό και είναι ανάλογο ενός βιολογικού υβριδικού οργανισμού δυο συγγενών οργανισμών. Γενικά μπορούμε να ισχυριστούμε ότι ένας ΕΑ μπορεί να υβριδοποιηθεί με οποιοδήποτε άλλο αλγόριθμο αναζήτησης. Ένας τέτοιος υβριδικός αλγόριθμος που χρησιμοποιεί παραδοσιακές ευρετικές μεθοδους και εξελικτικές τεχνικές είναι και ο μιμητικός αλγόριθμος (memetic algorithm) ΜΑ . Η ονομασία αυτή προέρχεται από τον ελληνικής προέλευσης όρο **meme** ***Μίμησις***. Ο ψευδοκώδικας ενός απλού ΜΑ παρουσιάζεται στο σχήμα 1.9.

Ένας ΜΑ όπως κάθε μεταευρετικός αλγόριθμος έχει γενικώς πιο πολύπλοκη δομή από ένα εξελικτικό αλγόριθμο καθώς είναι σύνθεση ευρετικών και εξελικτικών αλγορίθμων που περιέχουν στοιχεία μίμησης βιολογικών ή κοινωνικών διαδικασιών. Ανάλογα με την υλοποίηση εργάζεται με κωδικοποιημένο σύνολο λύσεων ή με τις λύσεις αυτές καθεαυτές. Στην πρώτη περίπτωση συγγενεύει πιο πολύ με τους γενετικούς αλγορίθμους στην δεύτερη με τους εξελικτικούς. Εκτελεί αναζητήσεις χρησιμοποιώντας ένα πληθυσμό λύσεων και όχι μια μοναδική λύση. Ο χρόνος εκτέλεσης του είναι εν γένει μεγαλύτερος από αυτόν κάποιου αντίστοιχου ευρετικού αλλά σε αντιστάθμισμα αναμενουμε λύσεις καλύτερης ποιότητας από αυτές που παράγει ο αντίστοιχος ευρετικός ή εξελικτικός.

Η απόδοση των ΜΑ βελτιώνεται με την χρήση της τοπικής επιλογής (Local selection) σύμφωνα με την οποία ο δεύτερος από τους δύο γονείς επιλέγεται από τη γειτονιά (neighbourhood) του πρώτου και όχι από τον συνολικό πληθυσμό. Ως γειτονιά ορίζεται το σύνολο των ατόμων που ανήκουν μέσα στα όρια ενός γεωμετρικού σχήματος με κέντρο τον αρχικό γονέα. Ο πληθυσμός των ατόμων αντιμετωπίζεται ως μια σφαιρική επιφάνεια, ώστε στην περίπτωση που κάποιο σχήμα βγαίνει έξω από τα όρια του πίνακα τότε αυτό θεωρούμε ότι συνεχίζεται από την απέναντι πλευρά (βλεπε σχήμα 1.10). Με την εφαρμογή της τοπικής επιλογής LS επιτυγχάνεται η δημιουργία γειτονιών ατόμων με κοινά χαρακτηριστικά μέσα στον πληθυσμό. Αυτό οδηγεί τον αλγόριθμο σε μια παράλληλη αναζήτηση λύσεων σε διαφορετικές περιοχές του χώρου. Μια πολυ καλή ανάλυση υβριδικών μεταευρετικών τεχνικών στους οποίους ανήκουν και οι ΜΑ έγινε από τον Talbi [202]. Ο όρος μεταευρετικός αναφέρεται είτε σε ΕΑς είτε σε παραδοσιακούς ευρετικους αλγορίθμους που μπορούν να εφαρμοστούν σε διαφορετικά προβλήματα σε αντίθεση με τους ευρετικούς αλγορίθμους που σχεδιάστηκαν για την επιλυση συγκεκριμένων προβλημάτων.



Σχήμα 1.10: Γειτονιά σε σχήμα τετραγώνου

Συμβολισμός	Περιγραφή - Χαρακτηρισμός
L	Χαμηλού επιπέδου
H	Υψηλού επιπέδου
R	Ανεξάρτητη Εξέλιξη
C	Συνεξέλιξη
hom	Ομογενείς
het	Ετερογενείς
par	Διατμηση Χώρου Αναζήτησης
glo	Όλος ο χώρος αναζήτησης
spe	Ειδικού σκοπού προβλήματα
gen	Γενικού σκοπού προβλήματα

Πίνακας 1.1: Ταξινόμηση του Talbi για μεταευρετικούς αλγορίθμους

Στην ανάλυση αυτή γίνεται μια πολύ καλή αναφορά σε θέματα σχεδιασμού και εφαρμογής των μεταευρετικών αλγορίθμων. Στα θέματα σχεδιασμού έχουμε μια ιεράρχηση-κατηγοριοποίηση του τρόπου με τον οποίο γίνεται η υβριδοποιηση των αλγορίθμων.

Στον πίνακα 1.4 έχουμε σε συντομία την ταξινόμηση για θέματα σχεδιασμού υβριδικών μεταευρετικών αλγορίθμων.

1.4.1 Παραδείγματα

Όταν διάφοροι πληθυσμοί εξελίσσονται ανεξάρτητα καλούνται νησίδες[42, 3]. Οι ανεξάρτητοι μεταξύ τους ΕΑ (ένας ανά νησίδα) συνεργάζονται μέσω της ανταλλαγής ατόμων. Η ανταλλαγή αυτή ορίζεται ως μετανάστευση.

Ένα τέτοιο παραδειγμα αποτελεί και η προσέγγιση που παρουσιάζεται από τον Tanese [203] στην οποία χρησιμοποιείται ένα 4-D υπερκύβος με 64 επεξεργαστές ώς τοπολογία των πληθυσμών. Περιοδικά γόνοι με καλή ποιότητα, τυχαία διαλεγμένοι μεταναστεύουν μεταξύ των γειτονικών επεξεργαστών και αντικαθιστούν τους χειρότερους γόνους των ληπτών. Η ανταλλαγή δεν γίνεται πάντα με τους ιδιους επεξεργαστές, αλλα σε διαφορετικές διαστάσεις του υπερκύβου κάθε φορά.

Οι αλγόριθμοι που χρησιμοποιουνται είναι ομογενείς (δηλ. όλοι GA) και κάθενας από αυτούς λύνει το ίδιο πρόβλημα στο ίδιο διάστημα αναζήτησης. Το μοντέλο που χρησιμοποιειται χαρακτηριζεται ως υψηλού επιπέδου υβριδική συνεξέλιξη(High Level Co-Evolution Hybrid) και με βάσει την ταξινόμιση του Talbi ορίζεται ώς HCH(GA)(hom,glo,gen).

Ο Levine [131] χρησιμοποιήσε τρία είδη υβριδοποίησης. Πρώτον ένα GA του οποίου ο πληθυσμός βελτιώνεται από ένα αλγόριθμο τοπικής αναζήτησης (Local Search - LS) σε κάθε γενεά. Ο αλγόριθμος (LS) πραγματοποιεί τις ανεξάρτητες αναζητησεις για να βελτιώσει τα άτομα ενός πληθυσμού. Πρόκειται για μια χαμηλού επιπέδου υβριδική συνεξέλιξη (Low Level Co-evolutionary Hybrid) ετερογενών αλγορίθμων (GA και LS) και ορίζεται ως

LCH(GA(LS))(het,glo,gen).

Στην συνέχεια στην εργασία αυτή παρουσιάζεται ένα δευτερο επίπεδο στο οποίο ο αρχικός πληθυσμός δημιουργείται από ένα άπλησστο ευρετικό αλγόριθμο (Greedy Heuristic - GH) .

Τα αποτελέσματα αυτού χρησιμοποιούνται από ένα χαμηλού επιπέδου GA. Το μοντέλο που προκύπτει χαρακτηρίζεται ως υψηλού επιπέδου υβριδική ανεξαρτησία (High Level Relay Hybrid) και συμβολίζεται ώς εξής:

HCH(HRH(GH+LCH(GA(LS))(het,glo,gen)(het,glo,gen).

Σε τρίτη φάση (επίπεδο) οι πληθυσμοί ομαδοποιουνται σε νησίδες και η περιγραφή του μοντέλου με βάσει την ταξινομία του Talbi γίνεται ως εξής:

HCH(HRH(GH+LCH(GA(LS)) (het,glo,gen)) (het,glo,gen)) (hom,glo,gen).

Οι ΜΑ στην ταξινόμηση του Talbi ταξινομούνται όπως ακολούθως:

HCH(HRH(EA+LCH(EA(LS))(het,par,gen)(het,glo,gen).

Κεφάλαιο 2

Παράλληλος Προγραμματισμός

Σκοπός του κεφαλαίου αυτού ειναι να φέρει σε επαφή τον αναγνώστη με τις βασικές έννοιες της παράλληλου προγραμματισμού και της βιβλιοθήκης MPI. Το κεφάλαιο αυτό οργανώνεται ως εξής: στην επόμενη ενότητα παρουσιάζουμε σύντομα τις αρχιτεκτονικές των παράλληλων υπολογιστών. Στην ενότητα 2.2 αναφέρουμε τα κίνητρα για την μετακίνηση προς στην περιοχή παράλληλου υπολογισμού χαμηλού κόστους. Στην ενότητα 2.3 παρουσιάζουμε την αρχιτεκτονική μιας συστοιχίας υπολογιστών (cluster of computers). Στην ενότητα 2.4 περιγράφουμε την ταξινόμηση των συστοιχιών σε διάφορες κατηγορίες. Στην ενότητα 2.5 αναφέρουμε σύντομα τα περιβάλλοντα και τα εργαλεία που χρησιμοποιούνται για τον παράλληλο προγραμματισμό. Στην ενότητα 2.6 παρουσιάζουμε τις βασικές συναρτήσεις της βιβλιοθήκης MPI. Στην ενότητα 2.7 παραθέτουμε ορισμένα μοντέλα παράλληλου προγραμματισμού. Τέλος, στην ενότητα 2.8 παρουσιάζουμε μέτρα απόδοσης που χρησιμοποιούνται στην παράλληλη επεξεργασία.

2.1 Γενικές Αρχιτεκτονικές Υπολογιστών

Ο Flynn [86] ταξινόμησε τους υπολογιστές με βάση τον αριθμό των εντολών και των ρευμάτων δεδομένων που μπορούν να επεξεργαστούν ταυτόχρονα στις ακόλουθες τέσσερις κατηγορίες:

- Μοναδική Εντολή Μοναδικό Δεδομένο (Single Instruction Single Data - SISD): Ένα υπολογιστής SISD είναι μια μηχανή ενός επεξεργαστή που εκτελεί ακολουθιακά τις εντολές μία προς μία πάνω σε ένα ρεύμα δεδομένων.

- Μοναδική Εντολή Πολλαπλά Δεδομένα (Single Instruction Multiple Data - SIMD) : Ένας υπολογιστής SIMD είναι μια μηχανή από επεξεργαστές που εκτελούν μια κοινή ακολουθία εντολών πάνω σε διαφορετικά ρεύματα δεδομένων.

- Πολλαπλές Εντολές Μοναδικό Δεδομένο (Multiple Instruction Single Data - MISD): Ένας υπολογιστής MISD είναι μια μηχανή από επεξεργαστές που εκτελούν διαφορετικές εντολές πάνω στο ίδιο ρεύμα δεδομένων. Οι υπολογιστές που βασίζονται στο μοντέλο MISD δεν είναι χρήσιμοι σε πολλές εφαρμογές.

- Πολλαπλές Εντολές Πολλαπλά Δεδομένα (Multiple Instruction Multiple Data - MIMD): Ένας υπολογιστής MIMD είναι μια μηχανή από επεξεργαστές που εκτελούν πολλαπλές εντολές πάνω σε πολλαπλά ρεύματα δεδομένων. Οι επεξεργαστές του συστήματος αυτού λειτουργούν αυτόνομα αφού έχουν την δική τους μονάδα ελέγχου και τοπική μνήμη. Οι υπολογιστές της κατηγορίας αυτής είναι κατάλληλοι για την επίλυση πολλών εφαρμογών. Οι επεξεργαστές ενός συστήματος MIMD λειτουργούν ασύγχρονα σε αντίθεση με τους υπολογιστές SIMD. Όσο πιο ασύγχρονη και χαλαρή είναι η σύνδεση ενός MIMD

συστήματος τόσο μπορούμε να μιλάμε για κατανεμημένη επεξεργασία.

Οι υπολογιστές MIMD μπορούν επίσης να ταξινομηθούν με βάση την οργάνωση και διαχείριση της μνήμης τους σε δύο μεγάλες κατηγορίες:

1. Υπολογιστές MIMD Διαμοιραζόμενης Μνήμης (Shared Memory MIMD Machine): Στους υπολογιστές διαμοιράζαμενης μνήμης, που συχνά ονομάζονται και πολυεπεξεργαστές (multiprocessors) όλοι οι επεξεργαστές έχουν πρόσβαση σε διαμοιραζόμενη μνήμη μέσω της οποίας γίνεται και η επικοινωνία μεταξύ τους. Τα δεδομένα αποθηκεύονται στην διαμοιραζόμενη μνήμη και η τροποποίηση των δεδομένων αυτών από ένα επεξεργαστή φαίνονται και στους υπόλοιπους επεξεργαστές. Παραδείγματα διαμοιραζόμενης μνήμης είναι οι διάφοροι τύποι υπολογιστών Sun, SGI, SMP (Symmetric Multi-Processing), Cray Y-MP, Cray-4, κλπ.

2. Υπολογιστές MIMD Κατανεμημένης Μνήμης (Distributed Memory MIMD Machine): Στους υπολογιστές κατανεμημένης μνήμης, που συχνά ονομάζονται και πολυυπολογιστές (multicomputers) όλοι οι επεξεργαστές έχουν την δική τους τοπική μνήμη και η επικοινωνία μεταξύ των επεξεργαστών γίνεται με την ανταλλαγή μηνυμάτων μέσω του δικτύου διασύνδεσης (interconnection network) . Το δίκτυο διασύνδεσης των υπολογιστών μπορεί να οργανωθεί σε διάφορες τοπολογίες όπως δένδρο (tree) , δακτύλιο (ring) , πλέγμα (grid) και τόρος (torus).

Περισσότερες λεπτομέρειες για τα χαρακτηριστικά των παράλληλων υπολογιστών που αναφέραμε προηγουμένως παρουσιάζονται στα βιβλία [213, 171].

2.2 Υπολογισμοί Χαμηλού Κόστους και Κίνητρα

Η χρήση των παράλληλων και κατανεμημένων συστημάτων σαν μέσο για την παροχή υπολογισμών υψηλής απόδοσης για μεγάλα μεγέθη εφαρμογών έχουν ερευνηθεί εκτενώς στα τελευταία χρόνια. Όμως μέχρι πρόσφατα, τα οφέλη από αυτήν την έρευνα περιορίστηκαν στα άτομα τα οποία είχαν πρόσβαση σε μεγάλες και ακριβές υπολογιστικές πλατφόρμες. Σήμερα η τάση στον παράλληλο και κατανεμημένο υπολογισμό μετακινείται από τις ειδικές υπερυπολογιστικες πλατφόρμες σε φτηνά συστήματα γενικού σκοπού όπως η συστοιχία υπολογιστών που αποτελείται από επεξεργαστές ή σταθμούς εργασίας ή πολυεπεξεργαστές. Επιπλέον, ακόμη και στη περίπτωση των πολυεπεξεργαστών ή άλλων αρχιτεκτονικών βελτιώσεων της απόδοσης των υπολογιστών, η τάση βρίσκεται στο σχεδιασμό συνδεδεμένων συνεπεξεργαστών (attached co-processor) που προστίθενται στα παραδοσιακά συστήματα με κάρτες επέκτασης. Αυτή η μετακίνηση οφείλεται κυρίως στις πρόσφατες εξελίξεις στα δίκτυα υψηλής ταχύτητας και την βελτιωμένη απόδοση των μικροεπεξεργαστών και των σταθμών εργασίας. Αυτές οι εξελίξεις σημαίνουν ότι η συστοιχία υπολογιστών γίνεται ένα αποτελεσματικό μέσο για παράλληλο και κατανεμημένο υπολογισμό. Ένας ακόμη σημαντικός παράγοντας είναι η προτυποποίηση πολλών εργαλείων που χρησιμοποιούνται από τις παράλληλες εφαρμογές. Παραδείγματα τέτοιων προτύπων εργαλείων είναι η βιβλιοθήκη περάσματος μηνυμάτων MPI (Message Passing Interface) [89, 90] και η γλώσσα παραλληλισμού δεδομένων (data-parallel) HPF (High Performance Fortran) [105]. Η προτυποποίηση καθιστά δυνατή την ανάπτυξη, δοκιμή και εκτέλεση των εφαρμογών σε μια συστοιχία υπολογιστών και στο τελευταίο στάδιο τη μεταφορά τους με μικρή

τροποποίηση πάνω στις αποκλειστικές παράλληλες πλατφόρμες. Η ακόλουθη λίστα υπογραμμίζει μερικούς από τους λόγους που η συστοιχία υπολογιστών χρησιμοποιείται περισσότερο σε σχέση με τους ειδικούς παράλληλους υπολογιστές [187]:

- Οι σταθμοί εργασίας γίνονται ολοένα και ταχύτεροι. Η απόδοση του σταθμού εργασίας έχει αυξηθεί δραματικά στα τελευταία χρόνια και διπλασιάζεται κάθε 18 έως 24 μήνες. Αυτό φαίνεται από το γεγονός ότι στην αγορά κυκλοφορούν ταχύτεροι επεξεργαστές και αποτελεσματικοί πολυεπεξεργαστές (SMP).

- Το εύρος ζώνης των επικοινωνιών (communication bandwidth) ανάμεσα στους σταθμούς εργασίας αυξάνεται και η καθυστέρηση (latency) μειώνεται καθώς καινούργιες τεχνολογίες δικτύων και πρωτοκόλλων υλοποιούνται στα τοπικά δίκτυα.

- Οι συστοιχίες σταθμών εργασίας είναι πιο εύκολο να ολοκληρωθούν στα υπάρχοντα δίκτυα από ότι οι ειδικοί παράλληλοι υπολογιστές.

- Οι προσωπικοί σταθμοί εργασίας δεν χρησιμοποιούνται τόσο αποδοτικά από τους χρήστες, άρα υπάρχει διαθέσιμη λανθάνουσα ισχύς.

- Η ανάπτυξη εργαλείων για τους σταθμούς εργασίας είναι πιο ώριμη σε σχέση με τους ειδικούς παράλληλους υπολογιστές και αυτό οφείλεται κυρίως στην έλλειψη προτύπων που υπάρχουν στα περισσότερα παράλληλα συστήματα.

- Οι συστοιχίες σταθμών εργασίας είναι φτηνές και εύκολα διαθέσιμες σε σχέση με τις ειδικές παράλληλες υπολογιστικές πλατφόρμες.

- Οι συστοιχίες μπορούν εύκολα να επεκταθούν και η χωρητικότητα του κάθε κόμβου (node) μπορεί εύκολα να αυξηθεί, εγκαθιστώντας επιπλέον μνήμη ή επεξεργαστές.

Σε βασικό επίπεδο, μια συστοιχία είναι μια συλλογή από σταθμούς εργασίας ή υπολογιστές που είναι διασυνδεδεμένοι μέσω δικτύου. Η συστοιχία σταθμών εργασίας είναι κατάλληλη για εφαρμογές που δεν έχουν υψηλές απαιτήσεις επικοινωνίας αφού ένα τοπικό δίκτυο έχει υψηλές καθυστερήσεις και χαμηλό εύρος ζώνης επικοινωνίας. Αν όμως οι εφαρμογές έχουν υψηλές απαιτήσεις επικοινωνίας, τότε οι συστοιχίες πρέπει να αποτελούνται από σταθμούς εργασίας υψηλής απόδοσης και το δίκτυο να είναι με υψηλό εύρος ζώνης και να έχει χαμηλή καθυστέρηση. Μια τέτοια συστοιχία μπορεί να παρέχει γρήγορες και αξιόπιστες υπηρεσίες στις υπολογιστικά απαιτητικές εφαρμογές ακόμα και σε εφαρμογές με υψηλές απαιτήσεις επικοινωνίας.

2.3 Αρχιτεκτονική μιας Συστοιχίας Σταθμών Εργασίας

Μια συστοιχία ή ένα δίκτυο σταθμών εργασίας είναι ένας είδος παράλληλου και κατανεμημένου συστήματος η οποία αποτελείται από μια συλλογή διασυνδεδεμένων κόμβων υπολογιστών που δουλεύουν μαζί σαν μια ολοκληρωμένη πηγή υπολογισμού.

Ένας κόμβος υπολογιστής (computer node) μπορεί να είναι μια μηχανή (δηλαδή, προσωπικός υπολογιστής ή σταθμός εργασίας) ή ένα σύστημα πολυεπεξεργασίας (δηλαδή, SMP) με μνήμη, με λειτουργίες Εισόδου/Εξόδου (Ε/Ε) και με ένα λειτουργικό σύστημα. Γενικά, ο όρος

συστοιχία σημαίνει ότι συνδέονται δύο ή περισσότεροι υπολογιστές (ή κόμβοι) μαζί. Οι κόμβοι μπορεί να βρίσκονται σε ένα χώρο ή να είναι ξεχωριστά και να συνδέονται μέσω τοπικού δικτύου (Local Area Network) . Μια διασυνδεδεμένη συστοιχία υπολογιστών εμφανίζεται σαν ένα μοναδικό και ενιαίο σύστημα στους χρήστες και στις εφαρμογές. Τα βασικά συστατικά μιας συστοιχίας υπολογιστών είναι τα εξής:

1. Πολλαπλοί υπολογιστές υψηλής απόδοσης.
2. Σύγχρονα λειτουργικά συστήματα (όπως πυρήνας (micro-kernel))
3. Δίκτυα ή Διακόπτες υψηλής απόδοσης (όπως Fast Ethernet, Gigabit Ethernet)
4. Κάρτες διεπαφής δικτύου (Network Interface Cards - NICs)
5. Πρωτόκολλα επικοινωνίας και υπηρεσιών (όπως Active και Fast Messages)
6. Υποδομή λογισμικού της συστοιχίας (Cluster Middleware)
7. Περιβάλλοντα και εργαλεία παράλληλου προγραμματισμού (όπως μεταγλωτιστές, PVM (Parallel Virtual Machine) και MPI (Message Passing Interface) .

Η κάρτα διεπαφής δικτύου λειτουργεί σαν επεξεργαστής επικοινωνίας και είναι υπεύθυνη για τη μετάδοση και την λήψη πακέτων δεδομένων ανάμεσα στους κόμβους της συστοιχίας μέσω δικτύου.

Το λογισμικό επικοινωνίας προσφέρει μια γρήγορη και αξιόπιστη επικοινωνία μεταξύ των κόμβων της συστοιχίας. Συνήθως, η συστοιχία με ένα δίκτυο όπως το δίκτυο Myrinet χρησιμοποιεί πρωτόκολλα επικοινωνίας όπως τα active messages για ταχύτερη επικοινωνία μεταξύ των κόμβων της συστοιχίας.

Η υποδομή λογισμικού της συστοιχίας προσφέρει την δυνατότητα η ίδια συστοιχία να φαίνεται σαν ένα μοναδικό και ενιαίο παράλληλο σύστημα. Πρόκειται για ειδικό λογισμικό που επιτρέπει την ενιαία εγκατάσταση λογισμικού, διαχείριση υλικού και αποθηκευτικού χώρου, διαχείριση των διαφόρων πόρων κατά την εκτέλεση, όπως για παράδειγμα NPACI Rocks, Globus, κλπ. Τέλος, τα περιβάλλοντα παράλληλου προγραμματισμού προσφέρουν μεταφέρσιμα και αποτελεσματικά εργαλεία για την ανάπτυξη των εφαρμογών. Τέτοια εργαλεία είναι συνήθως βιβλιοθήκες περάσματος μηνυμάτων (message passing libraries) , αποσφαλματωτές (debuggers) και προφίλ απόδοσης παράλληλων προγραμμάτων (profilers) .

2.4 Ταξινομήσεις Συστοιχιών

Οι συστοιχίες ταξινομούνται σε πολλές κατηγορίες σύμφωνα με τα παρακάτω κριτήρια [21]:

- Ιδιοκτησία του κόμβου (node ownership) . Με βάση το κριτήριο αυτό οι συστοιχίες διακρίνονται σε δύο κατηγορίες: αποκλειστική συστοιχία (dedicated cluster) και μη αποκλειστική συστοιχία (nondedicated cluster). Η αποκλειστική συστοιχία αναφέρεται στο γεγονός ότι κάθε σταθμός εργασίας εκτελεί αποκλειστικά εργασίες ενός παράλληλου υπολογισμού, ενώ η μη αποκλειστική συστοιχία αναφέρεται στο ότι κάθε σταθμός εργασίας εκτελεί τις κανονικές του ρουτίνες και χρησιμοποιεί μόνους τους αδρανείς κύκλους της κεντρικής μονάδας επεξεργασίας CPU για να εκτελέσει παράλληλες εργασίες.

- Υλικό του κόμβου (node hardware) . Οι συστοιχίες μπορεί να αποτελούνται από προσωπικούς υπολογιστές ή από σταθμούς εργασίας ή από μηχανές πολυεπεξεργασίας όπως για παράδειγμα SMPs ή ακόμη και από επιμέρους τοπικά δίκτυα πλέγματος (grid network) .

- Είδος του κόμβου (node configuration). Το κριτήριο αυτό αφορά στην αρχιτεκτονική του κάθε κόμβου της συστοιχίας. Έτσι έχουμε τις ομοιογενείς συστοιχίες (homogeneous clusters) και τις ετερογενείς συστοιχίες (heterogeneous clusters) . Οι ομοιογενείς συστοιχίες έχουν κόμβους με παρόμοιες αρχιτεκτονικές (δηλαδή, όλοι οι κόμβοι έχουν τις ίδιες ταχύτητες επεξεργασίας), ενώ οι ετερογενείς συστοιχίες έχουν κόμβους που έχουν διαφορετικές αρχιτεκτονικές (δηλαδή, όλοι οι κόμβοι έχουν διαφορετικές ταχύτητες επεξεργασίας).

2.5 Βιβλιοθήκη MPI

Το MPI είναι η πιο διαδεδομένη πρότυπη βιβλιοθήκη για πέρασμα μηνυμάτων που μπορεί να χρησιμοποιηθεί για την ανάπτυξη μεταφέρσιμων προγραμμάτων περάσματος μηνυμάτων χρησιμοποιώντας τις γλώσσες προγραμματισμού C ή Fortran. Το πρότυπο MPI ορίζει συντακτικά και σημασιολογικά ένα σύνολο από συναρτήσεις βιβλιοθήκης που είναι χρήσιμες για την σύνταξη προγραμμάτων περάσματος μηνυμάτων. Το MPI αναπτύχθηκε από μια κοινότητα ερευνητών οι οποίοι προέρχονται από πανεπιστήμια και βιομηχανίες και υποστηρίζεται από όλους τους προμηθευτές υλικού. Επίσης, το MPI είναι διαθέσιμο σε περισσότερα παράλληλα υπολογιστικά συστήματα (από υπολογιστές διαμοιραζόμενης και κατανεμημένης μνήμης μέχρι και συστοιχίες υπολογιστών). Στην επόμενη ενότητα παρουσιάσουμε σύντομα την βιβλιοθήκη MPI την οποία χρησιμοποιούμε σε αυτήν την διατριβή. Περισσότερες πληροφορίες σχετικά με την βιβλιοθήκη MPI μπορούν να βρεθούν στα βιβλία [100, 90]. Η βιβλιοθήκη MPI διαθέτει πάνω από 125 ρουτίνες αλλά ο αριθμός των βασικών εννοιών είναι πολύ μικρότερος. Μπορούν να αναπτυχθούν

Συναρτήσεις MPI	Σημασία
MPI_Init	Εκκίνηση του MPI
MPI_Finalize	Τερματισμός του MPI
MPI_Comm_size	Προσδιορίζει τον αριθμό των διεργασιών
MPI_Comm_rank	Προσδιορίζει την σειρά της καλούσας διεργασίας
MPI_Send	Στέλνει ένα μήνυμα
MPI_Recv	Λαμβάνει ένα μήνυμα

Πίνακας 2.1: Ένα σύνολο από συναρτήσεις MPI

πλήρη προγράμματα περάσματος μηνυμάτων χρησιμοποιώντας μόνο έξι ρουτίνες όπως φαίνεται στον Πίνακα 2.1.

Στις επόμενες υποενότητες περιγράφουμε τις παραπάνω συναρτήσεις και τις βασικές έννοιες που είναι απαραίτητες για την σύνταξη σωστών και αποτελεσματικών προγραμμάτων περάσματος μηνυμάτων.

2.5.1 Εκκίνηση και Τερματισμός της βιβλιοθήκης MPI

Κάθε πρόγραμμα MPI πρέπει να περιέχει το αρχείο επικεφαλίδα *include"mpi.h"*. Το αρχείο mpi.h περιέχει ορισμούς και δηλώσεις που είναι απαραίτητες για την μεταγλώττιση ενός προγράμματος MPI. Το MPI χρησιμοποιεί σταθερό σχήμα για την ονομασία των ρουτινών του MPI. Όλες οι ρουτίνες του MPI αρχίζουν με το πρόθεμα MPI_ και τον επόμενο πρώτο χαρακτήρα κεφαλαίο γράμμα. Οι υπόλοιποι χαρακτήρες των περισσότερων τύπων δεδομένων του MPI γράφονται με κεφαλαία γράμματα. Η ρουτίνα MPI_Init καλείται πριν οποιαδήποτε κλήση ρουτινών MPI. Ο σκοπός της ρουτίνας αυτής είναι να εκκινήσει το περιβάλλον MPI. Η κλήση της ρουτίνας MPI_Init περισσότερο από μια φορά κατά την διάρκεια εκτέλεσης ενός

προγράμματος καταλήγει σε σφάλμα. Η ρουτίνα MPI_Finalize καλείται στο τέλος του υπολογισμού και εκτελεί διάφορες εργασίες καθαρισμού για να τερματίσει το περιβάλλον MPI. Δεν εκτελούνται κλήσεις MPI πριν την κλήση της ρουτίνας MPI_Init και μετά την κλήση της ρουτίνας MPI_Finalize. Οι ρουτίνες MPI_Init και MPI_Finalize πρέπει να καλούνται από όλες τις διεργασίες διαφορετικά η συμπεριφορά του MPI θα είναι απροσδιόριστη. Η σύνταξη των δύο παραπάνω ρουτινών για την *C* είναι ως εξής:

int MPI_Init(int *argc, char ***argv)

int MPI_Finalize()

Τα ορίσματα argc και argv της ρουτίνας MPI_Init είναι ορίσματα γραμμής - διαταγής του προγράμματος C. Η επιτυχής εκτέλεση των ρουτινών MPI_Init και MPI_Finalize επιστρέφει την σταθερά MPI_SUCCESS διαφορετικά ένα προκαθορισμένο αριθμό σφάλματος. Συνεπώς, ένα πρόγραμμα MPI έχει την ακόλουθη τυπική δομή:

include "mpi.h"

main(int argc, char *argv[])

MPI_Init(feargc,argv);

MPI_Finalize();

2.5.2 Κανάλια Επικοινωνίας

Τα κανάλια επικοινωνίας (communicator) χρησιμοποιούνται στο MPI για επικοινωνίες από σημείο σε σημείο (point-to-point) και συλλογικές (collective). Ένα κανάλι επικοινωνίας χρησιμοποιείται για να ορίζει ένα σύνολο από διεργασίες που μπορούν να επικοινωνούν μεταξύ τους. Το σύνολο των διεργασιών σχηματίζει μια περιοχή επικοινωνίας (communication domain). Πληροφορίες σχετικά με τις περιοχές επικοινωνίας αποθηκεύονται στις μεταβλητές του τύπου MPI_Comm. Στο MPI υπάρχει ένα εξ ορισμού κανάλι επικοινωνίας που ονομάζεται MPI_COMM_WORLD το οποίο περιλαμβάνει όλες τις διεργασίες που υπάρχουν σε μια παράλληλη εφαρμογή. Όμως, υπάρχουν περιπτώσεις που θέλουμε να πραγματοποιήσουμε επικοινωνία μόνο σε μια ορισμένη ομάδα διεργασιών, δηλαδή να ορίσουμε δικά μας κανάλια επικοινωνίας. Προς το παρόν, παρακάτω χρησιμοποιούμε το MPI_COMM_WORLD σαν όρισμα κανάλι επικοινωνίας σε όλες τις συναρτήσεις MPI που απαιτούν ένα κανάλι επικοινωνίας.

2.5.3 Πλήθος Διεργασιών και Σειρά Διεργασίας

Οι συναρτήσεις MPI_Comm_size και MPI_Comm_rank χρησιμοποιούνται για να προσδιορίζουν τον αριθμό των διεργασιών και την σειρά της καλούσας διεργασίας αντίστοιχα. Η σύνταξη των δύο παραπάνω συναρτήσεων είναι ως εξής:

int MPI_Comm_size(MPI_Comm comm, int *size)

int MPI_Comm_rank(MPI_Comm comm, int *rank)

Η συνάρτηση MPI_Comm_size επιστρέφει στην μεταβλητή size τον αριθμό των διεργασιών που ανήκουν στο κανάλι επικοινωνίας comm. Κάδε διεργασία που ανήκει σε ένα κανάλι επικοινωνίας

Τύπος δεδομένων του MPI	Τύπος δεδομένων της C
MPI_CHAR	signed char I
MPI_SHORT	signed short int
MPI_INT	signed int
MPI_LONG	signed long int
$MPI\_UNSIGNED\_CHAR$	unsigned char
$MPI\_UNSIGNED$	unsigned short int
$MPI\_UNSIGNED\_LONG$	unsigned long int
MPI_FLOAT	float
MPI_DOUBLE	double
MPI_LONG_DOUBLE	long double
MPI_BYTE	-
MPI_PACKED	-

Πίνακας 2.2: Αντιστοίχιση ανάμεσα στους τόπους δεδομένων που υποστηρίζει το MPI και τους τόπους δεδομένων που υποστηρίζει η C

αναγνωρίζεται μοναδικά από την σειρά της rank. Η σειρά μιας διεργασίας είναι ένας ακέραιος αριθμός που κυμαίνεται από 0 μέχρι n, όταν υπάρχουν n διεργασίες. Μια διεργασία μπορεί να προσδιορίζει την σειρά της σε ένα κανάλι επικοινωνίας από την κλήση της συνάρτησης MPI_Comm_rank όπου παίρνει δύο ορίσματα: το κανάλι επικοινωνίας και μια ακέραια μεταβλητή rank. Με την κλήση της συνάρτησης αυτής επιστρέφεται στη μεταβλητή rank τη σειρά της διεργασίας.

2.5.4 Επικοινωνία από Σημείο σε Σημείο

Η επικοινωνία από σημείο σε σημείο περιλαμβάνει πάντα δύο διεργασίες. Η μια διεργασία στέλνει ένα μήνυμα στην άλλη διεργασία. Γνωρίζουμε ότι υπάρχουν διάφορες εκδόσεις για την αποστολή και λήψη μηνυμάτων. Έτσι, παρακάτω περιγράφουμε τις συγχρονισμένες (blocking)

και μη συγχρονισμένες (nonblocking) συναρτήσεις για αποστολή και λήψη μηνυμάτων.

Συγχρονισμένες Συναρτήσεις

Στο MPI οι συγχρονισμένες συναρτήσεις για αποστολή ή λήψη μηνημάτων επιστρέφουν τον έλεγχο τους όταν η επικοινωνία ολοκληρωθεί. Οι συγχρονισμένες συναρτήσεις για αποστολή και λήψη μηνυμάτων στο MPI είναι οι MPI_Send και MPI_Recv, αντίστοιχα. Η συγχρονισμένη συνάρτηση MPI_Send επιστρέφει όταν το μήνυμα έχει σταλεί στον παραλήπτη. Η σύνταξη της συνάρτησης MPI_Send έχει ως εξής:

int MPI_Send(void *buf, int count, MPI_Datatype datatype, int dest, int tag, MPI_Comm comm) όπου

- buf είναι η διεύθυνση του ενδιάμεσου χώρου αποθήκευσης που περιέχει δεδομένα που πρόκειται να σταλούν.
- count είναι ο αριθμός των στοιχείων του τύπου δεδομένων που ορίζεται από την παράμετρο datatype που περιέχονται στον ενδιάμεσο χώρο αποθήκευσης buf.
- datatype είναι ο τύπος δεδομένων του MPI για το κάθε στοιχείο του buf. Στον Πίνακα 2.2 παρουσιάζεται η αντιστοίχηση ανάμεσα στους τύπους δεδομένων που υποστηρίζει το MPI και τους τύπους δεδομένων που υποστηρίζει η γλώσσα προγραμματισμού C. Σημειώνουμε ότι στο MPI υπάρχουν δύο τύποι δεδομένων που δεν υποστηρίζει C. Οι τύποι αυτοί είναι οι MPI_BYTE και MPI_PACKED. Ο τύπος MPI_BYTE αντιστοιχεί σε ένα byte (8 bits). Ο τύπος MPI_PACKED αντιστοιχεί σε μια συλλογή δεδομένων που έχει δημιουργηθεί με συνένωση.
- dest είναι η διεργασία παραλήπτη που θα λάβει το μήνυμα. Η παράμετρος αυτή είναι η

σειρά της διεργασίας παραλήπτη στην περιοχή επικοινωνίας που ορίζεται από το κανάλι επικοινωνίας comm.

- tag είναι ετικέτα που έχει μια ακέραια τιμή και χρησιμοποιείται για να διακρίνει τους διαφορετικούς τύπους μηνυμάτων. Η ετικέτα tag παίρνει τιμές από 0 μέχρι MPI_TAG_UB που είναι μια προκαθορισμένη σταθερά του MPI. Η τιμή της σταθεράς MPI_TAG_UB είναι τουλάχιστον 32,767.

- comm είναι το κανάλι επικοινωνίας που διαμοιράζεται από τις διεργασίες αποστολέα και παραλήπτη.

Η συγχρονισμένη συνάρτηση MPI_Recv επιστρέφει όταν έχει λάβει το μήνυμα και έχει αντιγραφεί το μήνυμα στο ενδιάμεσο χώρο αποθήκευσης του παραλήπτη. Η σύνταξη της συνάρτησης MPI_Recv έχει ως εξής: int MPI_Recv(void *buf, int count, MPI_Datatype datatype, int source, int tag, MPI_Comm comm, MPI_Status *status)

όπου

- buf είναι η διεύθυνση του ενδιάμεσου χώρου αποθήκευσης που περιέχει δεδομένα μετά την λήψη του μηνύματος.

- count είναι ο μέγιστος αριθμός των στοιχείων του τύπου δεδομένων που ορίζεται από την παράμετρο datatype που μπορεί να περιέχονται στον ενδιάμεσο χώρο αποθήκευσης buf. Ο αριθμός των δεδομένων που πραγματικά λαμβάνει πρέπει να είναι λιγότερο από τον αριθμό count. Αν όμως ο αριθμός των δεδομένων που λαμβάνει είναι μεγαλύτερος από το μέγεθος του ενδιάμεσου χώρου (count) τότε εμφανίζεται σφάλμα υπερχείλισης και η συνάρτηση επιστρέφει το σφάλμα MPI_ERR_TRUNCATE.

- datatype είναι ο τύπος δεδομένων του MPI για το μήνυμα. Ο τύπος αυτός πρέπει να ταυτίζεται με τον τύπο δεδομένων που ορίζεται στην συνάρτηση MPI_Send.

- source είναι η σειρά της διεργασίας αποστολέα του μηνύματος στην περιοχή επικοινωνίας που ορίζεται από το κανάλι επικοινωνίας comm. Στην περίπτωση που η παράμετρος

- source έχει οριστεί με τιμή MPI_ANY_SOURCE τότε λαμβάνει μηνύματα από οποιοδήποτε αποστολέα.

- tag είναι ετικέτα που δηλώνει τι είδους μηνύματα μπορεί να λάβει η διεργασία. Αν η ετικέτα αυτή έχει τιμή MPI_ANY_TAG τότε λαμβάνει μηνύματα με οποιαδήποτε τιμή ετικέτας.

- comm είναι το κανάλι επικοινωνίας που διαμοιράζεται από τις διεργασίες αποστολέα και παραλήπτη.

- status περιέχει πληροφορίες σχετικά με την λήψη του μηνύματος.

Μη Συγχρονισμένες Συναρτήσεις

Μια μη συγχρονισμένη συνάρτηση επιστρέφει αμέσως και συνεχίζει να εκτελεί την επόμενη εντολή του προγράμματος. Σε κάποια μεταγενέστερη χρονική στιγμή η συνάρτηση ελέγχει για την ολοκλήρωση της επικοινωνίας. Οι μη συγχρονισμένες συναρτήσεις για αποστολή και λήψη μηνυμάτων στο MPI είναι οι MPI_Isend και MPI_Irecv, αντίστοιχα. Η συνάρτηση MPI_Isend επιστρέφει ανεξάρτητα με την ολοκλήρωση της επικοινωνίας. Ενώ η συνάρτηση MPI_Irecv επιστρέφει ακόμα και αν δεν υπάρχει μήνυμα να λάβει. Η σύνταξη των δύο παραπάνω συναρτήσεων είναι ως εξής:

int MPI_Isend(void *buf, int count, MPI_Datatype datatype, int dest, int tag, MPI_Comm comm, MPI_Request *request)

int MPI_Irecv(void *buf, int count, MPI_Datatype datatype, int source, int tag, MPI_Comm comm, MPI_Request_request)

Παρατηρούμε από τις παραπάνω συναρτήσεις ότι έχουν παρόμοια ορίσματα με τις συγχρονισμένες συναρτήσεις εκτός από ένα όρισμα. Το όρισμα αυτό είναι το request που χρησιμοποιείται για τον έλεγχο ολοκλήρωσης της επικοινωνίας. Οι δύο παραπάνω συναρτήσεις μπορούν να ελέγξουν για το αν ολοκληρώθηκε η επικοινωνία με την χρήση των συναρτήσεων MPI_Wait και MPI_Test. Η συνάρτηση MPI_Wait περιμένει μέχρι να ολοκληρωθεί η επικοινωνία. Η συνάρτηση MPI_Test εξετάζει αν η επικοινωνία εκείνη τη στιγμή που καλείται η συνάρτηση έχει ολοκληρωθεί και επιστρέφει αμέσως με μια λογική απάντηση TRUE ή FALSE. Δεν παρουσιάζουμε την σύνταξη των συναρτήσεων αυτών διότι δεν τις χρησιμοποιούμε συχνά σε αυτή την διατριβή.

2.5.5 Συλλογική Επικοινωνία

Η συλλογική επικοινωνία περιλαμβάνει ένα σύνολο από διεργασίες σε σχέση με την επικοινωνία από σημείο σε σημείο που περιλαμβάνει μόνο δύο διεργασίες. Το σύνολο των διεργασιών που ορίζει το κανάλι επικοινωνίας συμμετέχουν στην συλλογική επικοινωνία. Η συλλογική επικοινωνία μπορεί να υλοποιηθεί από τον προγραμματιστή χρησιμοποιώντας τις συναρτήσεις MPI_Send και MPI_Recv. Όμως, η βιβλιοθήκη MPI παρέχει ένα σύνολο από ειδικές συναρτήσεις συλλογικής επικοινωνίας. Οι συναρτήσεις είναι οι παρακάτω:

Φράγμα (Barrier)

Η συνάρτηση για το φράγμα είναι η MPI_Barrier. Η συνάρτηση αυτή συγχρονίζει την καλούσα διεργασία με όλες τις διεργασίες που πρέπει να καλέσουν την συνάρτηση αυτή. Με άλλα λόγια συγχρονίζει όλες τις διεργασίες. Η σύνταξη της είναι ως εξής:

int MPI_Barrier (MPI_Comm comm)

Εκπομπή (Broadcast)

Η συνάρτηση για την εκπομπή είναι η MPI_Bcast. Η συνάρτηση αυτή στέλνει δεδομένα από μια διεργασία ρίζα προς όλες τις διεργασίες που μετέχουν στο κανάλι επικοινωνίας. Η σύνταξη της παραπάνω συνάρτησης έχει ως εξής:

int MPI_Bcast(void *buf, int count, MPI_Datatype datatype, int root, MPI_Comm comm)

όπου τα ορίσματα buf , count και datatype είναι παρόμοια με τα ορίσματα των συναρτήσεων MPI_Send και MPI_Recv. Το όρισμα root είναι η σειρά της διεργασίας ρίζας.

Συλλογή (Gather)

Η συνάρτηση για την συλλογή είναι η MPI_Gather. Με τη συνάρτηση αυτή συγκεντρώνονται δεδομένα από όλες τις διεργασίες που μετέχουν στο κανάλι επικοινωνίας στην διεργασία ρίζα. Η σύνταξη της συνάρτησης αυτής είναι ως εξής:

int MPI_Gather(void *sendbuf, int sendcount, MPI_Datatype sendtype, void *recvbuf, int recvcount, MPI_Datatype recvtype, int root, MPI_Comm comm)

όπου

- sendbuf είναι η διεύθυνση του ενδιάμεσου χώρου αποθήκευσης των δεδομένων της κάθε διεργασίας.

- sendcount είναι ο αριθμός των στοιχείων του τύπου δεδομένων που ορίζεται από την παράμετρο sendtype που περιέχονται στο sendbuf της κάθε διεργασίας.
- sendtype είναι ο τύπος δεδομένων του MPI για το κάθε στοιχείο του sendbuf της κάθε διεργασίας.
- recvbuf είναι η διεύθυνση του ενδιάμεσου χώρου αποθήκευσης της διεργασίας ρίζας που θα περιέχει τα δεδομένα τα οποία συλλέγονται από όλες τις διεργασίες.
- recvcount είναι ο αρίθμός των στοιχείων του τύπου δεδομένων recvtype τα οποία συλλέγονται από κάθε διεργασία.
- recvtype είναι ο τύπος δεδομένων για το κάθε στοιχείο του recvbuf που έλαβε από κάθε διεργασία.
- root είναι η σειρά της διεργασίας ρίζας.
- comm είναι το κανάλι επικοινωνίας που περιέχει όλες τις διεργασίες που μετέχουν στην συλλογική επικοινωνία.

Αναγωγή (Reduction)

Η συνάρτηση για την αναγωγή είναι η MPI_Reduce. Η συνάρτηση αυτή συνδυάζει τα δεδομένα όλων των διεργασιών χρησιμοποιώντας έναν τελεστή πράξης σε μια τιμή που αποθηκεύεται στην διεργασία ρίζα. Οι πράξεις μπορεί να είναι πρόσδεση, πολλαπλασιασμός, μεγαλύτερη τιμή, μικρότερη τιμή, το λογικό AND, κλπ. Η σύνταξη της συνάρτησης είναι ως εξής:

```
int MPI_Reduce(void *sendbuf, void *recvbuf, int count, MPI_Datatype datatype,
```

Τελεστές MPI	Σημασία
MPI_MAX	Μεγαλύτερη ΤιμήΙ
MPI_MIN	Μικρότερη Τιμή
MPI_SUM	Άθροισμα
MPI_PROD	Γινόμενο
MPI_LAND	Λογικό ΚΑΙ
MPI_LOR	Λογικό Ή

Πίνακας 2.3: Αντιστοίχιση ανάμεσα στους τόπους δεδομένων που υποστηρίζει το MPI και τους τόπους δεδομένων που υποστηρίζει η C

`MPI_Op op, int root, MPI_Comm comm)`

όπου τα ορίσματα sendbuf, recvbuf , count, datatype, root και comm είναι παρόμοια με τα ορίσματα των συναρτήσεων που έχουμε παρουσιάσει μέχρι τώρα. Το όρισμα ορ είναι ο τελεστής πράξης που συνδυάζει τα δεδομένα που περιέχονται στο sendbuf από κάθε διεργασία σε μια τιμή που αποθηκεύεται στο recvbuf της διεργασίας ρίζας.

2.6 Μοντέλο Συντονιστής - Εργαζόμενος

Στη διεθνή βιβλιογραφία της παράλληλης επεξεργασίας υπάρχουν πολλά μοντέλα τα οποία χρησιμοποιούνται στον παράλληλο προγραμματισμό. Περισσότερες λεπτομέρειες για τα μοντέλα παραλληλισμού αναφέρονται διεξοδικά στα [46, 197, 130]. Επομένως, στην ενότητα αυτή παρουσιάζουμε το παράλληλο μοντέλο Συντονιστής - Εργαζομένος .

2.6.1 Μοντέλο Συντονιστής - Εργαζόμενος

Το μοντέλο συντονιστής - εργαζόμενος (Master-Worker Model) αποτελείται από δύο οντότητες: τον συντονιστή και τους πολλαπλούς εργαζόμενους. Ο συντονιστής είναι υπεύθυνος για την διάσπαση του προβλήματος σε μικρές εργασίες, για την διανομή των εργασιών αυτών στους εργαζόμενους και για την συλλογή αποτελεσμάτων από τους εργαζόμενους για να παράγει το τελικό αποτέλεσμα του προβλήματος. Οι εργαζόμενοι εκτελούν την ακόλουθη διαδικασία σε ένα απλό κύκλο χρόνου: λαμβάνουν μια εργασία, επεξεργάζονται την εργασία και αποστέλουν το αποτέλεσμα της επεξεργασίας στο συντονιστή. Έτσι, η επικοινωνία λαμβάνει χώρα μόνο ανάμεσα στον συντονιστή και στους εργαζόμενους.

Το μοντέλο συντονιστής - εργαζόμενος μπορεί να χρησιμοποιήσει είτε την στατική διανομή (static load balancing) ή την δυναμική διανομή (dynamic load balancing) . Στην πρώτη περίπτωση, η διανομή των εργασιών γίνεται στην αρχή του υπολογισμού. Απο την άλλη μεριά, η δυναμική διανομή χρησιμοποιείται όταν ο αριθμός των εργασιών είναι μεγαλύτερος από τον αριθμό των επεξεργαστών ή όταν δεν γνωρίζουμε από την αρχή της εφαρμογής τον αριθμό των εργασιών.

Το μοντέλο συντονιστής - εργαζόμενος μπορεί να γενικευτεί σε ένα ιεραρχικό μοντέλο συντονιστής - εργαζόμενος (hierarchical master-worker model). Στο ανώτερο επίπεδο του ιεραρχικού μοντέλου ο συντονιστής διανέμει μεγάλα κομμάτια εργασιών στους συντονιστές του δευτέρου επιπέδου οι οποίοι διαμερίζουν και διανέμουν τις εργασίες στους αντίστοιχους εργαζομένους τους και επίσης κάθε συντονιστής του δευτέρου επιπέδου μπορεί να εκτελέσει κάποιες εργασίες. Το ιεραρχικό μοντέλο είναι κατάλληλο σε συστήματα διαμοιραζόμενης μνήμης ή κατανεμημένης

μνήμης εφόσον η αλληλεπίδραση είναι δύο επιπέδων (two-way) , δηλαδή ο συντονιστής γνωρίζει ότι πρέπει να διανέμει εργασίες και οι εργαζόμενοι γνωρίζουν να λαμβάνουν εργασίες από τον συντονιστή.

Όταν χρησιμοποιούμε το μοντέλο συντονιστής - εργαζόμενος πρέπει να λάβουμε υπόψη ότι ο συντονιστής δεν πρέπει να υποστεί συμφόρηση (bottleneck) η οποία μπορεί να εμφανιστεί όταν οι εργασίες είναι πολύ μικρές ή όταν οι εργαζόμενοι είναι σχετικά γρήγοροι. Σε αυτό το μοντέλο, η διασπορά των εργασιών (granularity) πρέπει να είναι τέτοια ώστε το κόστος του υπολογισμού να κυριαρχείται από το κόστος της διανομής των εργασιών και το κόστος του συγχρονισμού.

2.7 Μέτρα Απόδοσης

Σε αυτή την ενότητα ορίζουμε ορισμένα κοινά μέτρα απόδοσης που αναφέρονται σε συστοιχίες από ομοιογενείς σταθμούς εργασίας.

Ο χρόνος T ενός παράλληλου αλγορίθμου ορίζεται να είναι η μέγιστη χρονική διάρκεια που μπορεί να μεσολαβήσει από την εκκίνηση του αλγορίθμου στον πρώτο σταθμό εργασίας που θα αρχίσει ως τον τερματισμό του αλγορίθμου στον τελευταίο επεξεργαστή που θα ολοκληρώσει την εκτέλεση του.

Η επιτάχυνση (speedup) S_p ενός παράλληλου αλγορίθμου που εκτελείται σε p σταθμούς εργασίας ορίζεται να είναι ο λόγος του χρόνου του ακολουθιακού αλγορίθμου (όταν εκτελείται σε έναν από τους σταθμούς εργασίας) προς το χρόνο του παράλληλου αλγορίθμου στους p σταθμούς εργασίας και έχει ιδανική τιμή p. Έτσι, η επιτάχυνση ορίζεται από τον τύπο:

$$S_p = \frac{T_1}{T_p}$$

όπου T_i και T_p είναι χρόνοι εκτέλεσης του ίδιου αλγορίθμου (που υλοποιείται για ακολουθιακή και παράλληλη εκτέλεση) σε 1 και p σταθμούς εργασίας αντίστοιχα. Η αποδοτικότητα (efficiency) E_p ενός παράλληλου αλγορίθμου που εκτελείται σε p σταθμούς εργασίας ορίζεται να είναι ο λόγος της επιτάχυνσης S_p προς το πλήθος των σταθμών εργασίας p και έχει ιδανική τιμή 1, δηλαδή:

$$E_p = \frac{S_p}{p}$$

Η αποδοτικότητα E_p ορίζεται εναλλακτικά και ως το μέτρο του γινομένου επιφάνειας επί τον χρόνο,

$$AT_p = \frac{1T_1}{pT_p} = E_p$$

Επιπλέον το γινόμενο $SpEp$ ορίζεται και ως AT^2 δηλαδή,

$$AT_p^2 = \frac{1T_1^2}{pT_p^2} = S_p E_p$$

Τέλος, η συνάρτηση κλιμάκωσης (scalability) δίνει το χρόνο εκτέλεσης του αλγορίθμου για διαφορετικά μεγέθη προβλήματος, καθώς μεταβάλλεται ο αριθμός των σταθμών εργασίας ρ ανάλογα με τη μεταβολή του μεγέθους του προβλήματος. Η συνάρτηση αυτή μετρά τη δυνατότητα του αλγορίθμου να κάνει χρήση περισσότερων σταθμών εργασίας σε περίπτωση που αυξηθεί το μέγεθος του προβλήματος.

Κεφάλαιο 3

Μηχανισμοί Μιμητικών Αλγορίθμων

Ένας ΜΑ όπως κάθε μεταευρετικός αλγόριθμος έχει γενικώς πιο πολύπλοκη δομή από ένα εξελικτικό αλγόριθμο καθώς είναι σύνθεση ευρετικών και εξελικτικών αλγορίθμων που περιέχουν στοιχεία μίμησης βιολογικών ή κοινωνικών διαδικασιών. Ανάλογα με την υλοποίηση εργάζεται με κωδικοποιημένο σύνολο λύσεων ή με τις λύσεις αυτές καθεαυτές. Στην πρώτη περίπτωση συγγενεύει πιο πολύ με τους γενετικούς αλγορίθμους στην δεύτερη με τους εξελικτικούς. Εκτελεί αναζητήσεις χρησιμοποιώντας ένα πληθυσμό λύσεων και όχι μια μοναδική λύση. Ο χρόνος εκτέλεσης του είναι εν γένει μεγαλύτερος από αυτόν κάποιου αντίστοιχου ευρετικού αλλά σε αντιστάθμισμα αναμενουμε λύσεις καλύτερης ποιότητας από αυτές που παράγει ο αντίστοιχος ευρετικός ή εξελικτικός.

Στο κεφάλαιο αυτό αναφέρουμε τα κύρια στοιχεία που χαρακτηρίζουν του μιμητικούς αλγορίθμους και στην συνέχεια παρουσιάζουμε την βασική διάταξη ενός ΜΑ (ΔΜΑ). Τη ΔΜΑ την έχουμε παρουσιάσει στο XI Congreso LatinoIberoamericano de Investigation de Operaciones τον Οκτώβριο του 2002 [79] καθώς και στο 4th Metaheuristics International conference τον Ιούλιο του 2001 [71]. Αναφέρεται και από άλλους ερευνητές της περιοχής όπως για παράδειγμα στις εργασίες [24, 5, 12]. Στο τέλος του κεφαλαίου παρατίθενται ορισμένα παραδείγματα χρήσης της ΔΜΑ.

3.1 Ταξινόμηση των Μιμητικών Αλγορίθμων

3.1.1 Κύρια χαρακτηριστικά Μιμητικών Αλγορίθμων

Πληθυσμός

Το μέγεθος του πληθυσμού που αποτελεί ένα από τα βασικά χαρακτηριστικά των ΜΑ [156, 139, 126, 45], μπορεί να θεωρηθεί σαν ένα σημαντικό κριτήριο αξιολόγησης της ικανότητας του ΜΑ για διερεύνηση του χώρου λύσεων [123, 156, 139]. Το μέγεθος πληθυσμών μπορεί να είναι σταθερό ή μεταβαλλόμενο κατά τη διάρκεια της εξέλιξης. Τα άτομα που απαρτίζουν τον πληθυσμό και ανταλλάσσουν πληροφορία σε ένα ΜΑ καλούνται γονείς ενώ τα άτομα που πρόσφατα δημιουργήθηκαν ή τροποποιήθηκαν καλούνται παιδιά. Κάθε άτομο του πληθυσμού αντιπροσωπευει ένα σημείο του χώρου των πιθανών λύσεων ενός προβλήματος.

Η ανταλλαγή των πληροφοριών πραγματοποιείται με τελεστές. Τελεστές είναι ο ανασυνδυασμός, η μετάλλαξη και οι τεχνικές τοπικής αναζήτησης [156]. Ο αριθμός των γονέων που

ανασυνδυάζονται μεταξύ τους για την δημιουργία ενός απογόνου είναι ένα σημαντικό κριτήριο για την αξιολόγηση ενός οποιοδήποτε ΕΑ και καθορίζει πόσες πληροφορίες συγχωνεύονται ταυτόχρονα [3, 156, 49, 83, 195]. Για παράδειγμα, σε ένα ΜΑ ο αριθμός γονέων είναι σταθερός και ίσος με 2, αλλά υπάρχουν και άλλοι ΜΑ στους όποιους ο αριθμός γονέων μπορεί να ποικίλει κατά τη διάρκεια της εκτέλεσης του αλγορίθμου.

Εκτός από τους γονείς, η δημιουργία του απογόνου μπορεί επίσης να απαιτεί μερικές πληροφορίες για την μέχρι τώρα εξέλιξη του πληθυσμού. Αυτές οι πληροφορίες αντιπροσωπεύουν τη γνωση για τους κανόνες με τους οποίους ο πληθυσμός εξελίσσεται, και καλούνται *ιστορικό* του πληθυσμού [110, 123, 156]. Αυτό το ιστορικό λαμβάνεται υπόψη κατά την εκτέλεση, και ενημερώνεται. Σε κάθε άτομο του πληθυσμού αντιστοιχείται ένα μετρο της ποιότητας που διαθέτει στο χώρο αναζήτησης του προβλήματος το οποίο αντιμετωπίζεται και το οποίο κωδικοποιείται μέσω κάποιας συνάρτησης. Το ιστορικό είναι η διαθέσιμη πληροφορία για την εξέλιξη η οποία αξιοποιείται κατά το στάδιο της επιλογής βάση της ποιότητας των ατόμων. Η τοπική αναζήτηση συμβάλλει στην εύρεση των καλύτερων ατόμων για ανασυνδυασμό και μετάλλαξη [110, 123, 156]. Μετά απο την εκτέλεση της τοπικής αναζήτησης το ιστορικό ενημερώνεται και έπεται ανασυνδυασμός και μετάλλαξη μέσω των οποίων διαταράσσεται η δομή των ατόμων παρέχοντας περαιτέρω δυνατότητες διερεύνησης του χώρου αναζήτησης. Έτσι οι ΜΑ δεν βασίζονται μόνο σε τυχαίους κανόνες αλλά χρησιμοποιούν και πληροφορίες που αποκτούν κατά τη διερεύνηση του εφικτού χώρου οπότε τα βήματα δεν είναι εντελώς τυχαία αλλά προσαρμόζονται στις εκάστοτε συνθήκες [177].

Η συχνότητα ανταλλαγής πληροφοριών (exchange rate) είναι επίσης ένας σημαντικός παράγοντας και καθορίζει το πόσες πληροφορίες ανταλλάσσονται κατά μέσον όρο. Για παράδειγμα,

εάν λίγες πληροφορίες ανταλλάσσονται συχνά ή εάν πολλές πληροφορίες ανταλλάσονται σπάνια μεταξύ των ατόμων του πληθυσμού, τότε κατά μέσο όρο η ανταλλαγή των πληροφοριών είναι μικρή.

Η τοπολογία των ατόμων στο πληθυσμό δηλαδή ο τρόπος με τον οποιο διατάσσονται τα άτομα μέσα στον πληθυσμό παίζει επίσης καθοριστικό ρόλο στην διερεύνηση του χώρου λύσεων. Οι πιο συνηθισμένες τοπολογίες ατόμων ειναι πλέγματος, πλήρους διασύνδεσης, δακτυλίου, σταυρού και τετραγώνου [124, 157, 52, 47, 42].

Οι πηγές πληροφορίας ενός απογόνου περιγράφονται από τρία συστατικά, τον αριθμό των γονέων, το ιστορικό εξέλιξης (που σημειώνεται ως $h_{population}$), και την συχνότητα ανταλλαγής [110, 123, 156, 3].

Κωδικοποίηση

Οι λύσεις - άτομα του πληθυσμού σε έναν μιμητικό αλγόριθμο ΜΑ δεν διατηρούνται με την κανονική τους μορφή (π.χ. σαν αριθμοί) αλλά με μιά κωδικοποιημένη μορφή έτσι ώστε να προσομοιάσουν τα χρωμοσώματα, που κρύβουν κωδικοποιημένα τα χαρακτηριστικά του ατόμου, και να μπορέσουν να εφαρμοστούν πάνω τους οι γενετικοί τελεστές (πχ. αναπαραγωγή). Έτσι κάθε λύση είναι κωδικοποιημένη με μια συμβολοσειρά.

Οι μέθοδοι αναπαράστασης ποικίλουν μεταξύ των μεχρι τώρα υλοποιηθέντων ΜΑ. Τυπικά οι ΜΑ χρησιμοποιουν δυαδικές συμβολοσειρές. Αυτή η αναπαράσταση κάνει τους ΜΑ πιο ανεξάρτητους από το πρόβλημα, αφού μετά τον ορισμό της κατάλληλης κωδικοποίησης μπορούν να χρησιμοποιηθούν οι κλασσικές μορφές των τελεστών ανασυνδυασμού και μετάλλαξης

σε επίπεδο δυαδικής συμβολοσειράς. Χρησιμοποιουνται όμως κατά περιπτώσεις και άλλα είδη αναπαράστασης όπως η κωδικοποίηση Gray και η πραγματική κωδικοποίηση (real coding) [129, 177, 39, 45, 168, 59, 153]. Βασικό μειονέκτημα της δυαδικής αναπαράστασης είναι ότι οι συμβολοσειρές που αντιστοιχούν σε δύο συνεχόμενες τιμές μπορεί να διαφέρουν τόσο πολύ, ώστε να είναι αδύνατη η μετάβαση από μια τιμή στην άλλη μέσω της διαδικασίας μετάλλαξης [182].

Η κωδικοποίηση Gray συνίσταται στη μετατροπή των ακέραιων αριθμών σε δυαδικές συμβολοσειρές έτσι ώστε κάθε ζεύγος διαδοχικών αριθμών να διαφοροποιείται ως προς ένα ψηφίο και εξομαλύνει τη δράση του μηχανισμού μετάλλαξης καθώς στις περισσότερες περιπτώσεις μεταβάλλει ελάχιστα την τιμή της παραμέτρου. Υπάρχει ωστόσο ένας μικρός αριθμός περιπτώσεων κατά τις οποίες η μετάλλαξη μεταβάλλει δραστικά την τιμή της παραμέτρου [212]. Κατά συνέπεια, ο μετασχηματισμός Gray από την μια περιορίζει την επίδραση της μετάλλαξης αλλά από την άλλη αφήνει περιθώρια μετάβασης σε εντελώς διαφορετικές λύσεις, με αλλαγή ενός ψηφίου και μόνο [140, 143].

Η πραγματική κωδικοποίηση βελτιώνει την απόδοση των ΜΑ σε εφαρμογές συνεχών μεταβλητών, στις οποίες ζητείται ο εντοπισμός της λύσης με μεγάλη ακρίβεια[134, 102]. Με τον τρόπο αυτό αποφεύγεται η χρήση υπερβολικά μεγάλων συμβολοσειρών και εξοικονομείται ο χρόνος που δαπανάται κατά τη διαδικασία κωδικοποίησης/αποκωδικοποίησης [156, 123, 183].

Για κάθε μια από αυτές τις αναπαραστάσεις έχουν δημιουργηθει ειδικοί τελεστές ανασυνδυασμού και μετάλλαξης.

Ανασυνδυασμός

Ο ανασυνδυασμός στους MAs είναι πάντα ένας τελεστής ο οποίος με πιθανότητα p_c επιλέγει δύο γονείς από τον πληθυσμό, τους ανασυνδυάζει για να δημιουργήσει δύο νέα άτομα.[141]. Ο μηχανισμός του ανασυνδυασμού δεν εφαρμόζεται συνήθως σε όλα τα ζευγάρια των ατόμων που έχουν επιλεγεί για αναπαραγωγή. Οι συνήθεις προτεινόμενες θέσεις για την πιθανότητα ανασυνδυασμού είναι $p_c = 0{,}6$ [146, 198, 154, 81, 141], $p_c =$ [0.75...0,95] [118, 214, 177, 6, 161, 198, 154, 57]. Εάν δεν εφαρμοστεί η μέθοδος αυτή, οι απόγονοι γεννιουνται με απλή αντιγραφή των γονιών. Παραδείγματα μεθόδων ανασυνδυασμού που χρησιμοποιουνται είναι ανασυνδυασμός ενός σημειου, δυο σημείων, πολλών σημείων, μετατόπισης, στίξης, δειγματων, ευρετικός ανασυνδυασμός [18, 146, 198, 154, 81].

Μετάλλαξη

Ο τελεστής μετάλλαξης στους MAς αλλάζει περιστασιακά ορισμένα bit των ατόμων αντιστρέφοντάς τα . Η πιθανότητα μετάλλαξης $p_m \in [0, 1]$ ανά bit είναι συνήθως πολύ μικρή στους MA. Προτεινομενες τιμές για το ποσοστό μετάλλαξης γενικά στους EAς θεωρούνται $p_m = 0,001$, $p_m = 0,01$ και $p_m = [0,005....0,01]$, [191, 161, 16]. Η τιμή της συχνότητας μετάλλαξης είναι αντιστρόφως ανάλογη του μεγέθους πληθυσμού. Ο μηχανισμός της μετάλλαξης βοηθά τον μιμητικό αλγόριθμο να διαφεύγει από τα τοπικά ακρότατα, παρέχοντας μια επιπλέον συνιστώσα τυχαιότητας στη διαδικασία της εξέλιξης [111, 139, 123]. Μια άλλη λειτουργία της μετάλλαξης, η οποία έχει σημασία μόνο στην περίπτωση που έχει προσδιοριστεί αρκετά η βέλτιστη λύση, είναι η δημιουργία μικρών διαταραχών κοντά στην περιοχή του ακροτάτου για την επιτάχυνση

της διαδικασίας της σύγκλισης. Γενικά πάντως η επίδραση της μετάλλαξης στην αξιοπιστία και ταχυτητα των μιμητικών αλγορίθμων αλλα και γενικότερα των εξελικτικών αλγορίθμων [18, 65] είναι περιορισμένη σε σχέση με τους υπόλοιπους μηχανισμούς [139, 78].

Συνάρτηση Γειτονιάς

Ένα σημαντικός τελεστής στους ΜΑς όπως και στους υπόλοιπους ΕΑς για την ανταλλαγή των πληροφοριών μεταξύ των ατόμων είναι το πλήθος των ατόμων που μπορούν να ανταλλαχθούν. Ο αριθμός αυτός εξαρτάται από την συνάρτηση γειτονιάς $N : P \rightarrow (P)$ Η συνάρτηση γειτονιάς ορίζει για κάθε άτομο e ένα υποσύνολο P [153, 119, 98, 140].

Ένας τελεστής που απευθύνεται σε ένα άτομο $e \in P$ μπορεί έπειτα να ενεργήσει σε ένα άλλο άτομο το οποίο λαμβάνεται από το $N(e)$. Μπορούμε να σκεφτόμαστε τη συνάρτηση γειτονιάς ως ένα μέσο μέτρησης της καταλληλότητας που επιθυμούμε να βελτιώσουμε. Η επιλογή ατόμων ανάλογα με τις τιμές της ποιότητας τους σημαίνει ότι συμβολοσειρές με μια υψηλότερη τιμή έχουν και μεγαλύτερη πιθανότητα συνεισφοράς ενός ή περισσοτέρων απογόνων στην επόμενη γενιά.

Η γειτονιά όπως έχουμε ήδη αναφέρει ορίζει το σύνολο των ατόμων που ανήκουν μέσα στα όρια ενός γεωμετρικού σχήματος με κέντρο τον αρχικό γονέα. Με την εφαρμογή της τοπικής επιλογής επιτυγχάνεται η δημιουργία γειτονιών ατόμων με κοινά χαρακτηριστικά μέσα στον πληθυσμό. Τα άτομα μπορούν να ανασυνδυάζονται με διάφορα άλλα άτομα του πληθυσμού με τοπολογία πλήρους διασύνδεσης, δακτυλίου, σταυρού ή τετραγώνου. Εκτός από την κλασική περίπτωση υπάρχουν και πιο σύνθετες περιπτώσεις ΜΑ στις οποίες τα άτομα ενός πληθυσμού

μπορεί να ειναι τοποθετημένα με μια συγκεκριμένη τοπολογία μέσα στον πληθυσμό. Παραδείγματα τοπολογιών μπορεί να δεί ο αναγνώστης στις εργασίες [124, 156, 142].

Ιστορικό εξέλιξης

Για κάθε άτομο υπάρχουν κάποιες πληροφορίες εξέλιξης που δεν αφορούν το πρόβλημα που επιλυεται αλλά το πώς το άτομο συμπεριφέρεται. Για παράδειγμα ποιο είναι το ποσοστό μεταλλαξής του σε κάποια συγκεκριμένη στιγμή εκτέλεσης του αλγορίθμου (γενιά). Τέτοιες πληροφορίες αποτελούν το ιστορικό εξέλιξης του ατόμου και αναφέρονται ως $h_{individual}$. Το ιστορικό εξέλιξης περιέχει πληροφορίες για το πως συμπεριφέρθηκε το άτομο στις προηγούμενες γενιές και με βάση αυτό το ιστορικό γίνεται με κάποια διαδικασία πχ (μοντέλο ολικής γενεαλογικής αντικατάστασης) [25, 146, 98, 128, 164] η διαμορφώση της συμπεριφοράς στην τρέχουσα γενιά [16, 18, 185, 54]. Η σπουδαιότητα της ύπαρξης στοιχείων για την πορεία που ακολουθεί ο μιμητικός αλγόριθμος προς την εύρεση της λύσης είναι αυτονόητη. Βοηθάει στην εξαγωγή συμπερασμάτων και στον έλγχο των μεθόδων που χρησιμοποιούνται. Σε γενικές γραμμές τα στοιχεία εκείνα που θα πρέπει να καταγραφτούν στο ιστορικό είναι τα :

- Η καλύτερη και η χειρότερη λύση.
- Το ποσοστό του πληθυσμού που συγκλίνει προς την τρέχουσα καλύτερη λύση.
- Η μέση τιμή της ποιότητας του πληθυσμού.
- Το πλήθος των ανασυνδυασμών και των μεταλλάξεων που έχουν γίνει στο πέρας μιας γενιάς.
- Το πλήθος των γενεών κατα τις οποίες η καλυτερη λύση παραμένει στάσιμη.

Μέθοδος Αντικατάστασης

Η εξέλιξη ενός πληθυσμού επιτυγχάνεται με μια σειρά γενεών (επαναλήψεων). Έαν έχουμε αντικατάσταση ολόκληρου του πληθυσμού από τη μια γενιά στην άλλη τότε λέμε ότι χρησιμοποιείται το μοντέλο ολικής αντικατάστασης [139, 11, 95, 43, 98, 62, 153] ενώ στις περιπτωσεις που γινεται αντικατάσταση μέρους του πληθυσμού λέμε ότι γίνεται χρήση του μοντέλου σταθερής αντικατάστασης [139, 11, 95, 43, 98, 62, 216]. Κατά την παραλληλοποίηση των MAs όπως και στους υπόλοιπους EAs, η εξέλιξη του πληθυσμού μπορεί να ειναι ασύγχρονη ή σύγχρονη [148].

Στην περίπτωση της ασύγχρονης εξέλιξης , κάθε άτομο αλλάζει συνεχώς χωρίς να ελέγχει εάν και τα υπόλοιπα άτομα αλλάζουν επίσης. Αρχικά παράγεται ένα πλήθος σημείων μέσα από τον εφικτό χώρο, το οποίο χωρίζεται ανά ομάδες, οι οποίες αναπτύσσονται ανεξάρτητα. Από κάθε ομάδα παράγεται ένας βελτιωμένος πληθυσμός. Σε τακτά διαστήματα ο συνολικός πληθυσμός χωρίζεται σε νέες ομάδες, εξασφαλίζοντας τη διάδοση των πληροφοριών που έχουν συλλεγεί. Σταδιακά, όλα τα σημεία τείνουν προς το ολικό βέλτιστο του προβλήματος, υπό την προυπόθεση ότι το μέγεθος του αρχικού πληθυσμου είναι αρκετά μεγάλο [54, 119, 19, 216].

Ελιτισμός

Συνίσταται στην διατήρηση του καλύτερου ατόμου μιας γενιάς και αντιγραφή του στην επόμενη γενιά. Αυτό εξασφαλίζει ότι μια καλή λύση που βρέθηκε δεν πρόκειται να χαθεί εξαιτίας της επίδρασης κάποιου άλλου τελεστή ή της απειροελάχιστης πιθανότητας να μην επιλεγεί για αναπαραγωγή[139]. Επίσης διασφαλίζεται ότι η καλυτερη λύση κάθε γενιάς θα είναι καλύτερη

ή ίση από αυτή της προηγούμενης γενιάς. Η διατήρηση της καλύτερης μέχρι τώρα λύσης είναι ζωτικής σημασίας για έναν ΜΑ [174, 44, 84, 146, 98, 128, 123]. Το ελιτίστικο μοντέλο αποσκοπεί στο να μην χάνεται κατά την πορεία εξέλιξης κάποια λύση η οποία βρίσκεται κοντά στη βέλτιστη του προβλήματος, εξαιτίας της τυχαιότητας των μηχανισμών επιβίωσης. Έτσι στον ελιτισμό αν η τρέχουσα βέλτιστη λύση αποκλειστεί από την επόμενη γενιά, τότε υποχρεωτικά συμπεριλαμβάνεται σε αυτή ως πρόσθετο μέλος του πληθυσμού.

Μέθοδος Τοπικής Αναζήτησης

Οι ΜΑ ξεχωρίζουν στο σύνολο των ΕΑ για την ικανότητα τους να εντοπίσουν το ολικό βέλτιστο με σχετικά μικρό αριθμό δοκιμών [52, 139, 216, 140]. Στους υπόλοιπους EAs η εύρεση λύσης κοντινής στο απόλυτο βέλτιστο είναι αργή με τους υπόλοιπους τελεστές γιατί πάντα υπάρχει ο τυχαίος παράγοντας. Με την χρήση μιας μεθόδου τοπικής αναζήτησης (πχ. Προσομοιούμενη Ανόπτηση [215, 204, 92, 219, 114], Αποτρεπτική Αναζήτηση [155, 59, 53, 97, 56] ή της κατευθυνόμενης τοπικής αναζήτησης [206, 208, 205, 111, 163]) πριν το στάδιο του ανασυνδυασμου ή της μετάλλαξης κάθως και κατα την αρχικοποίηση του πληθυσμού υποβοηθείται η συνάρτηση καταλληλότητας στην εύρεση του τοπικού βέλτιστου[153, 146, 147]. Με την χρήση δε της κατευθυνόμενης τοπικής αναζήτησης [110, 111] αποφεύγεται η προσκόλληση σε τοπικά ελάχιστα. Η επιλογή όμως της κατάλληλης μεθόδου εξαρτάται συνήθως απο τη δυσκολία των προβλημάτων που επιλύονται έτσι ώστε ο πληθυσμός να μην προσκολλά σε τοπικά βέλτιστα.

3.2 Επίπεδα Πληθυσμών - Φαινόμενο Νησίδων

Στις προηγούμενες ενότητες αναφερθήκαμε στα κύρια χαρακτηριστικά ενός πληθυσμου ατόμων. Υπάρχουν όμως και άλλα επίπεδα περιγραφής τα οποία αναφέρονται σε περισσότερους πληθυσμούς ή σε ένα σύνολο πληθυσμών. Σκοπός της παρούσας ενότητας είναι η κατανόηση του πως χρησιμοποιούνται σε άλλα επίπεδα περιγραφής περισσοτέρων του ενός πληθυσμών τα στοιχεία της βασικής ΔΜΑ και η δημιουργία μια διάταξης που θα περιγράφει ένα ΜΑ βασισμένο σε πολλά επίπεδα . Η αναγκαιότητα περισσοτέρων επιπέδων για τη διερεύνηση του χώρου λύσεων ενός προβλήματος μας οδηγεί στην παραλληλοποίηση των ΜΑ. Έτσι στις επόμενες παραγράφους έχουμε μια σύντομη περιγραφή αυτών των χαρακτηριστικών.

Συνήθως, η έννοια του ανασυνδυασμού των γονέων χρησιμοποιείται μόνο όταν δημιουργείται ένα νέο άτομο με το συνδυασμό των πληροφοριών άλλων ατόμων του ίδιου πληθυσμού. Αυτή η έννοια μπορεί να γενικευτεί σε οποιαδήποτε ανταλλαγή των πληροφοριών για οποιοδήποτε ΕΑ [25, 33, 42, 3, 127, 200, 74, 76]. Για παράδειγμα ας εξετάσουμε την περίπτωση όπου διάφοροι πληθυσμοί (ή νησιά) χρησιμοποιούνται. Εάν ένα νησί δημιουργείται κατόπιν συλλογής ατόμων από δύο νησιά I_1 και I_2 τότε μπορούμε να θεωρήσουμε το νησί ως απόγονο των γονέων (πατρικών) νησίδων I_1 και I_2.

Τα περισσότερα από τα χαρακτηριστικά της ΔΜΑ μπορούν να ορίσουν πλήρως τέτοιες περιπτώσεις με την μόνη διαφοροποίηση οτι αντικαθιστούμε την έννοια του ατόμου με την έννοια της νησίδας. Έχοντας αυτό υπόψη, η ΔΜΑ μπορεί να περιγράψει περισσότερα του ενός επίπεδα πληθυσμων όπως συμβαινει και στην περιπτωση των νησιδων. Στη νέα διάταξη που θα καθορίζει ενα ΜΑ με νησίδες (σχ. 3.1) θα πρέπει να καθορίσουμε για κάθε επίπεδο επιπλέον

Island-Based Memetic Algorithm (IMA)

1. determine k initial islands $[pop^0, ..., pop^{k-1}]$
2. generation-count:= 0
3. Local-Search(k)
4. repeat
5. generation
6. generationcount := generationcount+1
7. for each island i
8. while $pop^i_{intermediate}$ not full do
9. Local-Search(pop), Elitism(pop)
10. select indi1 and indi2 in pop^i
11. (offsp1,offsp2) crossover(indi1,indi2)
12. put offsp1 and offsp2 in $pop^i_{intermediate}$
13. mutate each individual in $pop^i_{intermediate}$
14. $pop^i := pop^i_{intermediate}$
15. if generationcount is multiple of m
16. then the best individual of each island pop^i migrates to $pop^{(i+1)modk}$
17. until termination condition is met

Σχήμα 3.1: Μιμητικός Αλγόριθμος βασισμένος σε νησίδες

το στοιχείο e και ένα σύνολο S από στοιχεία καθώς και τον χώρο (τοπολογία) στην οποία θα γίνεται η μετανάστευση, η μεταφορά δηλαδή ολόκληρου του ατόμου e απο ένα σύνολο S σε ένα άλλο.

Ένα απλό παράδειγμα είναι και οι MA σε νησίδες που τοποθετούνται σε τοπολογία δακτυλίου, και οι μεταναστεύσεις γίνονται μόνο κατά μήκος αυτού του δαχτυλίου. Κάθε φορά που εκτελείται μια νέα γενεά, ένα αντίγραφο του καλύτερου ατόμου (με την καλύτερη ποιότητα) μεταναστεύει από ένα νησί στο επόμενο νησί [42, 124]. Κάθε νησί έτσι λαμβάνει ένα νέο άτομο που αντικαθιστά ένα από τα άτομά του που επιλέγονται τυχαία (μια άλλη πολιτική θα ήταν να αντικατασταθεί το άτομο με το χαμηλότερη ποιότητα)[156]. Άλλη μια τοπολογία ίσως η πιο συνηθισμένη στους MA είναι και αυτή της πλήρους διασταύρωσης.

(1)	**Μέγεθος Πληθυσμού**
(2)	**Κωδικοποίηση**
(3)	**Ανασυνδυασμός**
(4)	**Μετάλλαξη**
(5)	**Ιστορικό Εξέλιξης** $h_{population}$
(6)	**Μέθοδος Αντικατάστασης Πληθυσμού**
(7)	**Ελιτισμός**
(8)	**Μέθοδος Τοπικής Αναζήτησης**
(9)	**Δομή Πληθυσμού**

Πίνακας 3.1: Η βασική Διάταξη ενός Μιμητικού Αλγορίθμου ΔΜΑ

3.3 Βασικές Διατάξεις ΜΑ

3.3.1 Η βασική διάταξη ενός απλού Μιμητικού Αλγορίθμου ΔΜΑ

Για να μπορέσει κάποιος να συγκρίνει την απόδοση ενός ΜΑ με ένα άλλο ΜΑ είναι σημαντικό να έχει καταγεγραμμένες όλες τις πληροφορίες που αφορούν στα βασικά χαρακτηριστικά με τέτοιο τρόπο ώστε να μπορεί ανα πάσα στιγμή να τις χειριστεί. Είναι πολύ σημαντικό το να κατασκευαστεί εξ' αρχής ένας πίνακας στον οποίο θα έχουν κωδικοποιηθεί όλα τα κριτήρια αξιολόγησης. Στην ενότητα αυτή παραθέτουμε την διάταξη για ένα απλό μιμητικό αλγόριθμο.

- **Μέγεθος Πληθυσμού** (1): Υπάρχουν δυο περιπτώσεις. Όταν το μέγεθος πληθυσμού ειναι σταθερό παίρνει μια τιμή πχ [100,0,0] για μέγεθος πληθυσμού 100 ενώ όταν είναι μεταβαλλόμενο σε κάποια κλίμακα πχ. στο διαστημα από 100 μεχρι 400 με βήμα 50 τότε παίρνει τιμη (100,400,50).

- **Κωδικοποίηση** (2): Αναλογα με το είδος της κωδικοποίησης που χρησιμοποιείται

παίρνει τιμές 0 για δυαδική κωδικοποίηση, 1 για πραγματική αναπαράσταση.

- **Ανασυνδυασμός** (3): Ανάλογα με το είδος του ανασυνδυασμού καθορίζεται η παράμετρος αυτή. Παίρνει την τιμή [0,0.5,0.7] για ανασυνδυασμό ενός σημείου με πιθανότητα $p_c = (0.5, 0.7)$, [1,0.2,0.7] για ανασυνδυασμό δύο σημείων με πιθανότητα $p_c = (0.2, 0.7)$, [2,0.6,0.75] για ανασυνδυασμό πολλών σημειων και 3 για ευρετικό ανασυνδυασμό με πιθανότητα $p_c = (0.6, 0.75)$.

- **Μετάλλαξη** (4): Παίρνει τιμή [0,0,0] εάν δεν έχουμε καθόλου μετάλλαξη ενώ αν έχουμε μετάλλαξη σε κάποια κλίμακα πχ. από 0.001 έως 0.002 παίρνει τιμή [1,0.001,0.002].

- **Ιστορικό Εξέλιξης** (5): Ο αριθμός γονέων για κάθε απόγονο πρέπει να καθοριστεί. Ο τελεστής ιστορικό εξέλιξης $h_{population}$ παίρνει τιμή 1 όταν γίνεται χρήση του ιστορικού της εξέλιξης ενός ατόμου. Για παράδειγμα, $h_{population}$=2,1,1 σημαίνει ότι τα άτομα ανταλλάσσουν όλες τις πληροφορίες κάθε δύο γενεές, ενώ αν έχουμε $h_{population}$=50,0.5,0 ότι ανταλλάσσουν μόνο τις μισές από τις πληροφορίες τους κάθε 50 γενεές.

- **Μέθοδος Αντικαταστασης** (6): Η παράμετρος αυτή παίρνει τιμές 1 για το μοντέλο γενικής αντικατάστασης, 2 για το μοντέλο σταθερής κατάστασης.

- **Ελιτισμός** (7): Η παράμετρος αυτή παίρνει τιμή 1 οταν εφαρμόζεται η λειτουργία του ελιτισμού και 0 όταν δεν εφαρμόζεται.

- **Μέθοδος Τοπικής Αναζήτησης** (8): Παίρνει την τιμή 1 για χρήση προσομοιούμενης ανόπτησης, 2 για χρήση αποτρεπτικής αναζήτησης, 3 για χρήση κατευθυνόμενης τοπικής αναζήτησης.

- **Δομή Πληθυσμού** (9): Παίρνει τιμή για τοπολογία πλήρους διασύνδεσης την τιμή 0, για τοπολογια πλέγματος την τιμή 1, για τοπολογία δακτυλίου την τιμή 2.

3.3.2 Η βασική διάταξη ενός MA βασισμένου στο μοντέλο νησίδων ΔMAN

Στην ενότητα αυτή βασισμένοι στη βασική ΔMA και έχοντας υπόψη τα χαρακτηριστικά στοιχεία ενός MA με νησίδες παραθέτουμε τη βασική διάταξη MA με νησίδες ΔMAN. Η ΔMAN είναι μια επέκταση της ΔMA. Περίεχει μια γραμμή για κάθε επίπεδο πληθυσμού.

Η στήλη 1 αναφέρεται στο χαμηλότερο επίπεδο ενός MA βασισμένου στο μοντέλο των νησίδων και καθορίζει τον αριθμό των ατόμων e που ομαδοποιούνται σε ένα σύνολο S που καλείται νησίδα. Στη δεύτερη στήλη αναφερόμαστε στο επιπεδο 2 όπου εχουμε ομαδοποίηση των νησίδων (elements e του επιπέδου 1 και δημιουργία μεγαλύτερων νησίδων $S(e)$.

Στη σειρά 5 αναφερόμαστε στη τοπολογία των ατόμων σε μια νησιδα για το επίπεδο 1 και των νησίδων σε ένα σύνολο νησιδων για το επιπεδο 2. Οι χρησιμοποιούμενες τοπολογίες είναι οι τοπολογίες πλήρους διασύνδεσης, δακτυλίου και πλέγματος και οι τιμές είναι αντίστοιχα 1,2 και 3.

Στη σειρά 10 εχουμε τιμή 1 ή 0 ανάλογα με το αν κάνουμε καταγραφή του ιστορικού της μια νησίδας για το επίπεδο 2 ή ενός ατόμου για το επίπεδο 1.

		Επίπεδο 1	Επίπεδο 2
(1)	**Μέγεθος Πληθυσμών**		
(2)	**Κωδικοποίηση**		
(3)	**Ανασυνδυασμός**		
(4)	**Μετάλλαξη**		
(5)	**Δομή Νησιδων**		
(6)	**Μέθοδος Αντικατάστασης Νησίδων**		
(7)	**Ελιτισμός**		
(8)	**Μέθοδος Τοπικής Αναζήτησης**		
(9)	**Ιστορικό εξέλιξης** h_e		

Πίνακας 3.2: Η βασική Διάταξη ενός Μιμητικού Αλγορίθμου βασισμένου σε νησίδες ΔΜΑΝ

3.4 Παραδείγματα

Στην ενότητα αυτή παραθέτουμε απλά παραδείγματα εφαρμογής της ΔΜΑ και ΔΜΑΝ.

3.4.1 Απλός Μιμητικός Αλγόριθμος

Βασιζόμενοι στην περιγραφή που έγινε στην ενότητα 3.3.1 η ΔΜΑ για ένα αλγοριθμο με μέγεθος πληθυσμου κυμαινόμενο από 50 μέχρι 200 με βήμα 20, δυαδική αναπαράσταση, ανασυνδυασμό ενός σημείου με πιθανότητα 0.6 , μετάλλαξη με πιθανότητα 0.1, μεθοδο γενικής αντικατάστασης, χωρίς καταγραφή ιστορικού, με χρήση ελιτισμού, με κατευθυνόμενη τοπική αναζήτηση και τοπολογία πλέγματος για τη διάταξη των ατόμων στο χώρο αναζήτησης ορίζεται στον πίνακα 3.3. Για λόγους ευκολίας κατά τη συγγραφή του υπόλοιπου κειμένου εχει υιοθετηθεί για μια ΔΜΑ όπως αυτή του σχήματος 3.3 η εξής φόρμα : ΔΜΑ=(50, 200, 20 , 0 , 0, 0.6 , 1, 0.1, 0.1, 1 , 0 , 1 , 3 , 1)

(1)	**Μέγεθος Πληθυσμού**	50,200,20
(2)	**Κωδικοποίηση**	0
(3)	**Ανασυνδυασμός**	0,0.6
(4)	**Μετάλλαξη**	1,0.1,0.1
(5)	**Ιστορικό Πληθυσμού**	1
(6)	**Μέθοδος Αντικατάστασης Πληθυσμού**	0
(7)	**Ελιτισμός**	1
(8)	**Μέθοδος Τοπικής Αναζήτησης**	3
(9)	**Δομή Πληθυσμού**	1

Πίνακας 3.3: Ένα απλό παράδειγμα ΔΜΑ

3.4.2 Μιμητικός Αλγόριθμος βασισμένος σε νησίδες

Το δεύτερο παράδειγμα που παραθέτουμε αφορά ένα μιμητικό αλγόριθμο βασισμένο στο μοντέλο νησίδων. Στο παράδειγμα αυτό έχουμε για μέθοδο αντικατάστασης το μοντέλο γενικής αντικατάστασης, χρησιμοποιείται τοπολογία τόσο στο πρώτο επίπεδο όσο και στο δεύτερο δακτυλίου για την μετανάστευση των ατόμων και των νησίδων αντίστοιχα με ρυθμό μεταναστευσης 0.5 και 0.6 αντίστοιχα. Οι τιμές για την μετάλλαξη και τον ανασυνδυασμό είναι ίδιες με αυτές του προηγουμενου παραδείγματος. Με βάση τη ταξινόμηση του Talbi [202] οι ΜΑς σε νησίδες ορίζονται ως ακολούθως:

HCH(HRH(EA+LCH(EA(LS)) (het,par,gen)) (het,par,gen)) (hom,glo,gen).

(1)	**Μέγεθος Πληθυσμών**	100	200
(2)	**Κωδικοποίηση**	0	0
(3)	**Ανασυνδυασμός**	3,0.6	3,0.75
(4)	**Μετάλλαξη**	1,0.01,0.02	1,0.01,0.02
(5)	**Δομή Νησιδων**	2,0.5	2,0.6
(6)	**Μέθοδος Αντικατάστασης Νησίδων**	1	1
(7)	**Ελιτισμός**	1	1
(8)	**Μέθοδος Τοπικής Αναζήτησης**	3	3
(9)	**Ιστορικό Εξέλιξης**	1	1

Πίνακας 3.4: Ένα παράδειγμα ΔΜΑΝ

Κεφάλαιο 4

Έλεγχος απόδοσης Μιμητικών Αλγορίθμων

Στο κεφάλαιο αυτό επιχειρείται συγκριτική αξιολόγηση έξι αντιπροσωπευτικών αλγορίθμων σε προβλήματα αναλυτικών αντικειμενικών συναρτήσεων. Οι μέθοδοι που εξετάζονται είναι:

- Απλός Εξελικτικός Αλγόριθμος
- Γενετικός Αλγόριθμος
- Μιμητικός Αλγόριθμος με Αποτρεπτική Αναζήτηση
- Μιμητικός Αλγόριθμος με Προσομοιούμενη Ανοπτηση
- Μιμητικός Αλγόριθμος με Καθοδηγούμενη Τοπική Αναζήτηση
- Πολιτισμικός Αλγόριθμος

Σημαντικό μέρος των όσων παρουσιάζονται σε αυτό το κεφάλαιο έχει δημοσιευτεί στο έβδομο Πανελλήνιο Συνέδριο Πληροφορικής [64] στο διεθνές συνέδριο IEEE Systems Man and Cybernetics [65], και σε τρία άρθρα στα περιοδικά International of Computer Mathematics και Journal of Applied Mathematics and Computation [66, 75, 78] και αναφέρεται από αρκετές εργασίες άλλων ερευνητών όπως οι [113, 162, 163, 12].

4.1 Μεθοδολογία Αξιολόγησης Μιμητικών Αλγορίθμων

4.1.1 Επισκόπηση

Η χρήση συναρτήσεων ελέγχου (test functions) με στόχο την αξιολόγηση αλγορίθμων βελτιστοποίησης είναι συνήθης πρακτική (π.χ., [8, 184]). Στη βιβλιογραφία διατίθεται μεγάλο πλήθος θεωρητικών προβλημάτων βελτιστοποίησης, με διαφορετικά χαρακτηριστικά (π.χ. [190]). Τα προβλήματα αυτά κατατάσσονται σε κατηγορίες ανάλογα με:

- το πλήθος των μεταβλητών ελέγχου
- το πλήθος των ακρότατων
- τη γνώση της θέσης του ολικού ακρότατου
- τη γεωμετρία της επιφάνειας απόκρισης
- την ύπαρξη θορύβου ή ασυνεχειών στην αντικειμενική συνάρτηση

Κατά τη βελτιστοποίηση μη κυρτών συναρτήσεων το πλήθος των δοκιμών, και κατά συνέπεια ο χρόνος επίλυσης, αυξάνει σχεδόν εκθετικά με το πλήθος των μεταβλητών ελέγχου, ενώ αν η συνάρτηση είναι κυρτή, η σχέση δοκιμών-μεταβλητών ελέγχου είναι σχεδόν γραμμική [85, 205, 216]. Στην ιδεατή περίπτωση βελτιστοποίησης μιας τετραγωνικής συνάρτησης με τη μέθοδο συζυγών κλίσεων, απαιτούνται ακριβώς n δοκιμές για τον εντοπισμό του βέλτιστου, ανεξάρτητα από το σημείο εκκίνησης. Από την άλλη πλευρά, σε συναρτήσεις με περισσότερα από ένα ακρότατα (πολυκόρυφες) (multimodal) υπάρχει πάντοτε μη μηδενική πιθανότητα εγκλωβισμού του αλγορίθμου σε ένα από τα τοπικά βέλτιστα. Κατά κανόνα, οι πολυδιάστατες συναρτήσεις χρησιμοποιούνται για την αξιολόγηση της αποδοτικότητας (ταχύτητας) των αλγορίθμων βελτιστοποίησης, ενώ οι πολυκόρυφες για την αξιολόγηση της αποτελεσματικότητας τους [190].

Σε ορισμένα μαθηματικά προβλήματα, δεν είναι γνωστά ούτε η θέση του ολικού ακρότατου (άρα και η βέλτιστη τιμή της συνάρτησης) ούτε το πλήθος των τοπικών ακροτάτων. Τα προβλήματα αυτά προσεγγίζουν καλύτερα την πραγματικότητα, όπου ζητούμενο είναι όχι ο ακριβής εντοπισμός μιας συγκεκριμένης λύσης αλλά η εύρεση μιας όσο το δυνατόν πιο ικανοποιητικής λύσης, σε λογικά χρονικά πλαίσια. Στην περίπτωση αυτή, μέτρο της αποτελεσματικότητας είναι η διασπορά των λύσεων που εντοπίζονται σε ένα μεγάλο πλήθος στοχαστικά ανεξάρτητων εφαρμογών του προς αξιολόγηση αλγορίθμου.

Τα γεωμετρικά χαρακτηριστικά της επιφάνειας απόκρισης επηρεάζουν τόσο την αποτελεσματικότητα όσο και την αποδοτικότητα των αλγορίθμων βελτιστοποίησης. Γενικά, στις έντονα

μακρόστενες κοιλάδες, τις κορυφογραμμές και τους αυχένες η διαδικασία βελτιστοποίησης επιβραδύνεται σημαντικά, διότι είναι δύσκολος ο εντοπισμός της κλίσης της συνάρτησης. Ο θόρυβος, δηλαδή οι τυχαίες διαταραχές σε μια συνάρτηση, έχει ως αποτέλεσμα σε κάθε σημείο να μην αντιστοιχεί μονοσήμαντα η ίδια τιμή.

Ο Dejong [61] παρουσίασε ένα σύνολο από 5 συναρτήσεις οι οποίες και έμειναν γνωστές στη βιβλιογραφία ως συναρτήσεις Dejong. Οι συναρτήσεις που πρότεινε ο Dejong έχουν χαρακτηριστικά γνωρίσματα που αντικατοπτρίζουν αρκετές από τις δυσκολίες που χαρακτηρίζουν τα προβλήματα βελτιστοποίησης. Το προτεινόμενο από αυτόν σύνολο είναι το παρακάτω :

F1 : Sphere model
F2 : Generalized Rosenbrock
F3 : Step function
F4 : Quartic function with noise
F5 : Shekel foxholes

Οι συναρτήσεις αυτές απετέλεσαν ένα μέτρο σύγκρισης για πολλές μεταγενέστερες μελέτες και εφαρμογές GA. Μια εκτενή αναφορά στο παραπάνω σύνολο συναρτήσεων μπορεί κανείς να δεί στον Goldberg [98] (σελ. 106-120). Ο Baeck [15], δίνει έμφαση σε δύο κριτήρια ελέγχου α) στην εύρεση του ολικού βέλτιστου, β) στην γρήγορη σύγκλιση και χρησιμοποίησε δύο συναρτήσεις α) Sphere model β) Rastrigin Στην εργασία αυτή παρουσίασε τις βέλτιστες μέσες τιμές για 10 εκτελέσεις GA χρησιμοποιώντας το παρακάτω σύνολο παραμέτρων :

Πιθανότητα Ανασυνδυασμού : 0.6
Μεγεθος Πληθυσμού : 50
Μήκος αλφαριθμητικού : 32n (όπου n ο βαθμός αντικειμενικής συνάρτησης)
Πιθανότητα Μετάλλαξης : 0.001
Ανασυνδυασμό δύο σημείων
Αναλογική Επιλογή

Οι Patton et al [169] σκοπό τους είχαν να επισημάνουν τη σημασία της κατευθυνόμενης Gaussian μετάλλαξης GGM (Guided Gaussian Mutation) δοκιμάζοντας γενετικούς και εξελεκτικούς αλγορίθμους. Επανέλαβαν τα πειράματα τους 100 φορές για κάθε συνάρτηση και μας παρουσίασαν τις βέλτιστες μέσες τιμές και την τυπική απόκλιση. Χρησιμοποίησαν GA 500 γενεών και μέγεθος πληθυσμού από 200 ως 500 ενώ χρησιμοποίησαν τις εξής συναρτήσεις :

Sphere model — Rastrigin
Sphere-Hull — Schaffer
Schwefel 1 — Schweffel 2
Ackley — Griewangk
Bohachevsky — Rosenbrock

Ο Culberson στην εργασία του [58] στο τμήμα που αφορά τη σύγκριση των δύο μεθόδων (GA και της γενετικής σταθερότητας (genetic invariance) - GI) χρησιμοποιεί για συναρτήσεις ελέγχου της απόδοσης τις 5 συναρτήσεις DeJong. Χρησιμοποιήσε πιθανότητα μετάλλαξης ίση με 0.01 και μοντέλο GA το αντικατάστασης σταθερής κατάστασης. Για μέγεθος πληθυσμού δοκίμασε τις τιμές 70, 100, 150 ενώ ήλεγξε τα αποτελέσματα του με ή χωρίς μετάλλαξη.
Ο Meysenburg [150] προκειμένου να μετρήσει την απόδοση των GA χρησιμοποίησε τις παρακάτω συναρτήσεις :

De Jong F1 - F5
Rastrigin - Michalewicz
Schwefel's - Ackley's
Griewangk - Langerman

Οι Agoston Endre Eiben στην εργασία τους [83] χρησιμοποίησαν τις παρακάτω συναρτήσεις:

De Jong F1-F3
Griewangk

με τις παρακάτω παραμέτρους :

Μέγεθος Πληθυσμού : 400
30 εκτελέσεις ανά συνάρτηση
Tournament selection
Uniform Crossover
Crossover Probability : 0.6
Mutation Probability : 0.002

Ο Baeck [16] προτείνει τις συναρτήσεις :

Sphere Model - Step function
Ackley - Powell Fletcher
Weierstrass - Mandelbrot

Ο Pohlheim [172, 173] περιγράφει ένα αρκετά μεγάλο αριθμό συναρτήσεων για έλεγχο της απόδοσης των GA της εργαλειοθήκης του Matlab GEATbx. Οι συναρτήσεις αυτές είναι παρμένες από την βιβλιογραφία που αφορά τους GA, τις εξελικτικές στρατηγικές και την βελτιστοποίηση συναρτήσεων. Το σύνολο που χρησιμοποίησε είναι

DeJong's F1-F2
Langerman - Axis parallel hyper-ellipsoid
Michalewicz - Rotated hyper-ellipsoid
Branins - Rastrigin
Easom - DeJong's F1-F2
Langerman - Schwefel
Goldstein-Price's - Griewangk
Ackley

Εκτός από τις εργασίες αυτές που αναφέραμε και οι οποίες έγιναν από ερευνητές της περιοχής των EA μπορεί κανείς να δεί ότι υπάρχουν και άλλες προτάσεις για σύνολα συναρτήσεων από επιστήμονες που ανήκουν σε άλλες συναφείς με το αντικείμενο γνωστικές περιοχές. Ενδεικτικά αναφέρουμε μία από τις πλέον γνωστές εργασίες στο χώρο της βελτιστοποίησης συναρτήσεων, αυτή των More, Garbow, Hillstrom [116] οι οποίοι πρότειναν ένα σύνολο 34 συναρτήσεων το οποίο και αποτέλεσε έκτοτε σημείον αναφοράς για αρκετούς ερευνητές. Στην συγκεκριμένη

εργασία δίδεται έμφαση περισσότερο στη μέτρηση της απόδοσης του λογισμικού για βελτιστοποίηση συναρτήσεων. Οι συναρτήσεις που πρότειναν προσφέρονται για συγκριτικές δοκιμές διαφόρων αλγοριθμικών μεθόδων που χρησιμοποιούνται στη βελτιστοποίηση.

Πίνακας 4.1: Συναρτήσεις F1-F8

Συν. Ονομ.	Περιγραφή Συναρτήσεων	Ονομασία
$F1$	$f_1 = \sum_{i=1}^{2} x_i^2$, $-5.12 \leq x_i \leq 5.12$	Sphere model
$F2$	$f_2 = 100(x_1^2 - x_2)^2 + (1 - x_1)^2)$, $-2.048 \leq x_i \leq 2.048$	Rosenbrock
$F3$	$f_3 = \sum_{i=1}^{5} int(x_i)$, $-5.12 \leq x_i \leq 5.12$	Step
$F4$	$f_4 = \sum_{i=1}^{30}(ix_i^4 + Gauss(0,1))$, $-1.28 \leq x_i \leq 1.28$	Quatric
$F5$	$f_5(x_i) = 0.002 + \sum_{j=1}^{25}(\frac{1}{j} + + \sum_{i=1}^{2}(x_i - \alpha_{ij})^6), -65536 \leq x_i \leq 65536$	Foxholes
$F6$	$f_6 = 10V + \sum_{i=1}^{10}(-x_i sin(\sqrt{\lvert x_i \rvert}))$, $V = 4189.829, -500 \leq x_i \leq 500$	Rastrigin
$F7$	$f_7 = 20A + \sum_{i=1}^{20}(x_i^2 - 10\cos(2\pi x_i))$, $A = 10, -5.12 \leq x_i \leq 5.12$	Schwefel
$F8$	$f_8 = 1 + \sum_{i=1}^{10} \frac{x_i^2}{4000} - \prod_{i=1}^{10} \cos(\frac{x_i}{\sqrt{i}})$, $-600 \leq x_i \leq 600$	Griewang

4.1.2 Προτεινόμενο σύνολο συναρτήσεων για την αξιολόγηση Μιμητικών Αλγορίθμων

Για την αξιολόγηση των διαφόρων εξελικτικών μεθόδων που παρουσιάζονται στη διατριβή μας χρησιμοποιήθηκαν 8 μαθηματικές συναρτήσεις ελέγχου, τα χαρακτηριστικά μεγέθη των οποίων

Πίνακας 4.2: Συναρτήσεις F9-F14

Συν. Ονομ.	Περιγραφή Συναρτήσεων	Ονομασία
$F9$	$f_i = \sum_{i=2}^{n}(j-1)x_j t_i^{j-2}$	
	$-\sum_{j=1}^{n}(x_j t_i^{j-1})^2 - 1$	
	όπου $t_i = i/29, 1 \leq i \leq 29,$	Watson
	$f_{30} = x_1, f_{31} = x_2 - x_1^2 - 1,$	
	$x_0 = (0, ..., 0)$, $2 \leq n \leq 31, m = 31$	
$F10$	$f_{2i} - 1 = 10(x_{2i} - x_{i=1}^2),$	
	$f_{2i}(x) = 1 - x_{2i}$	
	$x_0 = (\xi_j)$	Extended Rosenbrock
	όπου $\xi_{2j} - 1 = -1.2, \xi_{2j} = 1$	
	n μεταβλητό αλλά $m = n$	
$F11$	$f_i(x) = x_i - 0.2$	
	$f_i(x) = \alpha^{1/2}(\exp[\frac{x_i}{10}] + \exp[\frac{x_{i-1}}{10}] - y_i),$	
	$2 \leq i \leq n$	
	$f_i(x) = \alpha^{1/2}(\exp[\frac{x_{i-n+1}}{10}] + \exp[\frac{-1}{10}]),$	
	$n \leq i \leq 2n$	Penalty II
	$f_{2n}(x) = (\sum_{i=1}^{n}(n-j+1)x_j^2) - 1$	
	όπου $\alpha = 10^5, \exp[\frac{i}{10}] + \exp[\frac{i-1}{10}]$	
	$x_0 = (\frac{1}{2}, ..., \frac{1}{2})$	
	n μεταβλητό $m = 2n$	
$F12$	$f_1(x) = 10^4 x_1 x_2 - 1$	
	$f_2(x) = \exp[-x_1] + \exp[-x_2] - 1.0001$	Powel badly scaled
	$x_0 = (0, 1)$, $n = 2, m = 2$	
$F13$	$f_i(x) = \exp[\frac{-\lvert y_i mix_2\rvert^{x_i}}{x_1}] - t_i$	
	$t_i = i/100 y_i = 25 + (-50\ln(t_i))^{2/3}$	Gulf research and development
	$x_0 = (5, 2.5, 0.15)$	
	$n = 3, n \leq m \leq 100$	
$F14$	$f_{4i-3}(x) = x_{4i-3} + 10x_{4i-2}$	
	$f_{4i-2}(x) = 5^{1/2} + (x_{4i-1} - x_{4i})$	
	$f_{4i-1}(x) = (x_{4i-2} + 2x_{4i-1}^2$	Extended Powel singular
	$f_{4i}(x) = 10^{1/2} + (x_{4i-3} - x_{4i})^2$	
	$x_0 = (\xi_j)$	
	όπου $\xi_{4j} - 3 = 3, \xi_{4j} - 2 = -1,$	
	$\xi_{4j} - 1 = 0, \xi_{4j} = 1$	
	n μεταβλητό $m = n$	

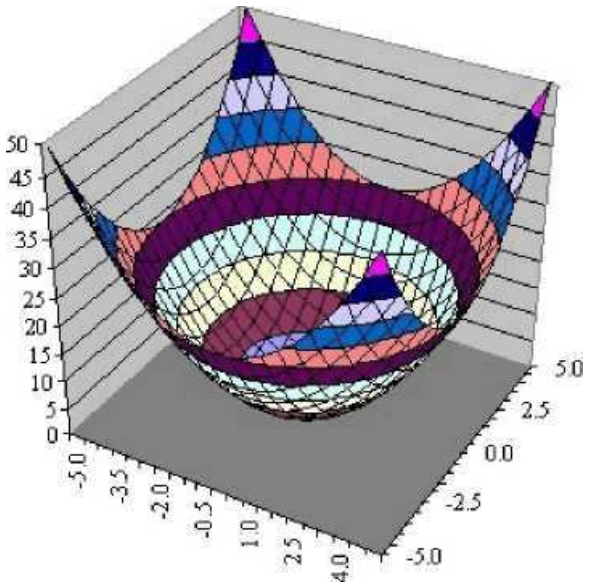
παρουσιάζονται στον Πίνακα 4.1 και 6 προβλήματα μη γραμμικού προγραμματισμού (Πίνακας 4.2).

Οι συναρτήσεις Dejong που είναι υποσύνολο στο δικό μας σύνολο συναρτήσεων ελέγχου είναι 5 προβλήματα που αφορούν στην ελαχιστοποίηση συναρτήσεων και μας βοηθούν αρκετά στο να μελετήσουμε τη συμπεριφορά αρκετών χαρακτηριστκών των ΕΑ και ειδικότερα των ΜΑ. Πιο συγκεκριμένα :

Το μοντέλο σφαίρας F1 είναι ένα κλασσικό παράδειγμα συνεχούς συνάρτησης. Είναι μονοκόρυφη (unimodal) , υψηλής κυρτότητας (strongly convex), ομαλή (smooth) , και συμμετρική. Γι΄ αυτό αποτελεί αρχική συνάρτηση για έλεγχο της απόδοσης κάθε μεθόδου βελτιστοποίησης. Χρησιμοποιείται στον έλεγχο του κατά πόσο γρήγορα επιτυγχάνεται η σύγκλιση και έχει καθιερωθεί στη θεωρία των εξελικτικών αλγορίθμων σαν μια από τις βασικές συναρτήσεις ελέγχου της απόδοσης κάθε είδους εξελικτικού αλγορίθμου. Θεωρείται ιδανική αντικειμενική συνάρτηση, καθώς ειναι τετραγωνική και άρα κυρτή σε όλο το πεδίο ορισμού της, παρουσιάζει απόλυτη ομαλότητα και έχει ένα μόνο ακρότατο στο σημείο (0,0,...,0). Η ονομασία μοντέλο σφαίρας πρόερχεται απο την μετάφραση του αγγλικού όρου (sphere) . Στην πραγματικότητα όμως η γραφική παράσταση της συνάρτησης δεν είναι σφαίρα αλλά παραβολοειδές (Σχήμα 4.1). (βλ. [151]).

Η Rosenbrock F2 είναι ομαλή, μονοπύθμενη , με έντονο πυθμένα (sharp ridge) και γενικά θεωρείται δύσκολη καθότι οι διάφοροι αλγόριθμοι που χρησιμοποιούνται δυσκολεύονται στο να βρούν καλές λύσεις για αυτήν. Η συνάρτηση έχει μόνο ένα ακρότατο στο σημείο (0,0,....,0), το οποίο βρίσκεται πάνω σε μια πολύ στενή χαράδρα, παραβολοειδούς μορφής (Σχήμα 4.2).

Η Step F3 βηματική συνάρτηση αναπαριστά προβλήματα επιπέδων επιφανειών (flat surfaces).

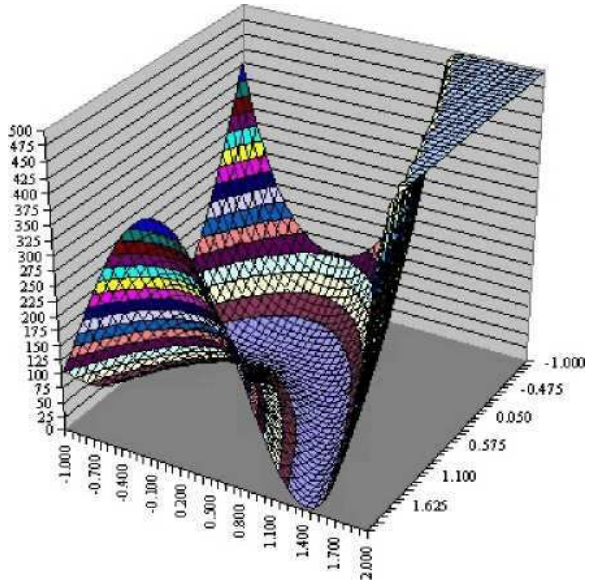


Σχήμα 4.1: Γραφική παράσταση μοντέλου σφαίρας

Τα επίπεδα αποτελούν εμπόδιο για αλγορίθμους βελτιστοποίησης, γιατί δεν μπορούν να πάρουν πληροφορίες σχετικά με το ποια λύση είναι η καλύτερη. Οι επίπεδες επιφάνειες καθιστούν αδύνατον τον προσδιορισμό μιας διεύθυνσης μεταβολής της τιμής της συνάρτησης. Κάθε ένα επίπεδο αντιστοιχεί σε ένα τοπικό ελάχιστο για την συνάρτηση (Σχήμα 4.3). Ανήκει στην κατηγορία των πολυκόρυφων συναρτήσεων. Η βελτιστοποίηση της βηματικής συνάρτησης δεν είναι παρά ένα στοιχείωδες πρόβλημα ακέραιου προγραμματισμού.

Η Quatric F4 είναι μία απλή μονοκόρυφη συνάρτηση με θόρυβο. Οι αλγόριθμοι δεν μπορούν να εφαρμοστούν ικανοποιητικά εξαιτίας του θορύβου Gaussian στα δεδομένα είναι ένα παράδειγμα συναρτήσεων τοπικών βέλτιστων (στη δική μας περίπτωση 25). Η δυσκολία έγκειται στην εύρεση του ολικού βέλτιστου.

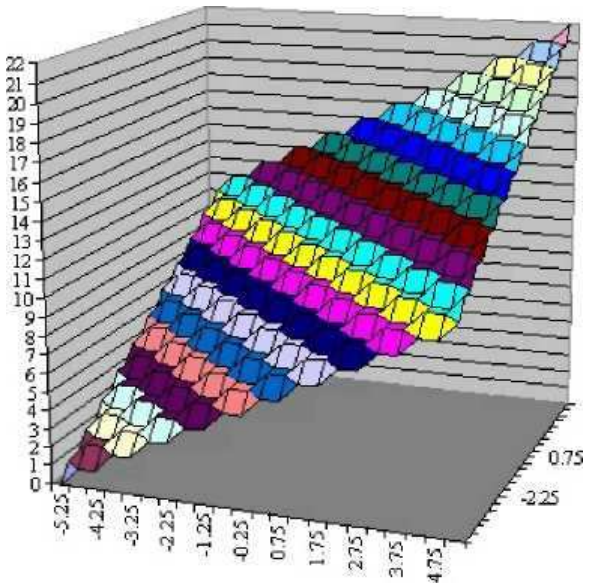
Οι συναρτήσεις Schwefel, Rastrigin, Griewangk είναι κλασσικά παραδείγματα πολυκόρυφων



Σχήμα 4.2: Γραφική παράσταση συναρτησης Rozenbrock

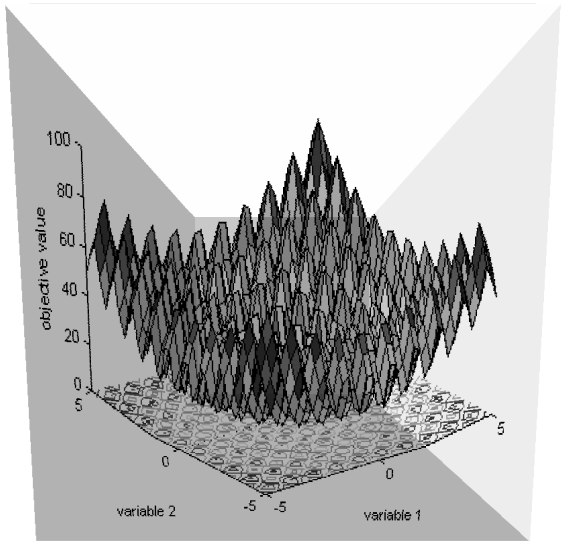
συναρτήσεων και είναι μη γραμμικές.

Για την συνάρτηση Rastrigin η βέλτιστη τιμή για κάθε παράμετρο μπορεί να καθοριστεί ανεξάρτητα από τις υπόλοιπες μεταβλητές. Έχει βαθμό πολυπλοκότητας O(nln(n)), όπου n ο αριθμός των παραμέτρων της συνάρτησης. Η επιφάνεια της συνάρτησης εξαρτάται από τις μεταβλητές A και ω, οι οποίες ελέγχουν το πλάτος και τη συχνότητα διαμόρφωσης αντίστοιχα. Όταν A=10, το επιλεγμένο πεδίο ορισμού κυριαρχείται από τη συχνότητα διαμόρφωσης. Είναι συνάρτηση συνεχής με πάρα πολλά τοπικά ελάχιστα. (Σχήμα 4.4). Τα τοπικά ελάχιστα βρίσκονται σε ένα ορθογώνιο πλέγμα με πλάτος 1. Η τιμή των τοπικών ελάχιστων αυξάνεται όσο αυξάνεται η απόσταση των σημείων από το ολικό ελάχιστο. Η Schwefel είναι πολυκόρυφη , συνεχής, μη γραμμική συνάρτηση και σχετικά πιο εύκολη από την Rastrigin και χαρακτηρίζεται από το δεύτερο καλύτερο ελάχιστο το οποίο απέχει πολύ από το ολικό βέλτιστο (Σχήμα 4.5). Το



Σχήμα 4.3: Γραφική παράσταση βηματικής συναρτησης

V μπαίνει για δική μας ευκολία και η ακριβής τιμή του εξαρτάται από τα χαρακτηριστικά του συστήματος. Η Griewang έχει βαθμό πολυπλοκότητας O(nln(n)), όπου n ο αριθμός των παραμέτρων της συνάρτησης. Οι όροι του αθροίσματος παράγουν μία παραβολή ενώ τα τοπικά βέλτιστα βρίσκονται πάνω από το επίπεδο της παραβολής (Σχήμα 4.6). Οι διαστάσεις του εύρους αναζήτησης αυξάνονται βάσει των όρων του γινομένου με αποτέλεσμα τα τοπικά ελάχιστα να γίνονται μικρότερα. Όσο πιο πολύ μεγαλώνουμε το εύρος αναζήτησης τόσο πιο επίπεδη γίνεται η συνάρτηση. Γενικά αποτελεί μια καλή συναρτηση για τον έλεγχο της αποδοσης των ΕΑ κυρίως επειδή το γινόμενο είναι η αιτία δημιουργίας υποπληθυσμών ισχυρά αλληλοεξαρτώμενων σε μοντέλα παράλληλων ΕΑ. Τα υπόλοιπα απο τα προβλήματα του συνόλου που καθορίσαμε για τον έλεγχο της απόδοσης των ΕΑ έχουν ληφθεί από τον χώρο του μαθηματικού προγραμματισμού και καλύπτουν κυρίως μη γραμμικά προβλήματα χωρίς περιορισμούς (βλ. [116]). Επιλέξαμε τα



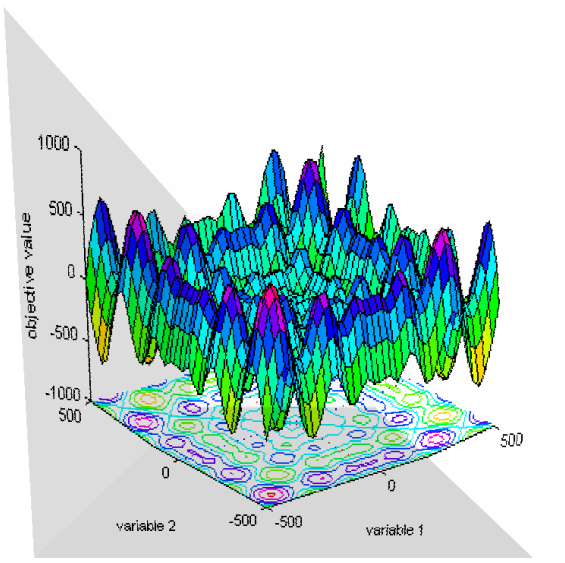
Σχήμα 4.4: Γραφική παράσταση της συνάρτησης Rastrigin

προβλήματα αυτά με τα εξής κριτήρια:

- Να έχουμε ποικιλία στον αριθμό των μεταβλητών (x)
- Να έχουμε εν γένει δύσκολες συναρτήσεις για βελτιστοποίηση, ακόμα και σε περιπτώσεις που η αναλυτική μορφή της συνάρτησης είναι γνωστή.

Η μορφοποίηση για την περιγραφή του καθενός από αυτά είναι :

- n ο αριθμός μεταβλητών του προβλήματος και m ο αριθμός των όρων της αντικειμενικής συνάρτησης.
- Οι όροι f_i της αντικειμενικής συνάρτησης, ώστε αυτή να δίνεται από το άθροισμα $f_(x) = \sum_{i=1}^{m} f_i^2(x)$
- το αρχικό σημείο εκκίνησης x_0

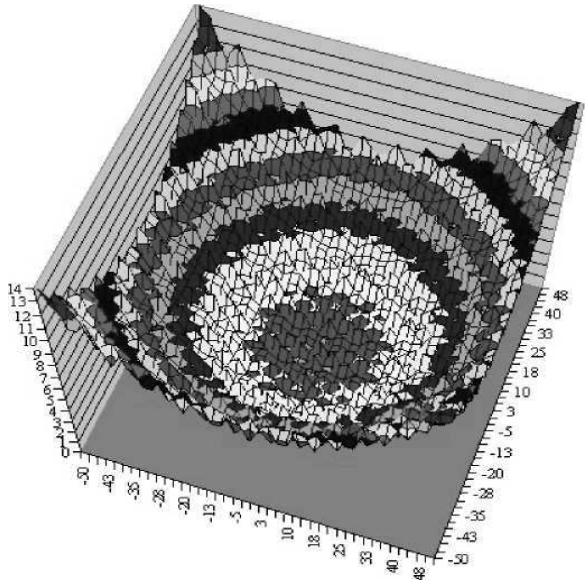


Σχήμα 4.5: Γραφική παράσταση της συνάρτησης schwefel

- το ή τα ελάχιστα καθώς και τα σημεία εμφάνισης τους όπου αυτά είναι γνωστά.

Η δυνατότητα επίλυσης ενός τέτοιου προβλήματος καθορίζεται από δύο παράγοντες. Πρώτα θα πρέπει να είναι δυνατή η διατύπωση μιας αυτόματης μεθόδου επίλυσης, δηλαδή μιας μεθόδου που να λύνει όλα τα προβλήματα που ανήκουν στην ίδια κατηγορία ακολουθώντας πάντα τα ίδια συγκεκριμένα βήματα. Μία τέτοια μέθοδος, στην οποία συγκαταλέγονται και οι ΕΑ, είναι απαραίτητη για να είναι δυνατός ο προγραμματισμός της μεθόδου και επομένως η επίλυση του προβλήματος με τη βοήθεια ηλεκτρονικού υπολογιστή.

Οι αντικειμενικές συναρτήσεις των συστημάτων που έχουμε προς επίλυση εκφράζουν το κριτήριο αριστοποίησης κατά περίπτωση. Τα μεγέθη των συστημάτων αντιπροσωπεύονται ανάλογα από τις σταθερές ή μεταβλητές του μαθηματικού προβλήματος ενώ οι περιορισμοί μεταξύ των μεταβλητών εκφράζουν τους νόμους που διέπουν το σύστημα.



Σχήμα 4.6: Γραφική παράσταση της συνάρτησης Griewank

Η μαθηματική διατύπωση σε αρκετά από τα προβλήματα μας είναι τέτοια που συνεπάγεται μεγάλη αύξηση στην υπολογιστική πολυπλοκότητα του μαθηματικού συστήματος. Η φύση του κάθε προβλήματος καθορίζει την δυνατότητα αλλά και την ευκολία με την οποία επιλύεται στον υπολογιστή μας. Οι ΕΑ αποτελούν και αυτοί με τη σειρά τους μια συστηματική διαδικασία που ακολουθείται για την επίλυση τέτοιων προβλημάτων και υπό την εννοια αυτή στη συγκεκριμένη εργασία χρησιμοποιείται το συγκεκριμένο σύνολο προβλημάτων για έλεγχο της απόδοσης αυτών.

4.1.3 Δείκτες επίδοσης εξελικτικών αλγορίθμων

Κατά κανόνα, κάθε μέθοδος βελτιστοποίησης ελέγχεται ως προς τους δύο παράγοντες την αποτελεσματικότητα και την αποδοτικότητα.

Μέτρο της αποτελεσματικότητας είναι η μέση απόκλιση από τη θεωρητικά βέλτιστη λύση για ένα πλήθος N στοχαστικά ανεξάρτητων εκτελέσεων του αλγορίθμου. Η στοχαστική ανεξαρτησία έγκειται στην εκκίνηση της διαδικασίας βελτιστοποίησης από διαφορετικές τυχαίες αρχικές συνθήκες (αρχική λύση ή πληθυσμό λύσεων). Ο δείκτης αποτελεσματικότητας ορίζεται ως ο λόγος των επιτυχών προς τις συνολικές εκτελέσεις του αλγορίθμου. Μια εκτέλεση κρίνεται επιτυχής εφόσον η τιμή που επιστρέφει βρίσκεται κοντά στη βέλτιστη του εκάστοτε προβλήματος βελτιστοποίησης, δηλαδή:

$$|f_k^{[i]} - f_k^*| < \alpha_k$$

όπου $f_k^{[i]}$ Η η λύση του k προβλήματος κατά την i εκτέλεση του αλγορίθμου,f_k^* η θεωρητικά βέλτιστη τιμή της συνάρτησης και α_k μια αυθαίρετη τιμή ανοχής, εξαρτώμενη από το βαθμό δυσκολίας του εκάστοτε προβλήματος. Στην περίπτωση κατά την οποία όλες οι δοκιμές συγκλίνουν στο ολικό ακρότατο, ο δείκτης αποτελεσματικότητας είναι 100%. Εφόσον επιλύονται K προβλήματα βελτιστοποίησης, εισάγεται ο μέσος δείκτης αποτελεσματικότητας, ο οποίος ορίζεται ως η μέση τιμή των επιμέρους δεικτών.

Σημειώνεται ότι, με το παραπάνω κριτήριο, ελέγχεται η τιμή και όχι η θέση του ολικού ακρότατου. Με άλλα λόγια, η εκτέλεση του αλγορίθμου θεωρείται επιτυχής εφόσον συγκλίνει σε οποιοδήποτε σημείο, η τιμή της συνάρτησης στο οποίο απέχει από τη θεωρητικά βέλτιστη λιγότερο από την ανοχή a_k. Το κριτήριο αυτό αντιπροσωπεύει καλύτερα την πραγματικότητα, όπου δεν είναι γνωστή η θέση του βέλτιστου αλλά μπορεί να είναι γνωστή, έστω και κατ΄ εκτίμηση, η βέλτιστη τιμή της αντικειμενικής συνάρτησης. Για παράδειγμα, σε προβλήματα ελαχιστοποίησης σφαλμάτων (π.χ., κατά τη βαθμονόμηση μαθηματικών μοντέλων) μπορεί να θεωρηθεί αποδεκτή

οποιαδήποτε λύση πλησιάζει την τιμή μηδέν.

Ως μέτρο της αποδοτικότητας ενός αλγορίθμου θα μπορούσε να θεωρηθεί ο χρόνος επίλυσης του προβλήματος βελτιστοποίησης, ανεξάρτητα αν η λύση που προκύπτει είναι ολικά βέλτιστη ή όχι. Ωστόσο, ο χρόνος είναι έννοια σχετική διότι εξαρτάται από εξωγενείς παράγοντες, όπως η ταχύτητα του επεξεργαστή και ο χρόνος υπολογισμού της τιμής της αντικειμενικής συνάρτησης. Ένα πιο αξιόπιστο μέτρο αποδοτικότητας είναι το πλήθος των σημείων δειγματοληψίας, δηλαδή η συχνότητα υπολογισμού της τιμής της συνάρτησης. Αυτή εξαρτάται σε σημαντικό βαθμό από τα κριτήρια σύγκλισης που υιοθετούνται. Προφανώς, όσο αυστηρότερο γίνεται το κριτήριο σύγκλισης, τόσο πιο πολύ μειώνεται η ταχύτητα του αλγορίθμου, ωστόσο τόσο περισσότερο αυξάνει η πιθανότητα σύγκλισης στο ολικό ακρότατο. Για να είναι αμερόληπτη η σύγκριση της αποδοτικότητας δυο αλγορίθμων, θα πρέπει τα κριτήρια σύγκλισης που υιοθετούνται να είναι παρόμοια.

Ένας άλλος παράγοντας που σχετίζεται με την αποδοτικότητα, αλλά είναι πολύ δύσκολο να εκτιμηθεί ποσοτικά, είναι η πολυπλοκότητα ενός αλγορίθμου. Κατά κανόνα, η επίδραση της πολυπλοκότητας στο συνολικό χρόνο υπολογισμών είναι αξιόλογη μόνο όταν το πλήθος των μεταβλητών του προβλήματος είναι αρκετά μεγάλο, οπότε οι διαδικασίες προσπέλασης της μνήμης του υπολογιστή απαιτούν σχετκά πολύ χρόνο.

Η αποτελεσματικότητα και η αποδοτικότητα ενός αλγορίθμου βελτιστοποίησης είναι έννοιες κατά κάποιο τρόπο αντικρουόμενες. Για παράδειγμα, η συστηματική αναζήτηση πάνω σε πλέγμα πολύ πυκνής διακριτοποίησης εγγυάται τον εντοπισμό του ολικού βέλτιστου (μεγάλη αποτελεσματικότητα), αλλά απαιτεί απαγορευτικό αριθμό δοκιμών (μικρή αποδοτικότητα). Από την άλλη πλευρά, με την εφαρμογή μιας μεθόδου κλίσης για τη βελτιστοποίηση μιας πολυκόρυφης

συνάρτησης είναι δυνατό να εντοπιστεί γρήγορα (μεγάλη αποδοτικότητα) μόνο το κοντινότερο στην αρχική λύση τοπικό ακρότατο (μικρή αποτελεσματικότητα).

4.2 Αξιολόγηση αλγορίθμων

4.2.1 Επιλογή και υλοποίηση αλγορίθμων

Στα πλαίσια της παρούσας εργασίας υλοποιήθηκαν και αξιολογήθηκαν από την κατηγορία των εξελικτικών μεθόδων οι παρακάτω αλγόριθμοι:

- Απλός Εξελικτικός Αλγόριθμος [99]
- Γενετικός Αλγόριθμος [107]
- Μιμητικός Αλγόριθμος με Αποτρεπτική Αναζήτηση [128, 144, 140]
- Μιμητικός Αλγόριθμος με Προσομοιούμενη Ανόπτηση [128, 140, 52]
- Μιμητικός Αλγόριθμος με Καθοδηγούμενη Τοπική Αναζήτηση [52, 144, 125]
- Πολιτισμικός Αλγόριθμος [194, 199]

Τα πειράματα εκτελέστηκαν σε ένα Pentium με ταχύτητα χρονισμού 200MHz, με 64 Mb RAM. Το λειτουργικό σύστημα ήταν Red Hat Linux 5. Κατά την διάρκεια των πειραμάτων μας ο υπολογιστής μας δεν εκτελούσε άλλες βαριές εργασίες. Επίσης παρατίθενται προς σύγκριση και τα αποτελέσματα που έδωσαν 3 αλγόριθμοι από την κατηγορία των ευρετικών μεθόδων. Αυτοί αλγόριθμοι ήσαν οι:

- Προσομοιούμενη Ανόπτηση [120]
- Αποτρεπτική Αναζήτηση [96]
- Καθοδηγούμενη Τοπική Αναζήτηση [207]

Τέλος οι αλγόριθμοι μας εχουν υλοποιηθεί σε γλώσσα C.

4.2.2 Αποτελέσματα

Κάθε πρόβλημα βελτιστοποίησης επλύθηκε $N = 30, 50, 100, 500, 1000$ φορές, ξεκινώντας από διαφορετικές τυχαίες αρχικές συνθήκες. Οι τιμές ανοχής (βάσει τον οποίων ελέγχθηκε αν η εκτέλεση του αλγορίθμου ήταν επιτυχής) ορίστηκαν ανάλογα με το βαθμό δυσκολίας του εκάστοτε προβλήματος και ήταν οι εξής:

Τιμές ανοχής των συναρτησεων		
f1 $\rightarrow \alpha = 0.1$	f2 $\rightarrow \alpha = 1$	f3 $\rightarrow \alpha = 0.0$
f4 $\rightarrow \alpha = 0.04$	f5 $\rightarrow \alpha = 0.6$	f6 $\rightarrow \alpha = 0.4$
f7 $\rightarrow \alpha = 0.32$	f8 $\rightarrow \alpha = 0.7$	f9 $\rightarrow \alpha = 0.2$
f10 $\rightarrow \alpha = 1.0$	f11 $\rightarrow \alpha = 1$	f12 $\rightarrow \alpha = 0.0$
f13 $\rightarrow \alpha = 0.2$	f14 $\rightarrow \alpha = 0.4$	

Εξελικτικός Αλγόριθμος

Στον Εξελικτικό Αλγόριθμο χρησιμοποιηθηκαν οι τρείς τυπικοί γενετικοί τελεστές (επιλογή με τον τροχό της ρουλέτας, διασταυρωση πολλαπλών σημείων, και μετάλλαξη) πάνω σε ένα πληθυσμό *pop* δυαδικά κωδικοποιημένων λύσεων. Αρχικά εντοπίζονται η καλύτερη x_1 και η

χειρότερη x_{n+1} και η δεύτερη χειρότερη x_n τιμή της συνάρτησης με βάση το κριτήριο

$$f(x_i) + Zlog(w)$$

όπου w ένας τυχαίος αριθμός που παράγεται από μια ομοιόμορφη κατανομή πιθανοτήτων και Z ένας τυχαίος αριθμός. Στο κριτήριο αυτό προστίθεται μια θετική, λογαριθμικά κατανεμημένη τυχαία μεταβλητή ανάλογη του τυχαίου αυτού αριθμού Z στην τιμή της συνάρτησης. Οι συνθήκες τερματισμού που τέθηκαν ήταν το κριτήριο σύγκλισης

$$\frac{|f(x_{n+1}) - f(x_1)|}{|f(x_{n+1})| + |f(x_1)|} < \epsilon/2$$

όπου ϵ μικρός θετικός αριθμός, ο οποίος εκφράζει τη μέγιστη επιτρεπόμενη ανοχή ως προς τη σχετική απόσταση μεταξύ των τιμών της συνάρτησης της καλύτερης και χειρότερης τρέχουσας λύσης. Αναλυτική περιγραφή του παραπάνω σχήματος υπάρχει στην εργασία [18]. Το κριτήριο αυτό αφορά πολύ μικρές περιοχές αναζήτησης. Άλλη μια συνθήκη τερματισμού που τέθηκε ήταν το μέγιστο πλήθος γενιών, GEN_{max}. Οι υπόλοιπες αλγοριθμικές παράμετροι εισόδου ήταν οι συχνότητες διασταύρωσης p_c και μετάλλαξης p_m.

Στην συνέχεια δινουμε αναλυτικά τα βήματα της μεθόδου

- Γεννάται ένας αρχικός πληθυσμό pop_0, αποτελούμενος από m σημεία ομοιόμορφα κατανεμημένα μέσα στον χώρο εφικτών λύσεων D.
- Υπολογίζεται η συνάρτηση καταλληλότητας $f(x)$ για κάθε άτομο στον πληθυσμό.
- Εφαρμόζεται ένας τελεστής ανασυνδυασμού με συχνότητα $p_c = 0.6 \sim p_c = 1$

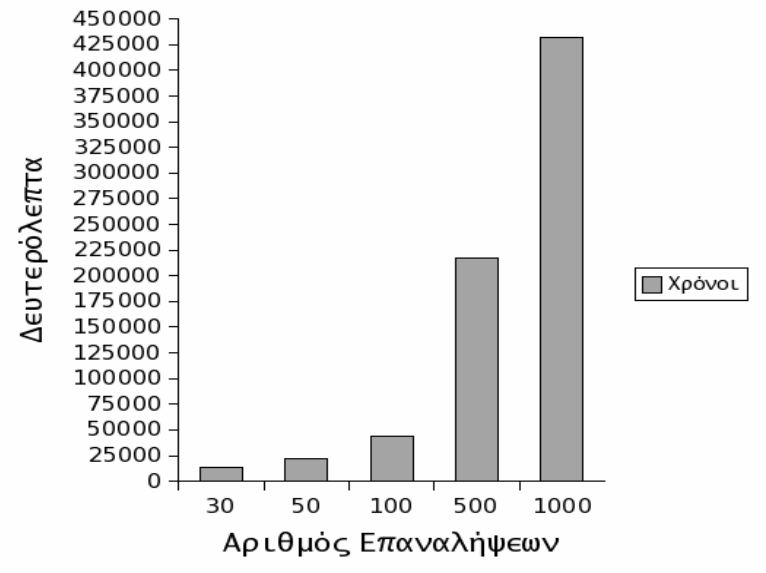
	Ονομασία	30	50	100	500	1000
$F1$	Sphere model	40	41	52	65	67
$F2$	Rosenbrock	33	38	48	60	65
$F3$	Step	22	26	46	64	62
$F4$	Quatric	21	26	46	58	65
$F5$	Foxholes	20	25	46	58	61
$F6$	Rastrigin	21	21	32	40	35
$F7$	Schwefel	21	23	28	46	36
$F8$	Griewang	21	22	23	30	37
$F9$	Watson	20	21	39	60	23
$F10$	Extended Rosenbrock	22	23	24	60	68
$F11$	Penalty II	20	21	26	53	64
$F12$	Powel badly scaled	21	24	37	53	80
$F13$	Gulf research and development	20	24	37	53	68
$F14$	Extended Powel singular	20	25	25	63	53
	Μέση Αποτελεσματικότητα	22.93	25.64	36.14	55.43	57.79

Πίνακας 4.3: Ποσοστά επιτυχίας EA (Επαναλήψεις 30,50,100,500,1000)

- Εφαρμόζεται ένας τελεστής μετάλλαξης με συχνότητα $p_m = (0, ..., 0.2)$ με βήμα 0.005.

- Ελέγξαμε και τα δύο μοντέλα αντικατάστασης του πληθυσμού. Το μοντέλο ολικής αντικατάστασης και το μοντέλο σταθερής αντικατάστασης. Πιο συγκεκριμένα στη πρώτη περίπτωση (μοντέλο γενεαλογικής αντικατάστασης) σε κάθε γενιά ο πληθυσμός αντικαθίσταται ολόκληρος ενώ στη δεύτερη περίπτωση μόνο ένα μέρος αυτού και πιο συγκεκριμένα στον αρχικό αλγόριθμο αντικατάστασης σταθερής κατάστασης ο αριθμός αυτών που δημιουργούνταν κάθε φορά ήταν λ=1 ή 2 ενώ ο αριθμός των ατόμων που εισηγοντο στον νέο πληθυσμό καθορίζετο από την συνάρτηση Q=pop(t) (όπου Q δεν μπορεί να είναι το κενό σύνολο ενώ το pop(t) αναφέρεται στον τρέχοντα πληθυσμό)

- Βάσει ενός τελεστή επιλογής, διαμορφώνεται η επόμενη γενιά $pop(t+1)$, η οποία περιέχει

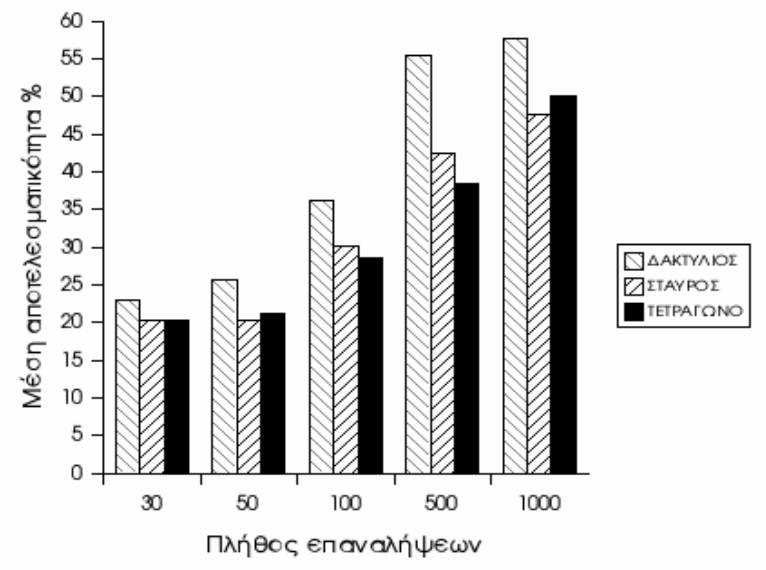


Σχήμα 4.7: Μέσοι Χρόνοι σε δευτερόλεπτα για ΕΑ

τα καλύτερα μέλη της προηγούμενης γενιάς.

- Εφόσον δεν ικανοποιούνται τα κριτήρια τερματισμού του αλγορίθμου,όπως η σύγκλιση του αλγορίθμου, η οποία ελέγχεται μέσω της τυπικής απόκλισης του δείγματος τιμών της αντικειμενικής συνάρτησης, το συνολικό πλήθος δοκιμών κλπ. Αν κανένα από τα κριτήρια δεν ικανοποιείται τότε επαναλαμβάνεται η διαδικασία εξέλιξης από το βήμα 2.

Είναι φανερό ότι η συμπεριφορά του απλού εξελικτικού αλγορίθμου δεν μπορεί να χαρακτηριστεί ικανοποιητική, ούτε ως προς την αποτελεσματικότητα ούτε (κυρίως) ως προς την αποδοτικότητα. Χαρακτηριστικό είναι το γεγονός της καθολικής αποτυχίας εντοπισμού του ολικού ακρότατου της 10-διάστατης Rozenbrock όπως και άλλων προβλημάτων του πίνακα 4.2. Τα αποτελέσματα των δοκιμών, για διαφορετικό αριθμό επαναλήψεων, συνοψίζονται στον Πίνακα



Σχήμα 4.8: Μέση αποτελεσματικότητα για τρία διαφορετικά είδη γειτονιάς ΕΑ

4.3. Στο σχήμα 4.7 βλέπουμε τους χρόνους σε δευτερόλεπτα που απαιτήθηκαν για διαφορετικό αριθμό επαναλήψεων. Στον πίνακα 4.4 έχουμε την βασική διάταξη ΔΜΑ που παρουσιάστηκε στην ενότητα 3.3.1 προσαρμοσμένη ανάλογως στην δομή ενός απλού ΕΑ, για κάθε μία από τις συναρτησεις μας. Στο σχήμα 4.8, δίνεται με την μορφή ραβδογράμματος η μέση αποτελεσματικότητα που βρέθηκε μέσα απο το σύνολο των πειραμάτων.

Η επίδραση των παραμέτρων εισόδου, δηλαδή του μεγέθους του πληθυσμού και των συχνοτήτων διασταύρωσης και μετάλλαξης, ήταν αρκετά σημαντική, και σε ορισμένες περιπτώσεις καθοριστική. Καθοριστική ήταν και η επίδραση της γειτονιάς όπως φαίνεται και στο σχήμα 4.8. Γενικά, αυξάνοντας το μέγεθος του πληθυσμού βελτιώθηκε η αξιοπιστία του αλγορίθμου, χωρίς

($F1$)	**ΔEA**	30	100	10	0	2	0.6	0.75	1	4	2	10	0.5	1	1	1	0	2
($F2$)	**ΔEA**	30	200	10	0	2	0.6	0.70	1	1	4	10	0.5	1	1	1	0	2
($F3$)	**ΔEA**	30	250	20	0	2	0.65	0.72	1	1	6	50	0.5	1	1	1	0	2
($F4$)	**ΔEA**	30	150	5	0	2	0.55	0.80	1	2	4	40	0.8	1	1	1	0	2
($F5$)	**ΔEA**	30	250	10	0	2	0.65	0.81	1	1	4	30	0.7	1	1	1	0	2
($F6$)	**ΔEA**	30	400	20	1	3	0.68	0.90	1	1	2	5	0.1	1	2	1	0	2
($F7$)	**ΔEA**	30	300	20	1	3	0.74	0.95	1	1	2	30	0.8	1	2	1	0	2
($F8$)	**ΔEA**	30	200	30	1	3	0.74	0.90	1	1	2	40	0.5	1	2	1	0	2
($F9$)	**ΔEA**	30	300	20	1	3	0.58	0.82	1	1	2	20	0.5	1	2	1	0	2
($F10$)	**ΔEA**	30	300	20	1	3	0.62	0.81	1	1	4	20	0.5	1	2	1	0	2
($F11$)	**ΔEA**	30	350	20	1	3	0.6	0.85	1	1	2	30	0.5	1	2	1	0	2
($F12$)	**ΔEA**	30	300	20	1	3	0.6	0.75	1	2	4	10	0.1	1	2	1	0	2
($F13$)	**ΔEA**	30	350	20	1	3	0.6	0.75	1	1	2	20	0.8	1	2	1	0	2
($F14$)	**ΔEA**	30	300	20	1	3	0.6	0.82	1	1	2	10	0.5	1	2	1	0	2

Πίνακας 4.4: Διατάξεις Εξελικτικού Αλγορίθμου

η βελτίωση αυτή να είναι τέτοια που να δικαιολογεί τη μεγάλη αύξηση του πλήθους των απαιτούμενων δοκιμών, με εξαίρεση τις περίπτωσεις των συναρτήσεων Griewank και Gulf research and development. Η αύξηση της τιμής της συχνότητας διασταύρωσης από 60% σε 100% δεν διαφοροποίησε τα αποτελέσματα, ενώ αντίθετα η αύξηση της συχνότητας μετάλλαξης από 0% σε 20% τα διαφοροποίησε σημαντικά, είτε προς τη θετική (όπως στην περίπτωση των συναρτήσεων Step, Quatric, Foxholes, Schefel, Rastrigin, Powel badly scaled) είτε προς την αρνητική κατεύθυνση (όπως στην περίπτωση της σφαιροειδούς συνάρτησης). Δηλαδή, αυξάνοντας την τυχαιότητα του αλγορίθμου αυξήθηκε η πιθανότητα εντοπισμού της ολικά βέλτιστης λύσης μόνο στην περίπτωση που η αντικειμενική συνάρτηση ήταν έντονα μη γραμμική, διαφορετικά μειώθηκε η αποτελεσματικότητα του αλγορίθμου. Η αύξηση τόσο της συχνότητας διασταύρωσης όσο και της συχνότητας μετάλλαξης είχε ως συνέπεια μεγαλύτερο υπολογιστικό φόρτο.

Γενετικός Αλγόριθμος

Τα βήματα του Γενετικού Αλγορίθμου μας υλοποιήθηκαν σε τρία στάδια:

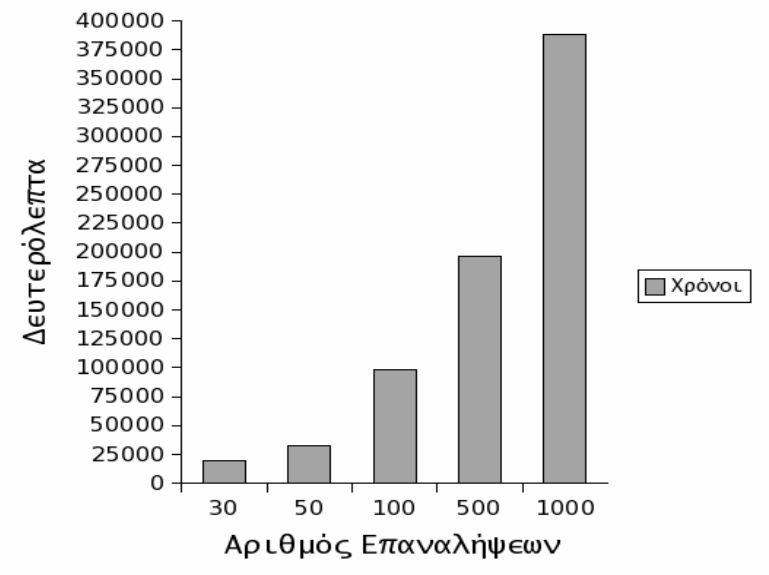
- Αρχικοποίηση
  - κωδικοποίηση των μεταβλητών σε γονίδια και έπειτα χρωμοσώματα.
  - αρχικοποίηση του πληθυσμού με την δημιουργία της πρώτης γενιάς.
  - εφαρμογή της συνάρτησης κόστους σε κάθε χρωμόσωμα.
- Αναπαραγωγή
  - αναπαραγωγή μέσω της επιλογής με βάση την τιμή της συνάρτησης κόστους του κάθε ατόμου.
  - εφαρμογή ανασυνδυασμού και μετάλλαξης που θα δώσει την επόμενη γενιά
- Αντικατάσταση γενιάς
  - αντικατάσταση της προηγούμενης γενιάς με την νέα
  - εφαρμογή της συνάρτησης κόστους σε κάθε χρωμόσωμα
  - έλεγχος του κριτηρίου τερματισμού

  Οι συνθήκες τερματισμού που τέθηκαν ήταν το κριτήριο σύγκλισης που χρησιμοποιηθηκε και στον εξελικτικό αλγόριθμο της προηγούμενης παραγράφου.Οι υπόλοιπες αλγοριθμικές παράμετροι εισόδου ήταν οι συχνότητες διασταύρωσης p_c και μετάλλαξης p_m. Ελέγξαμε και τα δύο μοντέλα αντικατάστασης του πληθυσμού. Το μοντέλο ολικής αντικατάστασης και το μοντέλο σταθερής αντικατάστασης.

	Ονομασία	30	50	100	500	1000
$F1$	Sphere model	44	48	54	62	63
$F2$	Rosenbrock	36	40	47	59	67
$F3$	Step	24	29	48	62	62
$F4$	Quatric	22	29	49	57	63
$F5$	Foxholes	23	27	50	55	61
$F6$	Rastrigin	24	24	27	29	25
$F7$	Schwefel	26	26	38	42	36
$F8$	Griewang	25	25	29	33	32
$F9$	Watson	23	27	35	63	23
$F10$	Extended Rosenbrock	24	29	34	62	68
$F11$	Penalty II	25	28	36	58	64
$F12$	Powel badly scaled	22	24	39	56	70
$F13$	Gulf research and development	23	28	39	57	59
$F14$	Extended Powel singular	24	27	35	60	53
	Μέση Αποτελεσματικότητα	26.07	29.36	40	53.92	53.28

Πίνακας 4.5: Ποσοστά επιτυχίας GA (Επαναλήψεις 30,50,100,500,1000)

Είναι φανερό ότι και η συμπεριφορά του γενετικού αλγορίθμου δεν μπορεί να χαρακτηριστεί ικανοποιητική. Χαρακτηριστικό είναι το γεγονός της καθολικής αποτυχίας εντοπισμού αρκετών προβλημάτων του πίνακα 4.2. Τα αποτελέσματα των δοκιμών, για διαφορετικό αριθμό επαναλήψεων, συνοψίζονται στον Πίνακα 4.5.Στο σχήμα 4.9 βλέπουμε τους χρόνους σε δευτερόλεπτα που απαιτήθηκαν για διαφορετικό αριθμό επαναλήψεων. Στον πίνακα 4.6 έχουμε την βασική διάταξη ΔΜΑ που παρουσιάστηκε στην ενότητα 3.3.1 προσαρμοσμένη ανάλογως στην δομή ενός απλού GA, για κάθε μία από τις συναρτησεις μας. Στο σχήμα 4.10, δίνεται με την μορφή ραβδογράμματος η μέση αποτελεσματικότητα που βρέθηκε μέσα απο το σύνολο των πειραμάτων με βάση την δομή γειτονιάς του πληθυσμού. Στον άξονα των y έχουμε τη μέση αποτελεσματικότητα σε ποσοστά % και

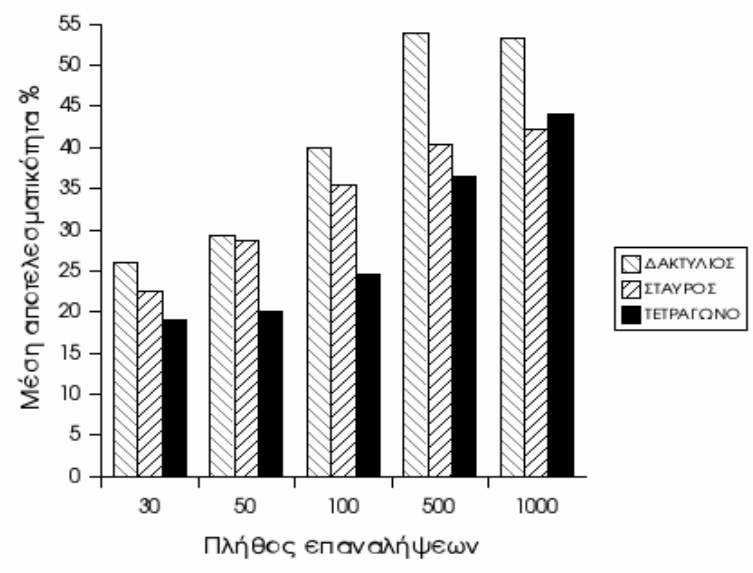


Σχήμα 4.9: Μέσοι Χρόνοι σε δευτερόλεπτα για GA

στον άξονα των x έχουμε τον αριθμό των επαναλήψεων.

Γενικά, αυξάνοντας το μέγεθος του πληθυσμού βελτιώθηκε η αξιοπιστία του αλγορίθμου, χωρίς η βελτίωση αυτή να είναι τέτοια που να δικαιολογεί τη μεγάλη αύξηση του πλήθους των απαιτούμενων δοκιμών. Η αύξηση της τιμής της συχνότητας διασταύρωσης από 60% σε 100% όπως και η αύξηση της συχνότητας μετάλλαξης από 0% σε 20% διαφοροποίησε προς το καλύτερο τα αποτελέσματα.

Η επίδραση των παραμέτρων εισόδου, δηλαδή του μεγέθους του πληθυσμού και των συχνοτήτων διασταύρωσης και μετάλλαξης, ήταν αρκετά σημαντική, και σε ορισμένες περιπτώσεις καθοριστική. Τα ποσοστά ανασυνδυασμου τάξης $70\% \sim 75\ \%$ βοηθούν στην αποφυγή της πρόωρης σύγκλισης του πληθυσμού, δηλαδή του μοντέλου των GA. Δίνεται έτσι η δυνατότητα για περαιτέρω διερεύνηση διαφορετικών περιοχών του χώρου



Σχήμα 4.10: Μέση αποτελεσματικότητα για τρία διαφορετικά είδη γειτονιάς GA

λύσεων και δημιουργίας γενετικής πληροφορίας η οποία καθώς εισάγεται στον πληθυσμό βοηθά στον να προχωρήσει η αναζήτηση του ολικού βέλτιστου. Το τελικό συμπέρασμα που προκύπτει για αυτή τη παράμετρο είναι ότι πολύ μικρά ποσοστά ανασυνδυασμού των γονέων με μικρά μεγέθη πληθυσμού υποβαθμίζουν και άλλο την απόδοση των GA. Η χρήση μεγάλων τιμών για το μέγεθος πληθυσμού δεν συνεπάγεται σε όλες τις περιπτώσεις καλύτερα αποτελέσματα και εκτός τούτου έχει σαν αποτέλεσμα την αύξηση του υπολογιστικού χρόνου. Αρχικά η απόδοση των μοντέλων με μέγεθος πληθυσμού 100 είναι καλυτερη από την απόδοση άλλων μεγεθών πληθυσμού, αλλά συνήθως καθηλώνεται ο αλγόριθμος σε τοπικά ελάχιστα. Μια μεγάλη τιμή για το μέγεθος πληθυσμού μπορεί να χρησιμοποιηθεί για να βελτιώσει την απόδοση off - line, δηλαδή όταν υπάρχει

$(F1)$	**ΔΓΑ**	30	100	10	0	2	0.6	0.75	1	4	2	10	0.5	1	1	1	0	2
$(F2)$	**ΔΓΑ**	30	200	10	0	2	0.6	0.70	1	1	4	10	0.5	1	1	1	0	2
$(F3)$	**ΔΓΑ**	30	250	20	0	2	0.65	0.72	1	1	6	50	0.5	1	1	1	0	2
$(F4)$	**ΔΓΑ**	30	150	5	0	2	0.55	0.80	1	2	4	40	0.8	1	1	1	0	2
$(F5)$	**ΔΓΑ**	30	250	10	0	2	0.65	0.81	1	1	4	30	0.7	1	1	1	0	2
$(F6)$	**ΔΓΑ**	30	400	20	1	3	0.68	0.90	1	1	2	5	0.1	1	2	1	0	2
$(F7)$	**ΔΓΑ**	30	300	20	1	3	0.74	0.95	1	1	2	30	0.8	1	2	1	0	2
$(F8)$	**ΔΓΑ**	30	200	30	1	3	0.74	0.90	1	1	2	40	0.5	1	2	1	0	2
$(F9)$	**ΔΓΑ**	30	300	20	1	3	0.58	0.82	1	1	2	20	0.5	1	2	1	0	2
$(F10)$	**ΔΓΑ**	30	300	20	1	3	0.62	0.81	1	1	4	20	0.5	1	2	1	0	2
$(F11)$	**ΔΓΑ**	30	350	20	1	3	0.6	0.85	1	1	2	30	0.5	1	2	1	0	2
$(F12)$	**ΔΓΑ**	30	300	20	1	3	0.6	0.75	1	2	4	10	0.1	1	2	1	0	2
$(F13)$	**ΔΓΑ**	30	350	20	1	3	0.6	0.75	1	1	2	20	0.8	1	2	1	0	2
$(F14)$	**ΔΓΑ**	30	300	20	1	3	0.6	0.82	1	1	2	10	0.5	1	2	1	0	2

Πίνακας 4.6: Διατάξεις Γενετικού Αλγορίθμου

διαθέσιμος χρόνος, ενώ μια μικρή τιμή απαιτείται όταν ο στόχος είναι η καλή απόδοση on line, δηλαδή όταν τα αποτελέσματα απαιτούνται σε σύντομο χρόνο. Καθοριστική ήταν και η επίδραση της γειτονιάς όπως φαίνεται και στο σχήμα 4.10. Δηλαδή, αυξάνοντας την τυχαιότητα του αλγορίθμου αυξήθηκε η πιθανότητα εντοπισμού της ολικά βέλτιστης λύσης μόνο στην περίπτωση που η αντικειμενική συνάρτηση ήταν έντονα μη γραμμική, διαφορετικά μειώθηκε η αποτελεσματικότητα του αλγορίθμου. Η αύξηση τόσο της συχνότητας διασταύρωσης όσο και της συχνότητας μετάλλαξης είχε ως συνέπεια μεγαλύτερο υπολογιστικό φόρτο.

Μιμητικός Αλγόριθμος με Αποτρεπτική Αναζήτηση

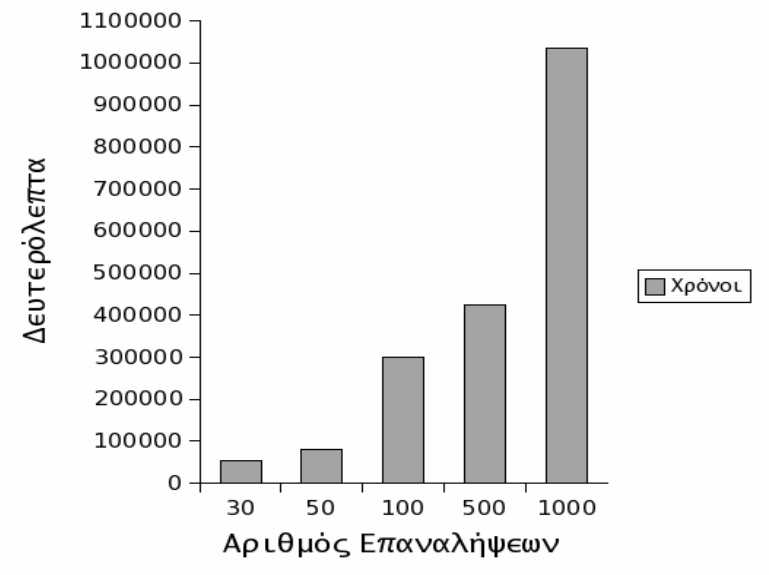
Η μελέτη αυτής της μεθοδου έγινε διαφοροποιώντας το μέγεθος του πληθυσμού, το οποίο ορίζεται βάσει του αριθμού των πληθυσμιακών ομάδων p και του μεγέθους κάθε ομάδας m . Στις υπόλοιπες παραμέτρους εισόδου του αλγορίθμου τέθηκαν οι τυπικές τιμές που προτείνουν ο Moscato και Norman [167], δηλαδή $q = n + 1$ (αριθμός γονέων που συμμετέχουν στην εξέλιξη κάθε ομάδας), $\alpha = 1$ (αριθμός βημάτων εξέλιξης πριν την ανάμιξη των ομάδων) και $\beta = m$ (ελάχιστο μέγεθος πληθυσμού κάθε ομάδας). Για τον τερματισμό του αλγορίθμου ελέγχονταν οι ακόλουθες συνθήκες [158, 146, 145]:

1. αν το πλήθος των δοκιμών έχει ξεπεράσει μια ανώτατη τιμή, ή
2. αν η ποσοστιαία βελτίωση της τρέχουσας βέλτιστης λύσης στον πληθυσμό μεταξύ 5 διαδοχικών κύκλων εξέλιξης είναι μικρότερη από μια ελάχιστη τιμή, ή
3. αν όλες οι λύσεις έχουν συγκλίνει σε μια μικρή περιοχή του εφικτού χώρου.

Στον πίνακα 4.8 έχουμε την βασική διάταξη ΔΜΑ για κάθε μία από τις συναρτησεις μας. Στο σχήμα 4.12, δίνεται με την μορφή ραβδογράμματος η μέση αποτελεσματικότητα που βρέθηκε μέσα απο το σύνολο των πειραμάτων.

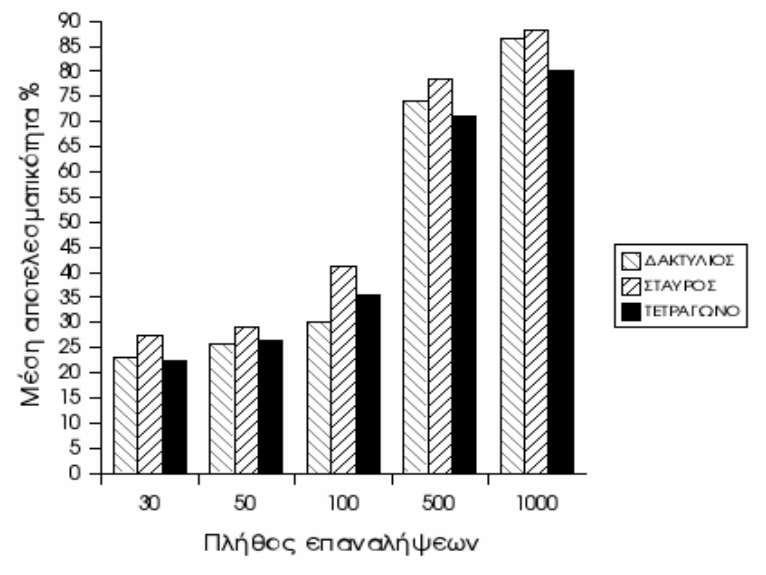
Τα βήματα που ακολουθήθηκαν στην μέθοδο αυτή ήταν:

- Γεννάται ένας αρχικός πληθυσμό pop_0, αποτελούμενος από $m > n + 1$ σημεία ομοιόμορφα κατανεμημένα μέσα στο χώρο των εφικτών λύσεων.
- Υπολογίζουμε τη συνάρτηση καταλληλότητας $f(m)$ για κάθε άτομο σε ένα πληθυσμό.



Σχήμα 4.11: Μέσοι Χρόνοι σε δευτερόλεπτα για MA TS

- Τα σημεία ταξινομούνται κατά αύξουσα τιμή της αντικειμενικής συνάρτησης και αποθηκεύονται σε ένα διάνυσμα $D = (m_i, f_i)$.
- Τα στοιχεία του D χωρίζονται σε p πληθυσμούς $pop^{[1]}, pop^{[2]}, ..., , pop^{[p]}$.
- Για κάθε ομάδα $pop^{[1...p]}$ εφαρμόζεται η διαδικασία της αναπαραγωγής.
- Σε κάθε ομάδα ξεκινάμε από ένα σύνολο λύσεων $m^*_{0...t}$ και παράγουμε $z = 3$ σύνολα γειτονικών λύσεων $M_{0...3}$ και το καλύτερο ποιοτικά εξ αυτών αποτελεί το νέα σύνολο υποψηφιων λύσεων s^*_1, γύρω από το οποίο γεννώνται οι επόμενες γειτονικές λύσεις.
- Μετά την αντικατάσταση του s από το $s+1$ το s μπαίνει στην απαγορευμένη λίστα.
- Μόλις εξαντληθει η χωρητικότητα της απαγορευμένης λίστας, αφαιρούμε το πρώτο



Σχήμα 4.12: Μέση αποτελεσματικότητα για τρία διαφορετικά είδη γειτονιάς MA TS

σύνολο λύσεων το οποίο είχε αποθηκευτεί πρώτο.

– Οι εξελιγμένες ομάδες $pop^{[1]}, pop^{[2]}, pop^{[3]}, ..., pop^{[p]}$ επανατοποθετούνται στο διάνυσμα D. Έτσι επιλέγεται η πλέον υποσχόμενη περιοχή, η περιοχή δηλαδή εκείνη στην οποία υπάρχει αυξημένη πιθανότητα να κείται η βέλτιστη λύση. Αυτη η περιοχή γίνεται ο νέος χώρος εφικτών λύσεων.

– Στην συνέχεια εφαρμοζουμε ένα τελεστή ανασυνδυασμού. Η συχνότητα ανασυνδυασμού είναι p_c=0.6 για μέγεθος πληθυσμού μεγαλύτερο του 100 και p_c=0.9 για μέγεθος πληθυσμού μικρότερο του 30.

– Μέσω ενός τελεστή μετάλλαξης ορισμένα άτομα του νέου χώρου εφικτών λύσεων αλλάζουν τιμή από 0 σε 1. Αντί για ένα συνολικό ποσοστό μετάλλαξης, μπορούν να

	Ονομασία	30	50	100	500	1000
$F1$	Sphere model	42	43	55	65	80
$F2$	Rosenbrock	40	39	49	64	80
$F3$	Step	32	28	54	68	90
$F4$	Quatric	24	30	53	59	78
$F5$	Foxholes	27	31	53	62	61
$F6$	Rastrigin	22	28	42	61	79
$F7$	Schwefel	25	23	32	68	77
$F8$	Griewang	23	29	30	76	30
$F9$	Watson	23	28	42	65	66
$F10$	Extended Rosenbrock	29	56	55	88	95
$F11$	Penalty II	21	22	34	66	53
$F12$	Powel badly scaled	31	25	44	78	53
$F13$	Gulf research and development	20	29	40	78	58
$F14$	Extended Powel singular	27	28	26	73	66
	Μέση Αποτελεσματικότητα	27,57	29,21	41,36	78,64	88,29

Πίνακας 4.7: Ποσοστά επιτυχίας MA TS (Επαναλήψεις 30,50,100,500,1000)

διατηρηθούν πιθανότητες μετάλλαξης για κάθε μεταβλητή κάθε ατόμου. Έτσι κάθε μεταβλητή μπορεί να έχει διαφορετική πιθανότητα μετάλλαξης. Αυτή η πιθανότητα μετάλλαξης μπορεί να κωδικοποιηθεί σε κάθε άτομο σαν επιπλέον πληροφορία και να εξελιχθεί μαζί με το άτομο. Έτσι επιτυγχάνεται η αυτο-προσαρμογή των παραμέτρων μετάλλαξης, ταυτόχρονα με την διερεύνηση του χώρου. Οι απόγονοι που προκύπτουν αντικαθιστούν τα χειρότερα άτομα του πληθυσμού αν η ποιότητα τους είναι τουλάχιστον τόσο καλή όσο του χειρότερου ατόμου του πληθυσμού (εξάλειψη του χειρότερου - (elimination of the worst)) .

– Ελέγχεται αν ο τρέχον πληθυσμός ικανοποιεί ένα τουλάχιστον από τα κριτήρια τερματισμού, δηλαδή η σύγκλιση του αλγορίθμου και το συνολικό πλήθος δοκιμών.

$F1$)	**ΔMA**	30	100	10	0	2	0.68	0.75	1	4	2	50	0.8	1	1	1	2	2
($F2$)	**ΔMA**	30	200	10	0	2	0.74	0.70	1	1	4	50	0.8	1	1	1	2	2
($F3$)	**ΔMA**	30	250	20	0	2	0.68	0.90	1	1	6	50	0.8	1	1	1	2	2
($F4$)	**ΔMA**	30	150	5	0	2	0.74	0.95	1	2	4	50	0.8	1	1	1	2	2
($F5$)	**ΔMA**	30	250	10	0	2	0.74	0.90	1	1	4	50	0.8	1	1	1	2	2
($F6$)	**ΔMA**	30	250	20	1	3	0.68	0.90	1	1	2	100	0.8	1	2	1	2	2
($F7$)	**ΔMA**	30	250	20	1	3	0.55	0.95	1	1	2	100	0.8	1	2	1	2	2
($F8$)	**ΔMA**	30	200	30	1	3	0.74	0.90	1	1	2	100	0.6	1	2	1	2	2
($F9$)	**ΔMA**	30	250	20	1	3	0.58	0.82	1	1	2	100	0.8	1	2	1	2	2
($F10$)	**ΔMA**	30	250	20	1	3	0.62	0.81	1	1	4	100	0.6	1	2	1	2	2
($F11$)	**ΔMA**	30	250	20	1	3	0.55	0.85	1	1	2	100	0.7	1	2	1	2	2
($F12$)	**ΔMA**	30	250	20	1	3	0.6	0.86	1	2	4	100	0.8	1	2	1	2	2
($F13$)	**ΔMA**	30	250	20	1	3	0.56	0.82	1	1	2	100	0.8	1	2	1	2	2
($F14$)	**ΔMA**	30	250	20	1	3	0.6	0.82	1	1	2	100	0.7	1	2	1	2	2

Πίνακας 4.8: ΔMA με TS

Αν κανένα από τα κριτήρια δεν ικανοποιείται, τότε ο αλγόριθμος επαναλαμβάνεται από το βήμα 4.

Ως μέτρο σύγκλισης του πληθυσμού ορίστηκε η κανονικοποιημένη ποσότητα:

$$exp\{1/n \sum_{j=1}^{n} log\langle\langle max(x_{1j}, x_{2j}, ..., x_{sj}) - min(x_{1j}, x_{2j}, ..., x_{sj})/x_j^{max} - x_j^{min}\rangle + \epsilon)\rangle\}$$

η οποία εκφράζει τη σχετική απόσταση των ακραίων λύσεων στον εκάστοτε πληθυσμό ως προς το εύρος του εφικτού χώρου. Ο αριθμός ε εκφράζει μια πολύ μικρή θετική ποσότητα, με την οποία εξασφαλίζεται ότι η λογαριθμική έκφραση δεν γίνεται ποτέ μηδέν. Τα αποτελέσματα της μεθοδου αυτής συνοψίζονται στον Πίνακα 4.7 για μέσο πλήθος λύσεων που διερευνήθηκαν $m = 1 \sim m = 100$. Στο σχήμα 4.11 βλέπουμε τους χρόνους

σε δευτερόλεπτα που απαιτήθηκαν για διαφορετικό αριθμό επαναλήψεων.

Από τα παραπάνω αποδεικνύεται ότι ο ΜΑ με αποτρεπτική αναζήτηση αντιμετώπισε με σχετική επιτυχία ορισμένα από τα προβλήματα, ενώ απέτυχε εντελώς σε άλλα. Η αποτελεσματικότητα της μεθόδου παρουσίασε βελτίωση με αύξηση του πλήθους των εκκινήσεων, χωρίς ωστόσο η βελτίωση αυτή να είναι ομοιόμορφη. Ο αλγόριθμος αυτός πραγματοποιεί αρχικά μια αδρή και στη συνέχεια μια πιο λεπτομερή διερεύνηση του εφικτού χώρου, Προφανώς, απαιτήθηκε η θεώρηση μεγαλύτερου πληθυσμού όσο αυξανόταν ο βαθμός δυσκολίας του προβλήματος. Έτσι, ενώ ήταν επαρκής μία και μόνο ομάδα για την εύρεση του ελαχίστου της διδιάστατης Rozenbrock και της Extended Rozenbrock, απαιτήθηκαν τέσσερις ομάδες για την επίλυση των ίδιων προβλήματων στις 10 διαστάσεις, ενώ οι 4 ομάδες δεν ήταν επαρκείς για την επίλυση των προβλήματων Penalty II, Rastrigin, Powel badly scaled με ικανοποιητική αξιοπιστία. Η ταχύτητα της μεθόδου παρουσίασε σημαντικές διαφορές, ανάλογα με το βαθμό δυσκολίας του εκάστοτε προβλήματος και ήταν προφανώς ανάλογη του πλήθους των εκκινήσεων. Έτσι, ο εντοπισμός του ολικού ακρότατου των πιο εύκολων συναρτήσεων ελέγχου, όπως της 10-διάστατης σφαιροειδούς, απαίτησε μικρό αριθμό δοκιμών, σε αντίθεση με πολυπλοκότερες συναρτήσεις, όπως για παράδειγμα η Rastrigin και η διδιάστατη Rozenbrock, για τις οποίες χρειάστηκαν μία ως δύο τάξεις μεγέθους περισσότερες δοκιμές. Ως προς το μέγεθος κάθε ομάδας είναι φανερό ότι υιοθετώντας την τιμή $m = 2n + 1$, την οποία προτείνουν και οι δημιουργοί του αλγορίθμου, προέκυψαν καλύτερα αποτελέσματα σε σχέση με την ελάχιστη δυνατή τιμή $m = n+ 1$.

Όσον αφορά τις συνάρτησεις Griewang και τη Watson, το υπερβολικά μεγάλο πλήθος

δοκιμών είχε ως αίτιο την αδυναμία διαφυγής του αλγορίθμου από κορυφογραμμές. Μια αντίστοιχη αδυναμία διαπιστώθηκε συγκρίνοντας τη συμπεριφορά του αλγορίθμου στις δύο συναρτήσεις Rozenbrock και Extended Rozenbrock. Στην περίπτωση της δυδιάστατης Rozenbrock, ο αλγοριθμος κατόρθωσε, έστω και με υπερβολικά μεγάλο αριθμό βημάτων, να προχωρήσει κατά μήκος της παραβολοειδούς χαράδρας και να συγκλίνει κοντά στο σημείο ελαχίστου. Αυξάνοντας τη διάσταση του προβλήματος, το αλγόριθμος απέτυχε να κάνει κάτι τέτοιο, καταλήγοντας έτσι (και μάλιστα με σχετικά μεγάλη ταχύτητα) πολύ μακριά από το ακρότατο της συνάρτησης. Η καλύτερη απόδοση της μεθόδου του ΜΑ με αποτρεπτική αναζήτηση, λαμβάνοντας υπόψη και το βαθμό δυσκολίας του προβλήματος, παρατηρήθηκε στην περίπτωση της 10-διάστατης συνάρτησης Griewank. Αντίθετα, τόσο στην περίπτωση της συνάρτησης Watson όσο και της βηματικής συνάρτησης, ο αλγόριθμος έδωσε πολύ φτωχά αποτελέσματα, παρόλο που διερεύνησε εξαιρετικά μεγάλο αριθμό λύσεων.

Μιμητικός Αλγόριθμος με Προσομοιούμενη Ανόπτηση

Στον ΜΑ με προσομοιούμενη Ανόπτηση τα βήματα που εφαρμόσαμε ήταν τα παρακάτω:

- Γεννάται ένας αρχικός πληθυσμό pop_0, αποτελούμενος από $m > n + 1$ σημεία ομοιόμορφα κατανεμημένα μέσα στο ήμισυ του χώρου των εφικτών λύσεων.
- Υπολογίζουμε τη συνάρτηση καταλληλότητας $f(m)$ για κάθε άτομο σε ένα πληθυσμό.

	Ονομασία	30	50	100	500	1000
$F1$	Sphere model	54	51	69	78	71
$F2$	Rosenbrock	50	44	63	72	78
$F3$	Step	30	35	54	82	78
$F4$	Quatric	36	33	62	77	76
$F5$	Foxholes	30	41	60	71	71
$F6$	Rastrigin	36	40	51	47	56
$F7$	Schwefel	34	33	42	51	60
$F8$	Griewang	30	36	39	39	37
$F9$	Watson	28	39	47	66	75
$F10$	Extended Rosenbrock	37	42	32	70	68
$F11$	Penalty II	29	28	39	71	71
$F12$	Powel badly scaled	33	31	55	72	71
$F13$	Gulf research and development	36	38	55	62	64
$F14$	Extended Powel singular	34	38	34	68	74
	Μέση Αποτελεσματικότητα	35.5	37.79	50.14	66.14	67.86

Πίνακας 4.9: Ποσοστά επιτυχίας MA με SA (Επαναλήψεις 30,50,100,500,1000)

– Επιλέγονται με κάποια μέθοδο επιλογής τα σημεία με την καλύτερη και χειρότερη τιμή της αντικειμενικής συνάρτησης και ορίζεται η αρχική θερμοκρασία: $T^0 = f^0_{max} - f^0_{min}$. Τα σημεία ταξινομούνται κατά αύξουσα τιμή της αντικειμενικής συνάρτησης και αποθηκεύονται σε ένα διάνυσμα $D = \{m_i, f_i\}$.

– Τα στοιχεία του D χωρίζονται σε p ομάδες $pop^{[1]}, pop^{[2]}, ..., pop^{[p]}$, κάθε μία από τις οποίες περιέχει o σημεία, έτσι ώστε $m^k_i = m_{k+pop_{i-1}}$ με $k = 1, 2, ..., p$ και $i = 1, 2, ..., o$.

– Τίθεται $k = 1$ σηματοδοτώντας την έναρξη νέου κύκλου θερμικής ισορροπίας.

– Προσδιορίζονται το καλύτερο και το χειρότερο σημείο του $pop^{[k]}$, $f^{[m]}_{min}$ και $f^{[m]}_{max}$ αντίστοιχα.

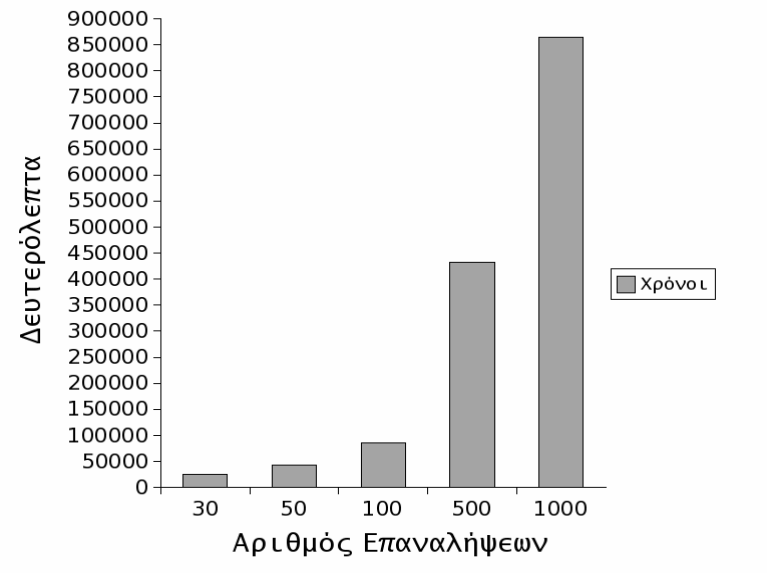
- Ελέγχεται αν η τρέχουσα θερμοκρασία του συστήματος ικανοποιεί τη συνθήκη : $T^m < \xi[f^m_{max} - f^m_{min}]$ όπου ξ παράμετρος του χρονοδιαγράμματος ανόπτησης.
- Από το σύνολο των σημείων αυτών $m_1, m_2, ..., m_p$ επιλέγεται μια κορυφή w βάσει του κριτηρίου Metropolis, η οποία και θεωρείται ως χειρότερη.
- Εφόσον $k < k_{max}$, τίθεται $k \rightarrow k+1$ και επιστρέφουμε στο βήμα 6.
- Εφόσον η τρέχουσα θερμοκρασία T είναι μεγαλύτερη από το ελάχιστο όριο T_{min}, μειώνεται με βάση την εξίσωση $T^{k+1} = \lambda T^k$ όπου λ συντελεστής που λαμβάνει τιμές στο διάστημα (0,1).
- Για κάθε ομάδα $pop^{[1]}, pop^{[2]}, pop^{[3]}, ..., pop^{[p]}$ εφαρμόζεται ενας τελεστής ανασυνδυασμού και ένας τελεστής μετάλλαξης με συχνότητα p_c=0.3 και p_m=0.05 αντίστοιχα.Η μετάλλαξη έχει σκοπό να εμποδίσει τον αλγόριθμο από το να παγιδευτεί μέσα σε τοπικά ακρότατα του προβλήματος.
- Τα εξελιγμένα δείγματα $pop^{[1]}, pop^{[2]}, pop^{[3]}, ..., pop^{[p]}$ επανατοποθετούνται στο διάνυσμα D και στη συνέχεια ταξινομείται το σύνολο των σημείων κατά αύξουσα τιμή της αντικειμενικής συνάρτησης.
- Στην συνέχεια εφαρμοζουμε ένα τελεστή ανασυνδυασμού στο νέο πληθυσμό. Η συχνότητα ανασυνδυασμού είναι p_c=0.6 για μέγεθος πληθυσμού μεγαλύτερο του 100 και p_c=0.9 για μέγεθος πληθυσμού μικρότερο του 30.
- Μέσω ενός τελεστή μετάλλαξης ορισμένα άτομα του νέου χώρου εφικτών λύσεων αλλάζουν τιμή από 0 σε 1. Η συχνότητα μετάλλαξης είναι ένας μικρός αριθμός p_m, ο οποίος λαμβάνει τιμές στο διάστημα $[0, ..., 0.2]$ με βήμα 0.005.

($F1$)	**ΔMA**	30	150	5	0	2	0.68	0.75	1	1	2	40	0.45	1	1	1	1	2
($F2$)	**ΔMA**	30	150	5	0	2	0.50	0.70	1	1	2	40	0.45	1	1	1	1	2
($F3$)	**ΔMA**	30	150	5	0	2	0.68	0.90	1	1	2	40	0.32	1	1	1	1	2
($F4$)	**ΔMA**	30	150	5	0	2	0.78	0.95	1	1	2	48	0.41	1	1	1	1	2
($F5$)	**ΔMA**	30	150	5	0	2	0.55	0.90	1	1	2	45	0.40	1	1	1	1	2
($F6$)	**ΔMA**	30	150	5	1	3	0.50	0.90	1	1	2	100	0.45	1	2	1	1	2
($F7$)	**ΔMA**	30	150	5	1	3	0.50	0.95	1	1	2	105	0.45	1	2	1	1	2
($F8$)	**ΔMA**	30	150	5	1	3	0.50	0.90	1	1	2	110	0.52	1	2	1	1	2
($F9$)	**ΔMA**	30	150	5	1	3	0.48	0.82	1	1	2	120	0.45	1	2	1	1	2
($F10$)	**ΔMA**	30	150	5	1	3	0.52	0.81	1	1	2	120	0.47	1	2	1	1	2
($F11$)	**ΔMA**	30	150	5	1	3	0.54	0.85	1	1	2	120	0.45	1	2	1	1	2
($F12$)	**ΔMA**	30	150	5	1	3	0.54	0.86	1	1	2	120	0.45	1	2	1	1	2
($F13$)	**ΔMA**	30	150	5	1	3	0.56	0.82	1	1	2	120	0.45	1	2	1	1	2
($F14$)	**ΔMA**	30	150	5	1	3	0.6	0.82	1	1	2	120	0.45	1	2	1	1	2

Πίνακας 4.10: ΔMA με SA

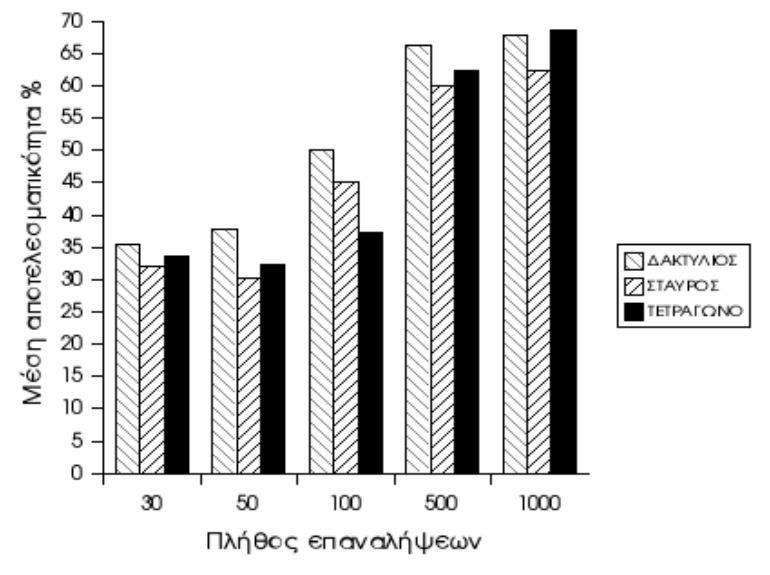
– Ελέγχεται αν ο τρέχον πληθυσμός ικανοποιεί ένα τουλάχιστον από τα κριτήρια τερματισμού, δηλαδή η σύγκλιση του αλγορίθμου και το συνολικό πλήθος δοκιμών. Αν κανένα από τα κριτήρια δεν ικανοποιείται, τότε ο αλγόριθμος επαναλαμβάνεται από το βήμα 6.

Τα αποτελέσματα των δοκιμών, για διάφορους συνδυασμούς παραμέτρων εισόδου, συνοψίζονται στον Πίνακα 4.9. Στον πίνακα 4.10 έχουμε την βασική διάταξη ΔMA που παρουσιάστηκε στην ενότητα 3.3.1 προσαρμοσμένη ανάλογως στην δομή ενός απλού ΕΑ, για κάθε μία από τις συναρτησεις μας. Στο σχήμα 4.14, δίνεται με την μορφή ραβδογράμματος η μέση αποτελεσματικότητα που βρέθηκε μέσα απο το σύνολο των πειραμάτων. Στον οριζόντιο άξονα x εχουμε το πλήθος των επαναλήψεων, στον άξονα z έχουμε τη δομές γειτονιάς του πληθυσμού και στον άξονα των y έχουμε τη μέση αποτελεσματικότα



Σχήμα 4.13: Μέσοι Χρόνοι σε δευτερόλεπτα για MA SA

σε ποσοστά %. Από τα αποτελέσματα φαινεται ότι η μεθοδος είναι πράγματι μια σχετικά καλή μέθοδος, αφού αντιμετώπισε με πολυ καλό ποσοστό σχεδόν τις περισσότερες από τις συναρτήσεις ελέγχου (με εξαίρεση την 10-διαστατη Extended Rozenbrock και την Gulf research and development), και μάλιστα με πολύ μικρότερο αριθμό δοκιμών σε σχέση με τον απλό εξελικτικό αλγόριθμο και τον μιμητικό αλγόριθμο με αποτρεπτική αναζήτηση. Συγκρίνοντας τα αποτελέσματα της μεθόδου του μιμητικού αλγορίθμου με προσομοιούμενη ανόπτηση με αυτά της μεθοδου της προσομοιουμενης ανόπτησης βλέπουμε ότι η πρώτη απαιτεί λιγότερες δοκιμές προκειμένου να επιτευχθεί μια ικανοποιητική προσέγγιση σε σχέση με την δεύτερη (σχήμα 4.21) Προφανώς, απαιτήθηκε η θεώρηση μεγαλύτερου πληθυσμού όσο αυξανόταν ο βαθμός δυσκολίας του προβλήματος. Η αύξηση της τιμής της συχνότητας διασταύρωσης από 60% σε 100% επαιξε σημαντικό



Σχήμα 4.14: Μέση αποτελεσματικότητα για τρία διαφορετικά είδη γειτονιάς MA SA

ρόλο στα αποτελέσματα, ενώ αντίθετα η αύξηση της συχνότητας μετάλλαξης από 1% σε 20% τα διαφοροποίησε όχι σημαντικά, πάντα όμως προς τη θετική κατεύθυνση. Δηλαδή, αυξάνοντας την τυχαιότητα του αλγορίθμου αυξήθηκε η πιθανότητα εντοπισμού της ολικά βέλτιστης λύσης μόνο στην περίπτωση που η αντικειμενική συνάρτηση ήταν έντονα μη γραμμική, διαφορετικά μειώθηκε η αποτελεσματικότητα του αλγορίθμου. Η αύξηση τόσο της συχνότητας διασταύρωσης όσο και της συχνότητας μετάλλαξης είχε ως συνέπεια μεγαλύτερο υπολογιστικό φόρτο. Η χρήση του ελάχιστο δυνατού μεγέθους πληθυσμού m = 2 + 1 σε 4 περιπτώσεις συναρτήσεων Sphere model, 2-Rozenbrock, 2-Extenden Rozenbrock, Griewang αποδειχτηκε επαρκής για την εξασφάλιση υψηλού

σχετικά ποσοστού επιτυχίας του αλγορίθμου. Είναι προφανές ότι με χρήση μικρου μεγέθους πληθυσμού απαιτείται μικρότερος αριθμός δοκιμών για τον εντοπισμό της βέλτιστης λυσης, και συνεπώς βελτιώνεται σημαντικά η αποδοτικότητα του αλγορίθμου.

Μιμητικός Αλγόριθμος με Καθοδηγουμενη Τοπική Αναζήτηση

Στον ΜΑ με Καθοδηγούμενη Τοπική Αναζήτηση τα βήματα που εφαρμόσαμε ήταν τα παρακάτω:

- Γεννάται ένας αρχικός πληθυσμό pop_0, αποτελούμενος από $m > n + 1$ σημεία ομοιόμορφα κατανεμημένα μέσα στο ήμισυ του χώρου των εφικτών λύσεων.
- Υπολογίζουμε τη συνάρτηση καταλληλότητας $f(m)$ για κάθε άτομο σε ένα πληθυσμό.
- Επιλέγονται με κάποια μεθοδο επιλογής τα σημεία με την καλύτερη και χειρότερη τιμή της αντικειμενικής συνάρτησης. Τα σημεία ταξινομούνται κατά αύξουσα τιμή της αντικειμενικής συνάρτησης και αποθηκεύονται σε ένα διάνυσμα $D = \{m_i, f_i\}$, τέτοιο ώστε το πρώτο στοιχείο του να αντιστοιχεί στο σημείο με την ελάχιστη τιμή της συνάρτησης.
- Τα στοιχεία του D χωρίζονται σε p ομάδες $pop^{[1]}, pop^{[2]}, ..., pop^{[p]}$, κάθε μία από τις οποίες περιέχει o σημεία, έτσι ώστε $m_i^k = m_{k+p_{i-1}}$ με $k = 1, 2, ..., p$ και $i = 1, 2, ..., o$.
- Για κάθε μια ομάδα $pop^{[1...p]}$ εφαρμόζουμε τον αλγόριθμο της κατευθυνόμενης τοπικής αναζήτησης GLS

- Για κάθε ομάδα $pop^{[1]}, pop^{[2]}, pop^{[3]}, ..., pop^{[p]}$ εφαρμόζεται ενας τελεστής ανασυνδυασμού και ένας τελεστής μετάλλαξης με συχνότητα p_c=0.2 και p_m=0.01 αντίστοιχα. Η μετάλλαξη έχει σκοπό να ενισχύσει τον αλγόριθμο στην προπάθεια να αποφευχθει ο εγκλωβισμός σε τοπικά ακρότατα του προβλήματος.

- Τα εξελιγμένα δείγματα $pop^{[1]}, pop^{[2]}, pop^{[3]}, ..., pop^{[p]}$ επανατοποθετούνται στο διάνυσμα D και στη συνέχεια ταξινομείται το σύνολο των σημείων κατά αύξουσα τιμή της αντικειμενικής συνάρτησης.

- Στην συνέχεια εφαρμοζουμε ένα τελεστή ανασυνδυασμού. Η συχνότητα ανασυνδυασμού είναι p_c=0.6 για μέγεθος πληθυσμού μεγαλύτερο του 100 και p_c=0.9 για μέγεθος πληθυσμού μικρότερο του 30.

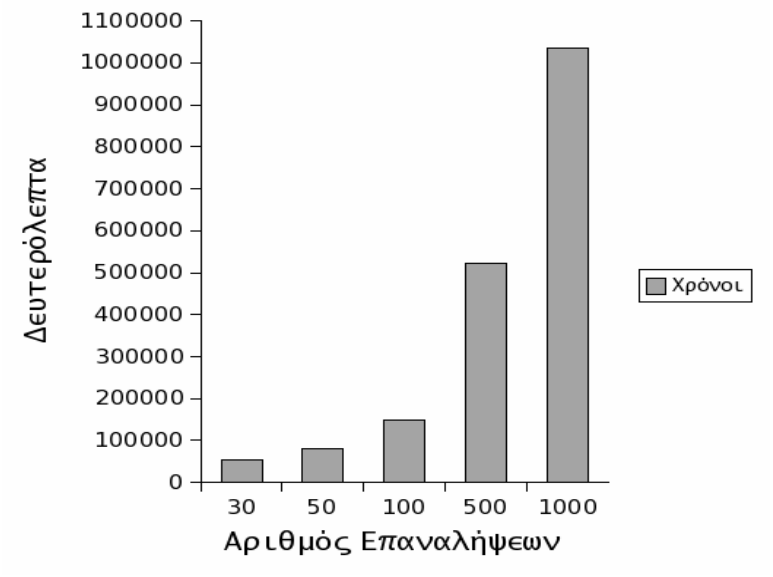
- Μέσω ενός τελεστή μετάλλαξης ορισμένα άτομα του νέου χώρου εφικτών λύσεων αλλάζουν τιμή από 0 σε 1. Η συχνότητα μετάλλαξης είναι ένας μικρός αριθμός p_m, ο οποίος λαμβάνει τιμές στο διάστημα $[0, ..., 0.2]$ με βήμα 0.005.

- Ελέγχεται αν ο τρέχων πληθυσμός ικανοποιεί ένα τουλάχιστον από τα κριτήρια τερματισμού, δηλαδή η σύγκλιση του αλγορίθμου και το συνολικό πλήθος δοκιμών. Αν κανένα από τα κριτήρια δεν ικανοποιείται, τότε ο αλγόριθμος επαναλαμβάνεται από το βήμα 4.

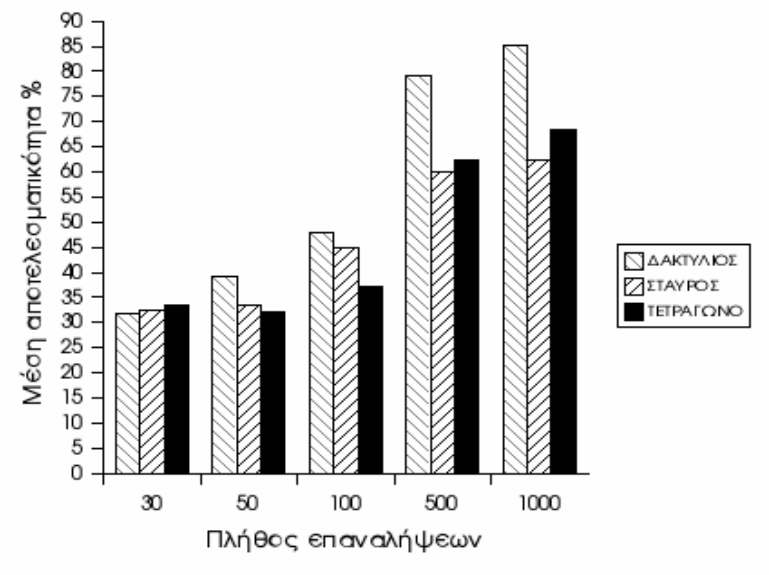
Τα αποτελέσματα των δοκιμών, για διάφορους συνδυασμούς παραμέτρων εισόδου, συνοψίζονται στον Πίνακα 4.12. Στον πίνακα 4.11 έχουμε την βασική διάταξη ΔΜΑ για κάθε μία από τις συναρτησεις μας. Στο σχήμα 4.16, δίνεται με την μορφή ραβδογράμματος η

($F1$)	**ΔMA**	30	250	50	0	2	0.6	0.75	1	1	2	50	0.4	1	1	1	3	2
($F2$)	**ΔMA**	30	250	50	0	2	0.6	0.75	1	1	4	50	0.4	1	1	1	3	2
($F3$)	**ΔMA**	30	250	50	0	2	0.6	0.75	1	1	6	50	0.4	1	1	1	3	2
($F4$)	**ΔMA**	30	250	50	0	2	0.6	0.75	1	2	6	50	0.4	1	1	1	3	2
($F5$)	**ΔMA**	30	250	50	0	2	0.6	0.75	1	1	6	50	0.4	1	1	1	3	2
($F6$)	**ΔMA**	30	250	50	1	3	0.6	0.75	1	1	6	100	0.4	1	2	1	3	2
($F7$)	**ΔMA**	30	250	50	1	3	0.6	0.75	1	1	6	100	0.4	1	2	1	3	2
($F8$)	**ΔMA**	30	250	50	1	3	0.6	0.75	1	1	6	100	0.4	1	2	1	3	2
($F9$)	**ΔMA**	30	250	50	1	3	0.6	0.75	1	1	6	100	0.4	1	2	1	3	2
($F10$)	**ΔMA**	30	250	50	1	3	0.6	0.75	1	1	6	100	0.4	1	2	1	3	2
($F11$)	**ΔMA**	30	250	50	1	3	0.6	0.75	1	1	2	100	0.4	1	2	1	3	2
($F12$)	**ΔMA**	30	250	50	1	3	0.6	0.75	1	2	4	100	0.4	1	2	1	3	2
($F13$)	**ΔMA**	30	250	50	1	3	0.6	0.75	1	1	2	100	0.4	1	2	1	3	2
($F14$)	**ΔMA**	30	250	50	1	3	0.6	0.75	1	1	2	100	0.4	1	2	1	3	2

Πίνακας 4.11: ΔMA με *GLS*



Σχήμα 4.15: Μέσοι Χρόνοι σε δευτερόλεπτα για MA GLS



Σχήμα 4.16: Μέση αποτελεσματικότητα για τρία διαφορετικά είδη γειτονιάς MA GLS

μέση αποτελεσματικότητα που βρέθηκε μέσα απο το σύνολο των πειραμάτων.

Από τα αποτελέσματα γίνεται φανερό ότι ο αλγόριθμος αυτός παρουσίασε πολύ καλή επίδοση, κυρίως ως προς την αποτελεσματικότητα αλλά και ως προς την αποδοτικότητα. Συγκεκριμένα, εντόπισε λύσεις της τάξης του 95% για 5 από τις συναρτήσεις ελέγχου, ενώ εμφάνισε πολύ υψηλή επίδοση στη Schwefel και στη Griewang, σχετικά καλή επίδοση στα προβλήματα Rastrigin, Watson, Penalty II, Powel badly scaled και πιο χαμηλή επίδοση παρουσίασε στο 10-διάστατο πρόβλημα Rozenbrock, Gulf research and development και Extenden powel singular. Και σε αυτήν την μέθοδο είναι προφανές ότι με χρήση μικρού μεγέθους πληθυσμού απαιτείται μικρότερος αριθμός δοκιμών για τον εντοπισμό της βέλτιστης λύσης, και συνεπώς βελτιώνεται σημαντικά η αποδοτικότητα του

	Ονομασία	30	50	100	500	1000
$F1$	Sphere model	47	58	62	80	95
$F2$	Rosenbrock	39	49	67	82	92
$F3$	Step	27	43	55	81	95
$F4$	Quatric	32	40	65	78	96
$F5$	Foxholes	26	36	61	70	97
$F6$	Rastrigin	32	37	43	65	79
$F7$	Schwefel	35	31	34	65	85
$F8$	Griewang	28	32	39	60	85
$F9$	Watson	26	34	56	71	80
$F10$	Extended Rosenbrock	27	33	30	71	81
$F11$	Penalty II	28	29	33	68	80
$F12$	Powel badly scaled	30	42	48	72	82
$F13$	Gulf research and development	34	40	45	70	82
$F14$	Extended Powel singular	34	44	33	78	81
	Μέση Αποτελεσματικότητα	31.79	39.14	47.93	79.10	85.20

Πίνακας 4.12: Ποσοστά επιτυχίας MA με GLS (Επαναλήψεις 30,50,100,500,1000)

αλγορίθμου.

Πολιτισμικός Αλγόριθμος

Στον Πολιτισμικό Αλγόριθμο τα βήματα της μεθόδου είναι τα παρακάτω:

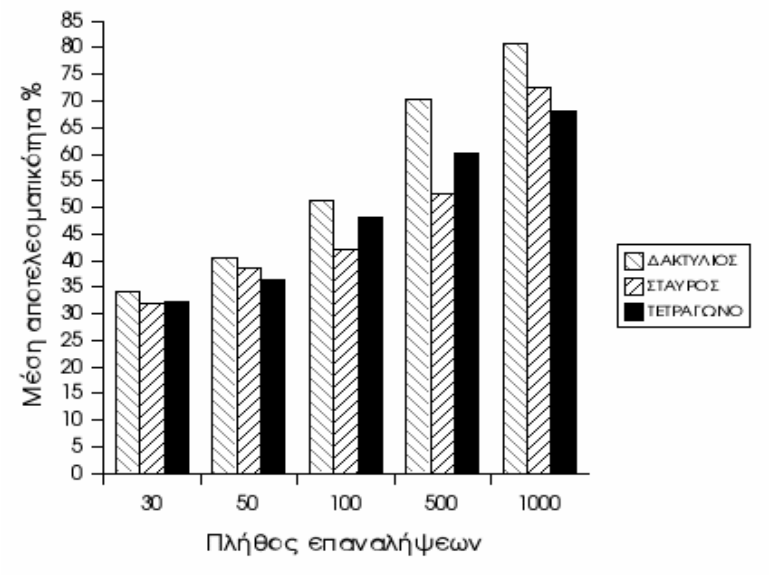
- Γεννάται ένας αρχικός πληθυσμό pop_0, αποτελούμενος από m σημεία ομοιόμορφα κατανεμημένα μέσα στον χώρο εφικτών λύσεων D.
- Υπολογίζεται η συνάρτηση καταλληλότητας $f(x)$ για κάθε άτομο στον πληθυσμό.
- Δημιουργείται ένα κενό αρχικό διάνυσμα B που αναπαριστά των χώρο πεποιθήσεων .

- Σε κάθε γενιά ο πληθυσμός μας εξελίσσεται βάση του σχήματος VIP[194].
- Στην φάση της ψηφοφορίας αξιολογούνται με βάση την τιμή της αντικειμενικής συνάρτησης. Τα σημεία ταξινομούνται κατά αύξουσα τιμή της αντικειμενικής συνάρτησης και αποθηκεύονται σε ένα διάνυσμα $B_{temp} = m_t, f_t$.
- Επιλέγονται με μια διαδικασία επιλογής κάποια στοιχεία απο το B_{temp} τα οποία και υπόκεινται στην διαδικασία της αναπαραγωγής.
- Εφαρμόζεται ένας τελεστής ανασυνδυασμού με συχνότητα $p_c = 0.6$
- Εφαρμόζεται ένας τελεστής μετάλλαξης με συχνότητα $p_m = (0, ..., 0.2)$ με βήμα 0.005.
- Στην συνέχεια τα στοιχεία του διανύσματος B_{temp} αντικαθιστούν τα στοιχεία του διανύσματος B. Ο ενημερωμένος πλέον χώρος πεποιθήσεων είναι πλέον έτοιμος για να επηρεάσει την εξέλιξη του πληθυσμού.
- Βάσει ενός τελεστή επιλογής, διαμορφώνεται η επόμενη γενιά $pop(t+1)$, η οποία περιέχει τα καλύτερα μέλη της προηγούμενης γενιάς. Ο τελεστής αυτός εφαρμόζεται ανάμεσα σε σημεία που επιλέγονται τόσο από το διαστημα πεποιθήσεων όσο και απο τον υπολοιπο πληθυσμό pop_t.
- Εφόσον δεν ικανοποιούνται τα κριτήρια τερματισμού του αλγορίθμου, όπως η σύγκλιση του αλγορίθμου, η οποία ελέγχεται μέσω της τυπικής απόκλισης του δείγματος τιμών της αντικειμενικής συνάρτησης, το συνολικό πλήθος δοκιμών ή αν προκύψει κάποια γνώστη στο χώρο πεποιθήσεων. Αν κανένα από τα κριτήρια δεν ικανοποιείται τότε επαναλαμβάνεται η διαδικασία εξέλιξης από το βήμα 3.

	Ονομασία	30	50	100	500	1000
$F1$	Sphere model	50	58	62	72	90
$F2$	Rosenbrock	40	49	67	70	95
$F3$	Step	30	43	55	68	92
$F4$	Quatric	37	40	65	78	90
$F5$	Foxholes	26	39	61	70	96
$F6$	Rastrigin	32	37	43	65	65
$F7$	Schwefel	40	40	55	65	65
$F8$	Griewang	30	32	39	65	70
$F9$	Watson	32	34	56	71	80
$F10$	Extended Rosenbrock	32	38	42	71	70
$F11$	Penalty II	30	29	45	68	78
$F12$	Powel badly scaled	30	42	48	72	80
$F13$	Gulf research and development	34	40	45	70	78
$F14$	Extended Powel singular	34	47	33	78	82
	Μέση Αποτελεσματικότητα	34.07	40.57	51.14	70.21	80.79

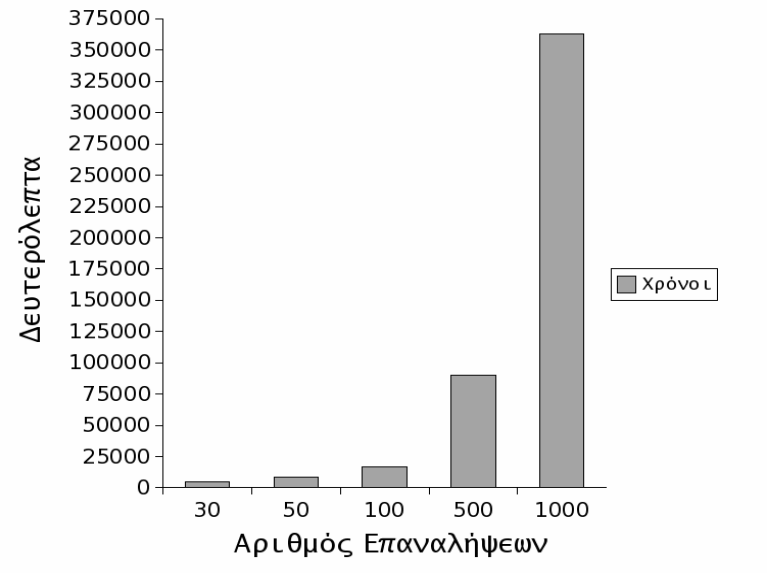
Πίνακας 4.13: Ποσοστά επιτυχίας CA (Επαναλήψεις 30,50,100,500,1000)

Τα αποτελέσματα των δοκιμών, για διάφορους συνδυασμούς παραμέτρων εισόδου, συνοψίζονται στον Πίνακα 4.13. Στο σχήμα 4.17, δίνεται με την μορφή ραβδογράμματος η μέση αποτελεσματικότητα που βρέθηκε μέσα απο το σύνολο των πειραμάτων. Στον οριζόντιο άξονα x εχουμε το πλήθος των επαναλήψεων, στον άξονα z έχουμε τη δομές γειτονιάς του πληθυσμού και στον άξονα των y έχουμε τη μέση αποτελεσματικότα σε ποσοστά %. Από τα αποτελέσματα φαινεται ότι η μεθοδος είναι πράγματι μια σχετικά καλυτερη μέθοδος από αυτή του απλού εξελικτικού αλγορίθμου, αφού αντιμετώπισε με καλύτερο ποσοστό σχεδόν τις περισσότερες από τις συναρτήσεις ελέγχου (με εξαίρεση την 10-διαστατη Extended Rozenbrock και την Powel badly scaled). Προφανώς,



Σχήμα 4.17: Μέση αποτελεσματικότητα για τρία διαφορετικά είδη γειτονιάς CA

απαιτήθηκε η θεώρηση μεγαλύτερου πληθυσμού όσο αυξανόταν ο βαθμός δυσκολίας του προβλήματος. Η αύξηση της τιμής της συχνότητας διασταύρωσης από 60% σε 100% επαιξε σημαντικό ρόλο στα αποτελέσματα, όπως και η χρήση της δομής δακτυλίου για τον πληθυσμό ενώ αντίθετα η αύξηση της συχνότητας μετάλλαξης από 1% σε 20% τα διαφοροποίησε όχι σημαντικά, πάντα όμως προς τη θετική κατεύθυνση. Η αύξηση τόσο της συχνότητας διασταύρωσης όσο και της συχνότητας μετάλλαξης είχε ως συνέπεια μεγαλύτερο υπολογιστικό φόρτο. Η χρήση της γνώσης που παρέχει ο χώρος πεποιθήσεων δίνει την δυνατότητα στον αλγόριθμο της αποφυγής εγκλωβισμού ευκολα σε τοπικά ακρότατα. Είναι προφανές ότι με χρήση μικρου μεγέθους πληθυσμού απαιτείται μικρότερος

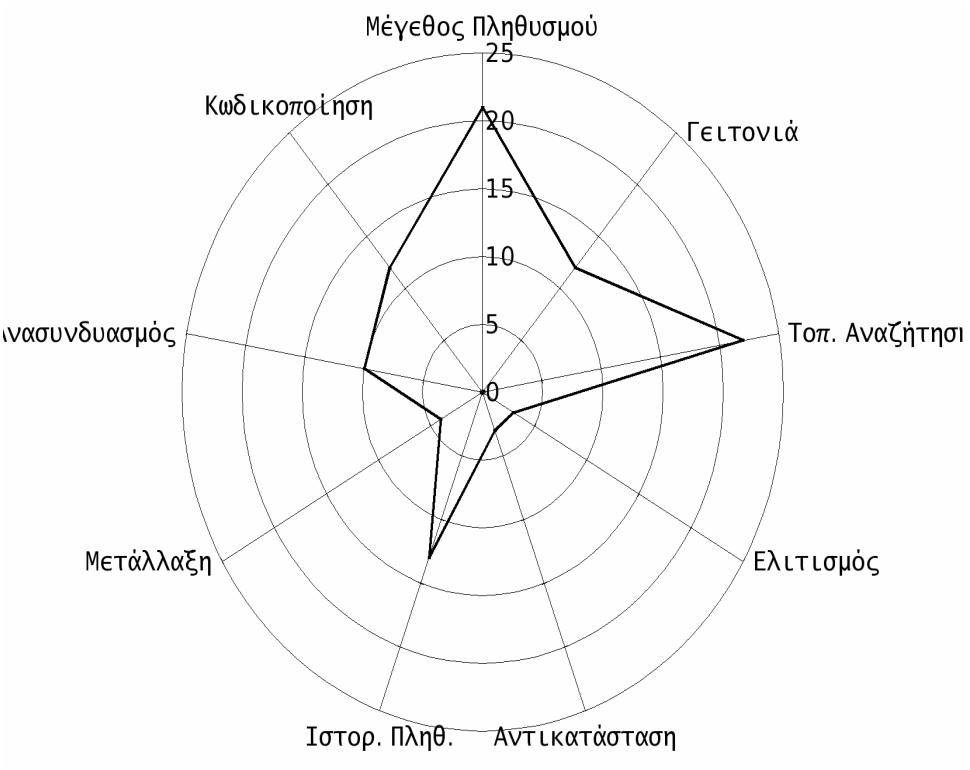


Σχήμα 4.18: Μέσοι Χρόνοι σε δευτερόλεπτα για CA

αριθμός δοκιμών για τον εντοπισμό της βέλτιστης λυσης, και συνεπώς βελτιώνεται σημαντικά η αποδοτικότητα του αλγορίθμου. Η μεθοδος αυτή βρήκε σχετικά καλές λύσεις στα προβλήματα μας.

4.3 Συμπεράσματα

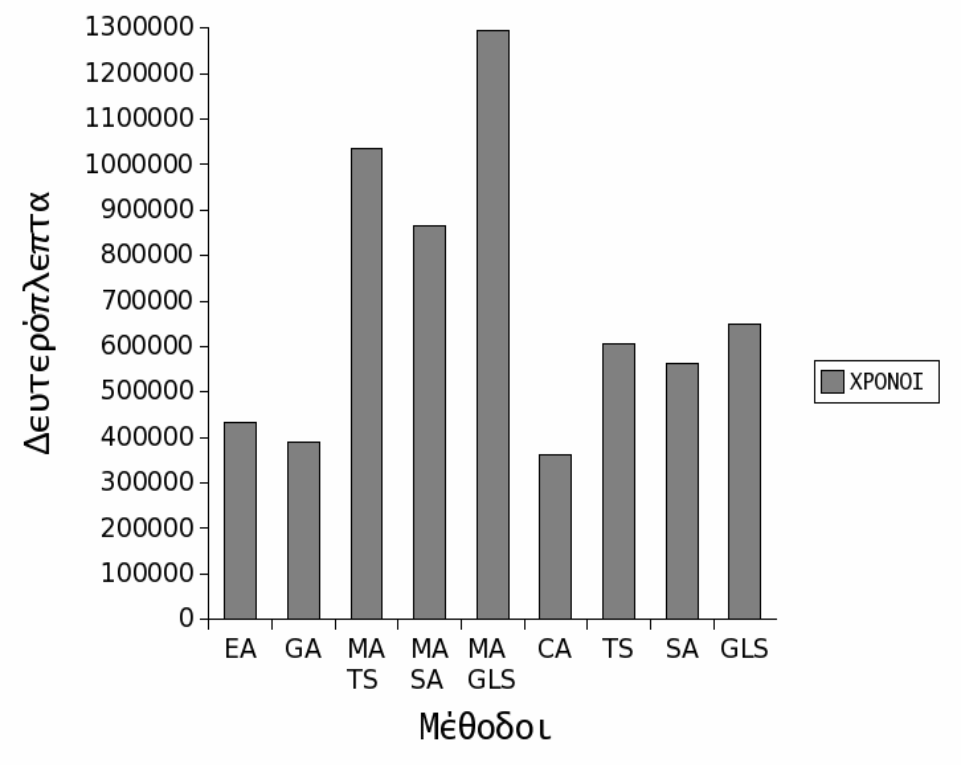
Τα συμπεράσματα που προκύπτουν μετά την ανάλυση που πραγματοποιήθηκε είναι: Ο μιμητικός αλγόριθμος με αποτρεπτική αναζήτηση αντιμετώπισε με σχετική επιτυχία ορισμένα από τα προβλήματα, ενώ απέτυχε εντελώς σε άλλα. Παρατηρήθηκε βέβαια αύξηση της αποτελεσματικότητας με αύξηση του πλήθους εκκινήσεων, ωστόσο τα περιθώρια βελτίωσης ήταν πεπερασμένα και εξαρτώμενα τόσο από τις ιδιαιτερότητες του εκάστοτε



Σχήμα 4.19: Ποσοστιαία επίδραση των κύριων χαρακτηριστικών της ΔΜΑ στην μεταβολή της μέσης αποτελεσματικότητας

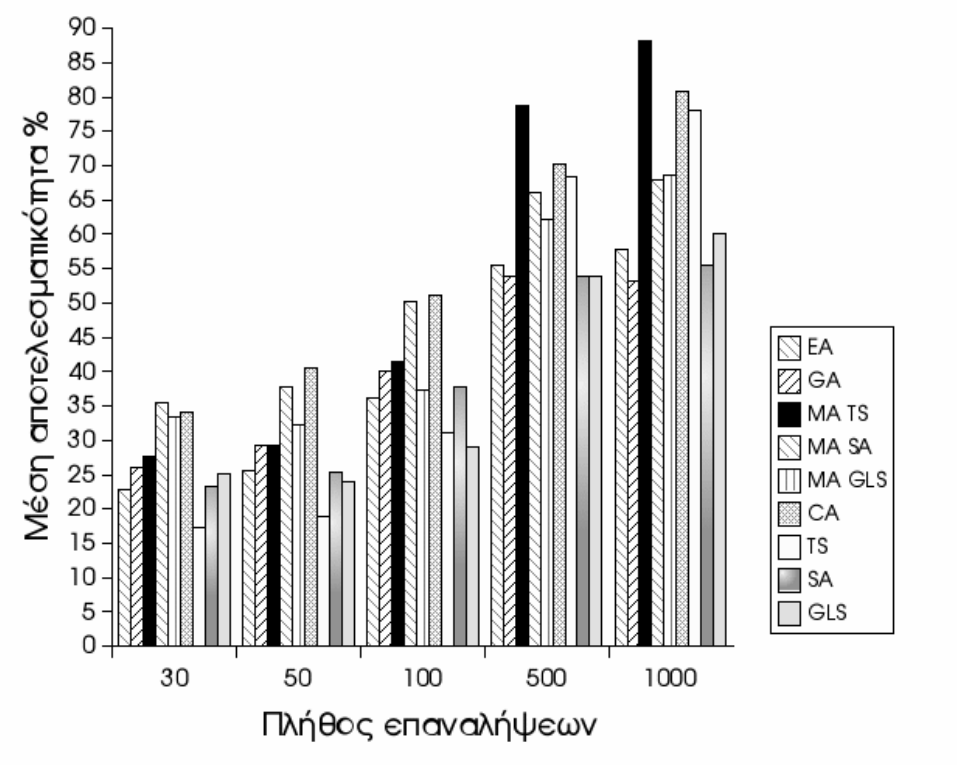
προβλήματος. Στο σχήμα 4.19 βλέπουμε την ποσοστιαία επίδραση στην μεταβολή της μέσης αποτελεσματικότητας των κύριων χαραχτηριστικών της ΔΜΑ και στο σχήμα 4.21 έχουμε μια σύγκριση της μέσης αποτελεσματικότητας οχτών μεθόδων για ένα σύνολο επαναλήψεων $30 \sim 1000$. Στις μεθοδους αυτές εχουμε συμπεριλάβει και τρείς ευρετικές μεθοδους την SA, TS, GLS.

Η επίδοση του απλού εξελικτικού αλγορίθμου, με χρήση δυαδικής κωδικοποίησης των μεταβλητών ελέγχου, δεν ήταν ικανοποιητική, όχι μόνο λόγω της σχετικά χαμηλής του αποτελεσματικότητας αλλά κυρίως εξαιτίας του υπερβολικά μεγάλου πλήθους δοκιμών



Σχήμα 4.20: Μέσοι χρόνοι σε δευτερόλεπτα για εννέα μεθόδους για 1000 επαναλήψεις

που απαιτείται για τη σύγκλιση στη βέλτιστη λύση. Η μέθοδος του μιμητικού αλγορίθμου με προσομοιούμενη ανόπτηση αποδείχθηκε ιδιαίτερα αποτελεσματική για τις περισσότερες συναρτήσεις ενώ ο μιμητικός αλγόριθμος με καθοδηγούμενη τοπική αναζήτηση αποδείχθηκε εξίσου αποτελεσματικός, μειονεκτώντας λίγο ως προς την ταχύτητα σύγκλισης. Κοινό χαρακτηριστικό όλων των μεθόδων ήταν η ευαισθησία τους ως προς ορισμένες αλγοριθμικές παραμέτρους εισόδου, όπως για παράδειγμα το μέγεθος του πληθυσμού. Κατά κανόνα, οι παράμετροι αυτές ορίζονται εμπειρικά, ωστόσο θα είχε ενδιαφέρον η ανάπτυξη τεχνικών αυτόματης ρύθμισης τους, αλλά το θέμα αυτό απαιτεί εκτενέστερη



Σχήμα 4.21: Σύγκριση της μέσης αποτελεσματικότητας εννέα μεθόδων

διερεύνηση.

Από τα παραπάνω γίνεται φανερή η υπεροχή των μιμητικών αλγορίθμων ενώ βλέποντας και τους πίνακες με τα αποτελέσματα των τεχνικών της προσομοιούμενης ανόπτησης, της τοπικής αναζήτησης και της καθοδηγούμενης τοπικής αναζήτησης για την επίλυση προβλημάτων ολικής βελτιστοποίησης συμπεραίνουμε ότι οι μεθόδοι αυτοί παρουσιάζουν μεγαλύτερη αποτελεσματικότητα όχι μόνο σε σχεση με αλλες εξελικτικές μεθόδους αλλα και σε σχεση με άλλες ευρετικές.

Αποδεικνύεται ότι κάτω από ορισμένες προϋποθέσεις, η ευρετική μέθοδος προσομοιωμένης ανόπτησης συγκλίνει πάντοτε στο ολικό ακρότατο όταν η θερμοκρασία τείνει στο μηδέν, σε προβλήματα συνδυαστικής βελτιστοποίησης [2]. Στην πράξη, ο ακριβής εντοπισμός του ολικού βέλτιστου απαιτεί υπερβολικά μεγάλο (θεωρητικά άπειρο) κύκλο επαναλήψεων, ωστόσο με κατάλληλη προσαρμογή του χρονοδιαγράμματος ανόπτησης μπορεί να επιτευχθεί ικανοποιητική προσέγγιση αυτού με λιγότερες δοκιμές. Για την ευρετική μέθοδο της αποτρεπτικής αναζήτησης δύο σοβαρά μειονεκτήματα της είναι ο φόρτος που απαιτείται σε προβλήματα πολλών μεταβλητών ελέγχου καθώς και το σχετικά μεγάλο πλήθος των αλγοριθμικών παραμέτρων εισόδου, ο ορισμός των οποίων γίνεται αυθαίρετα[201].

Τα αποτελέσματα των εκτελέσεων μας (σχήματα 4.8,4.12, 4.14, 4.16, 4.17) δίνουν μια σαφή ένδειξη σχετικά με το ποια δομή της γειτονιάς του πληθυσμού ενδείκνυται ως η καταλληλότερη για την πραγματοποίηση της επιλογής. Παρόλο όμως που η τοπική επιλογή σε μια γειτονιά με δομή δακτυλίου έδωσε καλύτερα αποτελέσματα σε σχέση με τις αλλες δύο περιπτώσεις δεν μπορούμε να συνάγουμε με ασφάλεια το συμπέρασμα ότι σε διαφορετικού είδους προβλήματα μια άλλη δομή δεν θα έδινε καλύτερα ή εφάμιλλα αποτελέσματα.

Στο σχήμα 4.20 βλέπουμε τους χρόνους σε δευτερόλεπτα που απαιτήθηκαν για την εκτέλεση 1000 επαναλήψεων που με βάση τα αποτελέσματα ήταν ικανοποιητικές για την επίτευξη ενός σχετικά καλού αποτελέσματος. Βλέπουμε ότι ιδίως στην περίπτωση των MA με τους οποίους πετύχαμε τα καλύτερα ποιοτικά αποτελέσματα οι χρόνοι είναι απαγορευτικά μεγάλοι. Τα μειονεκτήματα της προσομοιούμενης ανόπτησης μετά την υβριδοποίηση

με εξελικτικές τεχνικές ξεπερνιώνται σε μεγάλο βαθμό. Το ιδιο ισχύει και για τις αλλες δυο ευρετικές μεθόδους που δοκιμάσαμε. Θέμα σημαντικό που απασχόλησε και απασχολεί τους ερευνητές αποτελεί και η αύξηση κυρίως της αποδοτικότητας και η βελτίωση της αποτελεσματικότητας των ευρετικών και εξελεκτικών μεθόδων μέσω της παραλληλοποίησης, δηλαδή της αλγοριθμικής τεχνικής διάσπασης σύνθετων προβλημάτων σε επιμέρους υποπροβλήματα, τα οποία επιλύουν ηλεκτρονικοί υπολογιστές με πολλαπλούς επεξεργαστές. Σε αυτήν την κατεύθυνση βρίσκονταν και οι δημιουργοί των τριών μιμητικών αλγορίθμων που παρουσιάσαμε σε αυτό το κεφάλαιο. Και στις τρείς περιπτώσεις ο συνολικός πληθυσμός λύσεων επιμερίζεται σε ομάδες, σε κάθε μια από τις οποίες η εξέλιξη γίνεται ανεξάρτητα, ενώ κατά διαστήματα πραγματοποιείται ανταλλαγή πληροφοριών.

Κεφάλαιο 5

Παράλληλη Υλοποίηση Μιμητικών Αλγορίθμων

Σε αυτό το κεφάλαιο παρουσιάζουμε τέσσερις παράλληλες υλοποιήσεις μιμητικών αλγορίθμων με σκοπό να κερδίσουμε χρόνο σε χαμηλό κόστος. Οι τέσσερις προσεγγίσεις υλοποιούνται σε μια συστοιχία σταθμών εργασίας. Οι υλοποιήσεις συγκρίνονται με τις σειριακές εκτελέσεις και με παράλληλες υλοποιήσεις άλλων αντιπροσωπευτικών μεθοδολογιών.

Το κεφάλαιο είναι διαρθρωμένο ως εξής: στην ενότητα 5.1 κάνουμε μια σύντομη εισαγωγή στην περιοχή των παράλληλων υλοποιήσεων εξελικτικών και άλλων μεταευρετικών αλγορίθμων. Στην ενότητα 5.2 παρουσιάζονται τα επίπεδα παραλληλοποίησης και η επίδραση τους. Στην συνέχεια στην ενότητα 5.3 περιγράφεται η παράλληλη υλοποίηση των

μεθόδων του κεφαλαίου 4 στην πειραματική συστοιχία σταθμών εργασίας του πανεπιστημίου μας. Τέλος στην ενότητα 5.5 παρουσιάζονται και συγκρίνονται τα πειραματικά μας αποτελέσματα για τα 14 προβλήματα που περιγράφονται στο κεφάλαιο 4 καθώς και για το πρόβλημα του περιοδεύοντος πωλητή TSP.

Ένα μέρος των όσων παρουσιάζονται σε αυτό το κεφάλαιο έχουν δημοσιευτεί στο First Cracow Grid Workshop [69], στο 4th Metaheuristic International Conference [71], στο 3th meeting of the PAREO Euro working group on Parallel Processing in Operations Research [63], στο 4ο Πανελλήνιο Συνέδριο Υπολογιστικής Μηχανικής [77], στο 5ο συνέδριο Large Scale Computing καθώς και σε ένα άρθρο στο περιοδικό Journal of Applied Mathematics and Computation [78] και αναφέρεται στις παρακάτω εργασίες [162, 209, 115, 196] . Επιπλέον μέρος αυτού του κεφαλαίου έχει γίνει αποδεκτό για δημοσίευση στο περιοδικό Parallel Processing Letters.

5.1 Από ένα άτομο σε ένα πληθυσμό

Η ιδέα της ανάπτυξης εφαρμογών εξελικτικών αλγορίθμων σε συστήματα παράλληλης επεξεργασίας δεν είναι καινούργια. Η διερεύνηση μεγάλων χώρων αναζήτησης με πολύπλοκες συναρτήσεις ποιότητας και η χρήση πολυάριθμων πληθυσμών επιβάλλουν την ανάγκη γρήγορων, υπολογιστικά ισχυρών υλοποιήσεων. Η διαθεσιμότητα γρήγορων και φθηνών παράλληλων υπολογιστών, κάνει δυνατή την εφαρμογή των εξελικτικών και άλλων μεταευρετικών μεθόδων σε δύσκολα και σύνθετα προβλήματα στα οποία γενικώς απαιτείται η χρήση μεγάλων πληθυσμών. Επίσης μερικές δυσκολίες που αντιμετωπίζουν

αρκετοί από τους γνωστούς εξελικτικούς αλγορίθμους μπορούν να αντιμετωπιστούν πιο αποτελεσματικά με την χρήση των παράλληλων εξελικτικών αλγορίθμων [42].

Μία μέθοδος για να παραλληλίσουμε έναν ΕΑ είναι να εκτελέσουμε τον ίδιο αλγόριθμο σε πολλούς επεξεργαστές σαν ανεξάρτητες διεργασίες [42]. Η μόνη επικοινωνία ανάμεσα στις διεργασίες είναι ότι μία από αυτές συγκεντρώνει το τελικό αποτέλεσμα από όλες τις υπόλοιπες και το αναφέρει στο χρήστη [29].

Ένας δεύτερος τρόπος παραλληλοποίησης ενός ΕΑ είναι με την χρήση του μοντέλου συντονιστή εργαζομένου (master-worker), που παρουσιάζεται στις εργασίες [82, 42]. Ο συντονιστής είναι υπεύθυνος για την εγκατάσταση των εργαζομένων, την ταξινόμηση της ποιότητας των ατόμων και την επιλογή των γονέων, την επικοινωνία με τον χρήστη και την επίβλεψη του συνολικού πληθυσμού. Συνήθως ο συντονιστής δεσμεύει έναν επεξεργαστή και δημιουργεί τις διεργασίες που αντιστοιχούν στους εργαζομένους. Οι γονείς που επιλέγονται στέλνονται στους επεξεργαστές εργασίας (εργαζόμενους), όπου ανασυνδυάζονται ή και μεταλλάσσονται για να δημιουργήσουν τους απογόνους. Οι απόγονοι αξιολογούνται και επιστρέφονται στο κύριο επεξεργαστή. Σημαντικές προσπάθειες ταξινόμησης των παράλληλων εξελικτικών αλγορίθμων έγινε στο παρελθόν από τους T. Baeck και F. Hoffmeister [17] ενώ μια περιεκτική επισκόπηση των εφαρμογών των παράλληλων εξελικτικών αλγορίθμων μπορεί ο αναγνώστης να αναζητήσει στην εργασία των D. Duvivier et al. [14, 29] .

Τα υπάρχοντα μοντέλα μπορούν να ενταχθούν σε δύο γενικότερες κατηγορίες παραλληλισμού, οι οποίες αποκαλούνται χαμηλής ανάλυσης (coarse grained) [25, 55] και υψηλής ανάλυσης (fine grained) [55]. Η εφαρμογή διαφόρων μοντέλων εξέλιξης και στις δύο

κατηγορίες παραλληλισμού δημιουργεί πολυάριθμες παραλλαγές, που σε πολλές περιπτώσεις δύσκολα ταξινομούνται[42, 175]. Οι πιο γνωστοί ΕΑ χαμηλής ανάλυσης είναι οι κατανεμημένοι ΕΑ ή ΕΑ με νησίδες και οι πιο γνωστοί ΕΑ υψηλής ανάλυσης είναι οι κυταρρικοί ΕΑ και οι μαζικά παράλληλοι ΕΑ. Οι χαμηλής ανάλυσης ΕΑ εφαρμόστηκαν κατά κύριο λόγο σε MIMD πλατφόρμες ενώ οι υψηλής ανάλυσης σε SIMD. Γενικά όλοι οι παράλληλοι ΕΑ χαρακτηρίζονται όπως αναφέρεται στην εργασία των P. Calegari, G. Coray, A. Hertz, D. Kobler και P. Kuonen [41] από ένα είδος συμβιβασμού ανάμεσα στην ποιότητα λύσης και στην επιτάχυνση.

Στην ταξινόμηση του Alba και των συνεργατών του [29] συγχέονται οι παράλληλοι εξελικτικοί αλγόριθμοι που βασίζονται σε κατανεμημένους πληθυσμούς με τους παράλληλους υβριδικούς εξελικτικούς αλγορίθμους σε νησίδες. Ο πληθυσμός ενός παράλληλου ΕΑ μπορεί να διανεμηθεί σε μια απομακρυσμένη μονάδα επεξεργασίας και κατά συνέπεια μπορεί να θεωρηθεί ένα σύνολο διανεμημένων υποσυνόλων πληθυσμού που θα μπορούσαν να θεωρηθούν νησίδες. Όμως αυτό το σύνολο υποσυνόλων πληθυσμών διαμορφώνει έναν ενιαίο πληθυσμό υπό την προυπόθεση οτι οι νησίδες εξελίσσονται ανεξάρτητα και συνεργάζονται. Η επιλογή της χρήσης ενός πληθυσμού ή νησίδων και επιλογή της διανομής του πληθυσμού πρέπει να διαχωριστεί [29, 136].

Πολλές φορές στη βιβλιογραφία γίνεται κακή χρήση του όρου *Παράλληλος Εξελικτικός Αλγόριθμος* και χρησιμοποιείται μόνο για την περιγραφή ενός εξελικτικού αλγορίθμου βασισμένου στο μοντέλο νησίδων [180, 117].

Μια άλλη πολύ καλή πρόσφατη ταξινόμηση των παράλληλων μεταευρετικών αλγορίθμων έγινε από τον Talbi [202]. Ο όρος μεταευρετικός αναφέρεται είτε σε ΕΑς είτε σε

παραδοσιακούς ευρετικούς αλγορίθμους που μπορούν να εφαρμοστούν σε διαφορετικά προβλήματα σε αντίθεση με τους ευρετικούς αλγορίθμους που σχεδιάστηκαν για την επίλυση συγκεκριμένων προβλημάτων. Στην ανάλυση αυτή γίνεται μια πολύ καλή αναφορά σε θέματα σχεδιασμού και εφαρμογής των μεταευρετικών αλγορίθμων. Στα θέματα σχεδιασμού έχουμε μια ιεράρχηση-κατηγοριοποίηση του τρόπου με τον οποίο γίνεται η υβριδοποιηση των αλγορίθμων.

Η ταξινόμηση του Talbi παρουσιάζεται στο Κεφάλαιο 1 όπως εφαρμόζεται σε ακολουθιακούς αλγορίθμους. Εδώ η ταξινόμηση Talbi (βλ. 5.1) επεκτείνεται ώστε να περιλαμβάνει και τα θέματα παράλληλης υλοποίησης. Με τον όρο χαμηλού επιπέδου L αναφερόμαστε στις περιπτώσεις που ο πληθυσμός διαιρείται σε μικρότερους υποπληθυσμούς ενώ με τον όρο υψηλού επιπέδου αναφερόμαστε στις περιπτώσεις που ο πληθυσμός είναι χωρισμένος σε πολλές μικρές περιορισμένες γειτονικές περιοχές. Στην πρώτη περίπτωση οι υποπληθυσμοί διανέμονται σε διάφορους επεξεργαστές όπου και εξελισσονται. Στην δευτερη περίπτωση οι υποπληθυσμοί στέλνονται όλοι σε γειτονικούς επεξεργαστές. Οι όροι ανεξάρτητη εξέλιξη και συνεξέλιξη R και C αναφέρονται αντίστοιχα στην περιπτώση που η εξέλιξη γίνεται ανεξάρτητα σε κάθε επεξεργαστή και στην περίπτωση όπου οι πληθυσμοί εξελίσσονται σε νησίδες που έχουν την δυνατότητα να επικοινωνούν απευθείας η μια με την άλλη. Αν ο MA είναι ετερογενής (het), μπορεί να απαιτούνται διάφορες πιθανώς διαφορετικές παραλληλοποιήσεις. Αν κάθε μέθοδος χειρίζεται ένα διαφορετικό πρόβλημα (spe) ή ένα επιμέρους πρόβλημα (par), τότε η κατανομή των MA σε διαφορετικές μονάδες επεξεργασίας επιτρέπει την κατανομή των δεδομένων αναλόγως, και ως εκ τούτου υπάρχει ένα κέρδος στη χρήση της μνήμης. Στην αντίθετη περίπτωση, κατά την οποία

Συμβολισμός	Περιγραφή - Χαρακτηρισμός
L	Χαμηλού επιπέδου
H	Υψηλού επιπέδου
R	Ανεξάρτητη Εξέλιξη
C	Συνεξέλιξη
hom	Ομογενείς
het	Ετερογενείς
par	Διατμηση Χώρου Αναζήτησης
glo	Όλος ο χώρος αναζήτησης
spe	Ειδικού σκοπού προβλήματα
gen	Γενικού σκοπού προβλήματα

Πίνακας 5.1: Ταξινόμηση του Talbi

όλες οι μέθοδοι αναζητούν στον ίδιο χώρο αναζήτησης (glo, gen) δεν μπορεί να υπάρχει κέρδος στη χρήση της μνήμης. Στη διατριβή μας θα ασχοληθούμε με την παράλληλη υλοποίηση των ΜΑ. Ο χρόνος εκτέλεσης των ακολουθιακών ΜΑ είναι πολύ μεγάλος και σε αρκετές περιπτώσεις απαγορευτικός σε ένα συμβατικό υπολογιστή. Στα πλαίσια της προσπάθειας οι αλγόριθμοι αυτοί να γίνουν πιο γρήγοροι, αποτελεσματικοί και ευέλικτοι εντάσσεται και η υλοποίηση αυτών σε διάφορες νησίδες επεξεργαστών της πειραματικής συστοιχίας του Πανεπιστημίου Μακεδονίας.

5.1.1 Παράλληλοι Εξελικτικοί Αλγόριθμοι Χαμηλής Ανάλυσης (Μοντέλο Μετανάστευσης)

Πολλοί εξελικτικοί αλγόριθμοι και άλλοι υβριδικοί μεταευρετικοί αλγόριθμοι με κατανεμημένες πληθυσμιακές δομές έχουν περιγραφεί στην διεθνή βιβλιογραφία [60]. Στα

μοντέλα χαμηλής ανάλυσης ο πληθυσμός διαιρείται σε υποπληθυσμούς. Κάθε υποπληθυσμός εξελίσσεται ανεξάρτητα από τους υπόλοιπους για κάποιο αριθμό γενιών. Μετά την περίοδο απομόνωσης ένας αριθμός ατόμων ανταλλάσσεται μεταξύ των υποπληθυσμών [93]. Ο αριθμός των ατόμων που ανταλλάσσονται, η μέθοδος επιλογής των ατόμων προς μετανάστευση και η μέθοδος μετανάστασταστευσης καθορίζει την γενετική απόκλιση μεταξύ των υποπληθυσμών και την ανταλλαγή πληροφορίας μεταξύ των υποπληθυσμών.

Οι απομονωμένοι υποπληθυσμοί βοηθούν στην διατήρηση της γενετικής απόκλισης. Οι γόνοι ενός υποπληθυσμού είναι σχετικά απομονωμένοι από τους γόνους των υπολοίπων. Έτσι κάθε υποπληθυσμός διερευνά ένα διαφορετικό μέρος του χώρου λύσεων.

Αρκετοί ερευνητές χρησιμοποιούν μια τέτοια τεχνική για να επιτευχθεί μεγαλύτερη ταχύτητα και για να επιτραπεί σε πολύ μεγαλύτερα πληθυσμιακά μεγέθη να αναμειχθούν σε λογικούς χρονους [193] ενώ άλλοι χρησιμοποιούν πολυάριθμους μικρούς, συνεχώς αλληλοεπηρεαζόμενους υποπληθυσμούς οι οποίοι εξελίσσονται παράλληλα [50].

Στα μοντέλα μετανάστευσης ο πληθυσμός είναι χωρισμένος σε λίγους, σχετικά απομονωμένους υποπληθυσμούς [40]. Κάθε υποπληθυσμός χρησιμοποιεί έναν ανεξάρτητο ΕΑ παράλληλα με τους άλλους, για την βελτιστοποίηση των γόνων του. Κατά διαστήματα, κάποιοι από τους καλύτερους γόνους μεταναστεύουν τυχαία μεταξύ οποιονδήποτε υποπληθυσμών. Σε μερικές υλοποιήσεις η μετανάστευση των καλυτέρων γόνων γίνεται μεταξύ γειτονικών πληθυσμών (Βηματικό μοντέλο). Συνηθως στους μεταευρετικούς δεν γίνεται διάκριση μεταξύ του μοντέλου νησίδων και του βηματικού μοντέλου και αντιμετωπίζονται μέσα στα πλαίσια του μοντέλου μετανάστευσης [202, 152].

Πολλές φορές χρησιμοποιηθηκε το μοντέλο της μετανάστευσης για γενετικούς αλγορίθμους GA ή εξελικτικούς αλγορίθμους EA με τυχαία μετανάστευση γόνων[21, 3]. Το ολικό βέλτιστο βρέθηκε με μεγαλύτερη επιτυχία κάνοντας χρήση του μοντέλου μετανάστευσης, παρά με την χρήση απομονωμένων πληθυσμών. Η σύγκλιση ήταν ταχύτερη με την χρήση μεγάλου ποσοστού μετανάστευσης [4].

Πολλά στοιχεία για την αποτελεσματικότητα και την αποδοτικότητα των παράλληλων EA σε σχέση με άλλες εξελικτικές τεχνικές μπορούμε να δουμε στις εργασίες των P. Surry και N. J. Radcliffe [177], των C.B. Pettey, M. R. Leuze και J. J. Grefenstette [170] και των E. Alba και J. M. Troya [10].

Οι μιμητικοί αλγόριθμοι χαρακτηρίζονται ως καλές μέθοδοι για την διερεύνηση αρκετών προβλήματων [154, 159, 167, 153] με μεγάλους χώρους αναζήτησης [146, 177, 47, 91, 156]. Αυτό οφείλεται στο οτι τα άτομα που απαρτίζουν τον πληθυσμό εξελίσσονται ανεξάρτητα προς την επίτευξης μιας καλής υποψήφιας λύσης [160, 156, 72]. Το μέγεθος των περιοχών που εξερευνούνται ταυτόχρονα από ένα MA εξαρτάται από τον αριθμό των ατόμων του πληθυσμού όπως και στους υπόλοιπους εξελικτικούς αλγορίθμους[18]. Ο αριθμός των ατόμων που απαιτείται για να έχουμε μια καλή διερεύνηση του χώρου αναζήτησης ειναι συνήθως μεγάλος [155, 138, 65]. Ένας ακόμα λόγος για τον οποίο θεωρούνται καλές μεθοδοι είναι ότι χρησιμοποιούν μεθόδους τοπικής αναζήτησης πριν την εξέλιξη του πληθυσμού [156]. Το γεγονός ότι η διαχείριση πολλών ατόμων είναι ιδιαίτερα χρόνοβόρα [10, 210, 211] και ότι το ποσό ανεξάρτητης επεξεργασίας που απαιτείται για την εξέλιξη των ατόμων ειναι σχετικά μεγάλο μας οδηγησε στον παραλληλισμό των εν λόγω μεθόδων.

5.2 Επίπεδα Παραλληλοποίησης

Ένα επίπεδο παραλληλοποίησης οποιουδήποτε Παράλληλου Εξελικτικού ή Μεταευρετικού Αλγορίθμου πιο γενικά, περιγράφει τον τεμαχισμό (granularity) αυτού σε μικρότερες μονάδες (οντότητες). Το επίπεδο καθορίζεται από το μικρότερο μέγεθος x των οντοτήτων που διανέμονται σε κάθε μια από τις μονάδες επεξεργασίας. Στον πίνακα 5.2 βλέπουμε τα τέσσερα πρώτα επίπεδα παραλληλοποίησης για ένα εξελικτικό αλγόριθμο.

Το επίπεδο παραλληλοποίησης L_{-1} υποδηλώνει μια προσέγγιση υψηλής ανάλυσης παραλληλοποίησης. Σε αυτό το επίπεδο κάθε μονάδα επεξεργασίας χειρίζεται ένα μόνο τμήμα ενός ατόμου και ο αριθμός των απαραίτητων μονάδων επεξεργασίας είναι μεγάλος. Επίσης, το αναμενόμενο φορτίο επικοινωνίας είναι πολύ μεγάλο. Αν οι αλφαριθμοσειρές (bit) των ατόμων έχουν κωδικοποιηθεί δυαδικά, τότε μια μονάδα επεξεργασίας ευθύνεται για ορισμένα στοιχεία ενός ατόμου (π.χ. η μονάδα επεξεργασίας 5 ευθύνεται για το 5ο στοιχείο κάθε αλφαριθμοσειράς δυαδικών στοιχείων bit). Ο αριθμός των απαιτούμενων μονάδων επεξεργασίας και του φορτίου επικοινωνίας είναι υψηλός.

Το επίπεδο παραλληλοποίησης L_0 είναι το μέσο επίπεδο καθώς αντιστοιχεί στην κατάτμηση ενός πληθυσμού σε υποπληθυσμούς. Σύμφωνα με την κατάτμηση αυτή οι πληροφορίες σχετικά με ολόκληρο τον πληθυσμό πρέπει να διατηρούνται συνεπείς, και συνεπώς το φορτίο επικοινωνίας θα είναι πιθανότατα υψηλό. Η παραλληλοποίηση αυτή ορίζεται ως ολική παραλληλοποίηση (global parallelization) στο [42]. Το επίπεδο παραλληλοποίησης L_1 είναι το πιο δημοφιλές επίπεδο στη βιβλιογραφία καθώς είναι το πιο εύκολο να εφαρμοστεί σε ΕΑ [42, 3]. Η βασική μας ιδέα είναι η εφαρμογή ενός αρχικού ΜΑ σε

Επίπεδο	Οντότητα ανα μονάδα επεξεργασίας	Μέγεθος x για κάθε οντότητα
$L_{-1}(x)$ Αποκωδ. στοιχείου	1 αποκωδικ. τμήμα ατόμου	$x \in [1, \|ind\| - 1]$
$L_0(x)$ Ατόμου	1 υποπληθυσμός	$x \in [1, \|pop\| - 1]$
$L_1(x)$ Πληθυσμού	1 υποσυνολο αρχιπελάγους	$x \in [1, \|arch\| - 1]$
$L_2(x)$ Αρχιπέλαγους	1 σύνολο πληθυσμών	.....

Πίνακας 5.2: Διάφορα επίπεδα Παραλληλοποίησης Μιμητικών Αλγορίθμων

ανεξάρτητους πληθυσμούς (ή νησίδες). Η απλή εφαρμογή ενός παράλληλου ΜΑ επιπέδου L_1 δεν είναι πολύπλοκη: κάθε μονάδα επεξεργασίας χειρίζεται μια απομακρυσμένη νησίδα και εφαρμόζει το σειριακό ΜΑ τοπικά (υπό τον όρο ότι ο αριθμός των ατόμων είναι τουλάχιστον διπλάσιος από τον αριθμό των διαθέσιμων μονάδων επεξεργασίας. Στην αντίθετη περίπτωση, οι νησίδες με ένα άτομο θεωρούνται άτομα και συνεπώς επιπέδου L_0). Το L_1 περιλαμβάνει, επίσης, πιο πολύπλοκα σχέδια που επιτρέπουν την ύπαρξη περισσότερων νησίδων από ό,τι μονάδες επεξεργασίας. Οι υποπληθυσμοί που αποτελούν τμήμα ενός μόνου πληθυσμού δεν πρέπει να αναμειγνύονται με ανεξάρτητες νησίδες καθώς δεν εξελίσσονται σύμφωνα με τον ίδιο αλγόριθμο. Πολλές ανεξάρτητες νησίδες απαρτίζουν ένα αρχιπέλαγος . Τα μειονεκτήματα της υψηλής ανισορροπίας φορτίου απο μια άνιση διανομή των νησίδων [28] και το μειονέκτημα υψηλής επιβάρυνσης επικοινωνίας [29] μπορούν να εξαλειφθούν από μια μέθοδο προεπεξεργασίας διανομής των νησίδων, όπου οι υποπληθυσμοί τοποθετούνται σε ένα από τους p σταθμούς εργασίας έτσι ώστε η διαφορά ανάμεσα στο μέγεθος του υποπληθυσμού στο μικρότερο και μεγαλύτερο σταθμό εργασίας να ελαχιστοποιείται.

Η ΔΜΑ που παρουσιάστηκε στο κεφάλαιο 3 μας δίνει την δυνατότητα να δώσουμε όλες τις απαραίτητες πληροφορίες για τους μιμητικούς και γενικότερα για τους υβριδικούς μεταευρετικούς αλγορίθμους. Με την ταξινόμηση όμως του Talbi [202] που αναφέραμε και στο κεφάλαιο 1 διακρίνουμε τις παρακάτω περιπτώσεις παράλληλων μεταευρετικών αλγορίθμων τις οποίες και χρησιμοποιήσαμε για τον περιγραφή και των δικών μας υλοποιήσεων.

Υβριδική συνεξέλιξη υψηλού επιπέδου-(HCH)

Η υβριδική συνεξέλιξη υψηλού επιπέδου HCH αντιστοιχεί στο μοντέλο νησίδας επιπέδου L_1. Η επιλογή του καλύτερου επιπέδου L_l ($L_l \in [L_{-1}, L_0]$) βασίζεται στην εφαρμογή των αλγοριθμικών βημάτων σε κάθε νησίδα. Για παραδειγμα αναφέρουμε την περίπτωση ενος ΕΑ στον οποίο περιοδικά οι απόγονοι με καλή ποιότητα, τυχαία διαλεγμένοι μεταναστεύουν μεταξύ των γειτονικών επεξεργαστών και αντικαθιστούν τους χειρότερους γόνους των ληπτών. Η ανταλλαγή στο παράδειγμα αυτό δεν γίνεται πάντα με τους ιδιους επεξεργαστές. Οι αλγόριθμοι που χρησιμοποιουνται είναι ομογενείς και κάθενας από αυτούς λύνει το ίδιο πρόβλημα στο ίδιο διάστημα αναζήτησης.

Υβριδική ανεξαρτησία υψηλού επιπέδου-(HRH)

Η τάξη υβριδική ανεξαρτησία υψηλού επιπέδου HRH περιγράφει ανεξάρτητους μεταευρετικούς που εκτελούνται διαδοχικά. Για παράδειγμα θα αναφέραμε ένα ΕΑ ο οποίος μπορεί να χρησιμοποιηθεί για την εξεύρεση μιας λύσης, η οποία θα βελτιωθεί από έναν αλγόριθμο τοπικής αναζήτησης, ή ένα ΜΑ ο οποίος μπορεί να λάβει ως δεδομένο τα αποτελέσματα

ενός αλγορίθμου τοπικής αναζήτησης . Και οι δύο λύσεις μπορούν να εφαρμοστούν ακόμα και η μία μετά την άλλη. Η παραλληλοποίηση ενός τέτοιου υβριδικού αλγόριθμου μπορεί να πραγματοποιηθεί παραλληλοποιώντας ανεξάρτητα κάθε μεταευρετική μέθοδο. Η εκτέλεση των μεταευρετικών μπορεί να παραγματοποιηθεί στη συνέχεια στις ίδιες μονάδες επεξεργασίας, σειριακά, καθώς κάθε μια απαιτεί το τελικό αποτέλεσμα της προηγούμενης. Επίσης, θα ήταν δυνατή μια παράλληλη εφαρμογή με προσέγγιση διασωλήνωσης, αν προγραμματίζονται πολλές διαδοχικές εφαρμογές και αν οι διαφορετικές φάσεις είχαν περίπου τον ίδιο χρόνο εκτέλεσης. Παρόλα αυτά, η απόδοση μιας μόνο εκτέλεσης του αλγόριθμου δεν θα ήταν καθόλου καλή καθώς θα χρησιμοποιούνται άμεσα μόνο οι μονάδες επεξεργασίας που είναι υπεύθυνες για ένα μεταευρετικό βήμα.

Υβριδική συνεξέλιξη χαμηλού επιπέδου-(LCH)

Η τάξη υβριδική συνεξέλιξη χαμηλού επιπέδου LCH χαρακτηρίζει τους αλγόριθμους στους οποίους ένα μεταευρετικό βήμα αντικαθίσταται από μια άλλη μεταευρετική λειτουργία. Ένα παράδειγμα για αυτή την περίπτωση αποτελεί και η περίπτωση στην οποία ο πληθυσμός ενός MA βελτιώνεται από ένα αλγόριθμο τοπικής αναζήτησης LS σε κάθε γενεά. Ο αλγόριθμος (LS) πραγματοποιεί ανεξάρτητες αναζητησεις για να βελτιώσει τα άτομα ενός πληθυσμού.

Υβριδική ανεξαρτησία χαμηλού επιπέδου-(LRH)

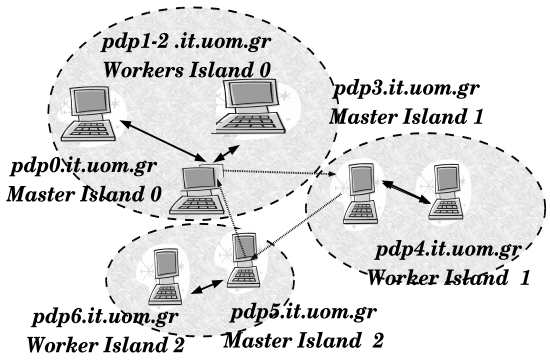
Η τάξη υβριδική ανεξαρτησία χαμηλού επιπέδου LRH αναπαριστά τους αλγόριθμους

στους οποίους μια μεταευρετική μέθοδος ενσωματώνεται σε μια ευρετική μιας και μόνο λύσης. Στην περίπτωση αυτή ο ΜΑ μπορεί να παραλληλιστεί σαν να ήταν μόνος και ανεξάρτητος. Καθώς ο αλγόριθμος θα πρέπει να εφαρμοστεί πολλές φορές (για να παρέχει το τελικό του αποτέλεσμα στον τελικό υβριδικό αλγόριθμο όπου ενσωματωνεται), ένα σύστημα απομνημόνευσης των συνολικών δεδομένων του προβλήματος μεταξύ δύο εκτελέσεων οδηγεί στην άσκοπη δαπάνη χρόνου καθώς θα πρέπει να γίνει εκ νέου ανάγνωση αυτού.

5.3 Παράλληλη Υλοποίηση Μιμητικών Αλγορίθμων

Για τις παράλληλες υλοποίησεις μας χρησιμοποιησαμε τη πειραματική συστοιχία του Πανεπιστημίου Μακεδονίας. Στο σχήμα 5.1 έχουμε ένα παράδειγμα κατάτμησης 3 νησίδων 7 επεξεργαστών της πειραματικής συστοιχίας του Πανεπιστημίου Μακεδονίας. Κάθε νησίδα απαρτίζεται από ένα σταθμό εργασίας συντονιστή και μια συλλογή από σταθμούς εργασίας εργαζομένους. Ο συντονιστής καθε νησίδας στη γενική περίπτωση χρησιμοποιείται για να διαμερίσει μια ομάδα πληθυσμών σε ένα σύνολο από διάφορους μικρότερους υποπληθυσμούς.

Κάθε νησίδα επεξεργαστών εκτελεί το ίδιο αλγοριθμικό μοντέλο αν πρόκειται για τις μεθοδους PMA1-PMA3 ή ένα διαφορετικό από τις υπόλοιπες αν πρόκειται για την PMA4.



Σχήμα 5.1: Κατανομή 3 νησίδων σε 7 επεξεργαστικές μοναδες της πειραματικής συστοιχιας του ΠΑΜΑΚ

Στην πρώτη περίπτωση έχουμε ομοιογένεια ως προς τα είδη των αλγορίθμων.

Στην αρχή της εκτέλεσης ενός ΜΑ τα δεδομένα πρέπει συνήθως να κατανέμονται σε κάθε μονάδα επεξεργασίας. Η κατανομή μπορεί σε γενικές γραμμές να είναι χρονοβόρα ανάλογα με τα χαρακτηριστικά του δικτυου που χρησιμοποιείται ή του παραλληλου συστήματος (κυρίως της αρχιτεκτονικής του). Είναι σημαντικό το κατά πόσο κάθε μονάδα επεξεργασίας μπορεί να έχει πρόσβαση στο ίδιο σύστημα αρχείων για να πάρει τις πληροφορίες της ή κατά πόσο μια μόνο μονάδα επεξεργασίας μπορεί να φορτώσει τις πληροφορίες και να τις μεταδώσει σε άλλες μονάδα επεξεργασίας $p-1$. Τα προβλήματα ανταγωνισμού πρέπει να αποφεύγονται όσο το δυνατόν περισσότερο. Το πρόβλημα αυτό δεν αφορά συγκεκριμένα έναν παράλληλο ΜΑ: είναι κοινό σε όλους του παράλληλους αλγόριθμους. Η μελέτη του, επομένως, είναι πολύ γενική για να συζητηθεί στο παρόν κεφάλαιο. Έτσι, θεωρούμε ότι στο υπόλοιπο κεφάλαιο οι πληροφορίες διατίθενται τοπικά σε κάθε μονάδα επεξεργασίας στην αρχή της εκτέλεσης του προγράμματος.

Η επιλογή της καλύτερης τεχνικής παραλληλοποίησης εξαρτάται από τον ίδιο τον αλγόριθμο, αλλά και από κάποιες ιδιότητες του προβλήματος, όταν είναι γνωστές εκ των προτέρων (κλίμακα μεγέθους νησίδας, ποσοστά ανασυνδυασμού κλπ.). Ο υπολογισμός της τιμής της ποιότητας απαιτεί πολύπλοκες προσομοιώσεις και ο χρόνος που αφιερώνεται στην εργασία αυτή μπορεί να είναι μεγαλύτερος από εκείνον της ίδιας της εξελικτικής διαδικασίας. Το κριτήριο αυτό δεν εμφανίζεται στην ΔΜΑ καθώς συνδέεται σε σημαντικό βαθμό με το πρόβλημα. Παρολ΄ αυτά, πρέπει να ληφθεί υπόψη κατά την παραλληλοποίηση ενός ΜΑ.

Επίπεδα : Τα επίπεδα παραλληλισμού που χαρακτηρίζουν την υλοποίηση μας είναι τρία και ορίζονται βάσει του αριθμού των μονάδων επεξεργασίας p και του αριθμού των νησίδων I που αποτελούνται από n άτομα η καθεμία. Πιο συγκεκριμένα

- Αν $p \leq I$ τότε το επίπεδο παραλληλισμού είναι $L_1\left(\left[\frac{I}{p}\right]\right)$.
- Αν $I < p < 2I$ τότε το επίπεδο του παραλληλισμού είναι $L_1(1)//L_0\left(\left[\frac{n}{2}\right]\right)$.
- Αν $2I \leqslant p$ τότε το επίπεδο παραλληλισμού είναι $L_0\left(\left[\frac{n}{\left[\frac{p}{I}\right]}\right]\right)$.

Η κατανομή των I νησίδων σε p μονάδες επεξεργασίας γίνεται με βάσει τα ακόλουθα:

- Αν $p \leqslant I$ τότε $(I \bmod p)$ μονάδες επεξεργασίας χειρίζονται $\left\lceil \frac{I}{p} \right\rceil$ ενώ οι υπόλοιπες μονάδες επεξεργασίας χειρίζονται $\left\lfloor \frac{I}{p} \right\rfloor$.
- Αν $I < p < 2I$ τότε κάθε μονάδα επεξεργασίας χειρίζεται ένα πληθυσμό μεγέθους n ατόμων ή εναλλακτικά ένα υποπληθυσμό μεγέθος $\left\lfloor \frac{n}{2} \right\rfloor$ ή $\left\lceil \frac{n}{2} \right\rceil$.

– Αν $2I \leqslant p$ τότε κάθε μονάδα επεξεργασίας χειρίζεται ένα υποπληθυσμό μεγέθους $\left\lceil \frac{n}{\lceil \frac{p}{I} \rceil} \right\rceil$

Αν $p \leqslant I$, οι νησίδες κατανέμονται στις μονάδες επεξεργασίας χωρίς να επιμερίζονται περαιτέρω. Η θεωρητική επιτάχυνση (με μηδενικό κόστος επικοινωνίας) σε αυτή την περίπτωση είναι $S^{p \leqslant I}_{Iislands}(p) = \frac{I}{\left\lceil \frac{I}{p} \right\rceil}$.
Αν $p > I$, οι μονάδες επεξεργασίας χειρίζονται η κάθεμια ένα μέρος της νησίδας. Δηλαδή μια νησίδα χωρίζεται σε $\left\lfloor \frac{p}{I} \right\rfloor$ τμήματα. Το μέγεθος του μεγαλυτερου υποπληθυσμού ειναι $\left\lceil \frac{n}{\lfloor \frac{p}{I} \rfloor} \right\rceil$ και η θεωρητική επιτάγχυνση (χωρίς κόστος επικοινωνίας) είναι$S^{p>I}_{IsizenIslands}(p) = \frac{I \times n}{\left\lceil \frac{n}{\lfloor \frac{p}{I} \rfloor} \right\rceil}$
Στο τέλος της εκτέλεσης ενός ΜΑ, τα καλύτερα άτομα που εντοπίστηκαν πρέπει να είναι γνωστά τουλάχιστον σε μια μονάδα επεξεργασίας που έχει επιλεχθεί εκ των προτέρων. Η γνώση του καλύτερου ατόμου πρέπει, με τον τρόπο αυτό, να συγκεντρωθεί σε μια μονάδα επεξεργασίας στο τέλος της εκτέλεσης ή να διατηρείται ενημερωμένη κατά τη διάρκεια της εκτέλεσης η μονάδα επεξεργασίας.

Ανταλλαγή Πληροφορίας:Με τη διαδικασία της ανταλλαγής πληροφοριών μεταξύ των πληθυσμών πετύχαμε ελάττωση των μη αποτελεσματικών διαιρέσεων του χώρου αναζήτησης. Η ανταλλαγή της πληροφορίας μεταξύ των υποπληθυσμών τους επιτρέπει να συνεργάζονται ενώ με την χρήση της τοπικής αναζήτησης ενισχύει την εκμετάλλευση ελπιδοφόρων περιοχών του χώρου αναζήτησης κάθως και τη μείωση της πιθανότητας να απωλεσθεί χρήσιμο γενετικό υλικό. Ειδικά στην μεθοδο PMA4 είναι δυνατόν μια από

τις ετερογενείς μεθοδολογίες που εκτελούνται στις διάφορες νησίδες επεξεργαστών να έχει μεγαλύτερη επίδραση στην απόδοση των υπολοίπων.

Εξέλιξη: Η εξέλιξη γενεαλογικής αντικατάστασης απαιτεί συγχρονισμό μεταξύ των διαδοχικών γενεών. Έτσι, είναι πολύ ευαίσθητη στη δίκαιη κατανομή μιας εργασίας. Η εφαρμογή σε ένα δίκτυο υπολογιστών φαίνεται κατάλληλη, αν η τοπολογία του χώρου του ΜΑ μπορεί να απεικονιστεί σε αυτή του δικτύου του υπολογιστή. Παρολ΄ αυτά, δεν είναι συνήθως απαραίτητο να συγχρονίσουμε τις μονάδες επεξεργασίας μεταξύ διαδοχικών γενεών καθώς ο συγχρονισμός αυτός είναι μια ανεπιθύμητη ενέργεια για την απόδοση σε συστημα ανταλλαγής μηνυμάτων.

Η εξέλιξη σταθερής κατάστασης είναι σύγχρονη, αλλάζει, όμως, μόνο μερικές οντότητες e σε κάθε γενιά. Το σύνολο S χρησιμοποιείται ως δεξαμενή όπου επιλέγονται ή αντικαθίστανται τα στοιχεία. Οι πληροφορίες του συνόλου είναι επομένως απαραίτητες. Καθώς το μεγαλύτερο μέρος του υπολογιστικού φορτίου οφείλεται στην εξέλιξη των οντοτήτων ο έλεγχος της καταλληλοτητας των οντοτήτων μπορεί να κατανεμηθεί σε απομακρυσμένες μονάδες επεξεργασίας. Επομένως, η εξέλιξη σταθερής κατάστασης θεωρείται κατάλληλη για το μοντέλο συντονιστή / εργαζομένου που εφαρμόζεται σε κάθε νησίδα. Ο συντονιστής διαχειρίζεται το σύνολο S και ελέγχει τους εργαζόμενους που χειρίζονται τα στοιχεία e. Μεταξύ των συντονιστών και των εργαζομένων ανταλλάσσεται μόνο ένας μικρός αριθμός οντοτητων, τα οποία πρέπει να αλλάξουν, και επομένως το φορτίο επικοινωνίας παραμένει χαμηλό[42].

Ανασυνδυασμός και Μετάλλαξη: Η μέθοδος που ακολουθείται για το στάδιο του ανασυνδυασμού και της μετάλλαξης επιδρά σημαντικά στο φορτίο επικοινωνίας ενός

παράλληλου αλγόριθμου που πρέπει να ανταλλάξει τα στοιχεία αυτά αυξάνεται στην ίδια αναλογία. Συνήθως εφαρμόζεται αλγόριθμος τοπικής αναζήτησης (πχ. κατευθυνόμενη τοπική αναζήτηση) σε όλα σχεδόν τις οντότητες e ταυτοχρόνως. Ο μηχανισμός του ανασυνδυασμού δεν εφαρμόζεται συνήθως σε όλα τα ζευγάρια των ατόμων που έχουν επιλεγεί για αναπαραγωγή. Οι προτεινόμενες θέσεις για την πιθανότητα ανασυνδυασμού είναι $p_c = 0{,}6..0{,}95$ [146, 198, 154, 81, 141].

Οι προτεινομενες τιμές για το ποσοστό μετάλλαξης είναι $p_m = 0,001$, $p_m = 0,01$ και $p_m = [0,005....0,01]$, [191, 161, 16]. Η τιμή της συχνότητας μετάλλαξης είναι αντιστρόφως ανάλογη του μεγέθους πληθυσμού. Ο μηχανισμός της μετάλλαξης βοηθά τον μιμητικό αλγόριθμο να διαφεύγει από τα τοπικά ακρότατα, παρέχοντας μια επιπλέον συνιστώσα τυχαιότητας στη διαδικασία της εξέλιξης [111, 139, 123]. Μια άλλη λειτουργία της μετάλλαξης, η οποία έχει σημασία μόνο στην περίπτωση που έχει προσδιοριστεί αρκετά η βέλτιστη λύση, είναι η δημιουργία μικρών διαταραχών κοντά στην περιοχή του ακροτάτου για την επιτάχυνση της διαδικασίας της σύγκλισης.

Ιστορικό Εξέλιξης: Αν το μέγεθος μιας οντότητας (πληθυσμού, νησίδας) S είναι μεγάλο, ο πρώτος απλός τρόπος παραλληλοποίησης του αλγορίθμου είναι ο καταμερισμός των στοιχείων του S σε υποσύνολα p μεγέθους $[\frac{|S|}{p}]$. Αν το σύνολο S κατατμηθεί σε διάφορες μονάδες επεξεργασίας και αν το ιστορικό του χρησιμοποιείται ως πηγή πληροφοριών, τότε το ιστορικό αυτό θα πρέπει να διατηρείται σταθερό και κάθε μονάδα επεξεργασίας πρέπει να μπορεί να έχει πρόσβαση σε αυτό, όταν είναι απαραίτητο. Το υπολογιστικό κόστος της πρόσβασης αυτής είναι υψηλό, όσον αφορά την επικοινωνία,

όποια τεχνική και αν χρησιμοποιηθεί για να πραγματοποιηθεί. Επίσης, το ιστορικό εξέλιξης αλλάζει σε κάθε γενιά (ή ακόμα και πιο συχνά) αυξάνοντας το φορτίο επικοινωνίας [55]. Η επικοινωνία που απαιτείται για την ανεύρεση των απαιτούμενων πληροφοριών από τους απογόνους είναι ανάλογη προς τον αριθμό των γονέων, την ποσότητα των πληροφοριών που ανταλλάσσεται από τους γονείς και τη συχνότητα των ανταλλαγών αυτών. Το πιθανό αυτό φορτίο επικοινωνίας καθορίζεται από τον αριθμό των γονέων και το ποσοστό ανταλλαγής. Αν οι τιμές είναι χαμηλές, συνιστάται ο διαχωρισμός του συνόλου S.

5.3.1 Μοντέλο Συντονιστής Εργαζόμενος

Η προσέγγιση Συντονιστή / Εργαζομένου είναι κατάλληλη για να επιταχυνθεί το στάδιο της τοπικής αναζήτησης του αλγόριθμου. Το μέγεθος των στοιχείων δεν πρέπει να είναι πολύ σημαντικό σε σύγκριση με την πολυπλοκότητα του αλγορίθμου τοπικής αναζήτησης, ειδάλλως θα ήταν καλύτερο να μην γίνει παραλληλοποίηση αυτού του τμήματος του ΜΑ. Κατά την υλοποίηση του Συντονιστή / Εργαζομένου κάναμε τις ακόλουθες παραδοχές:

- Οι σταθμοί εργασίας έχουν ένα αναγνωριστικό myid και αριθμούνται από 1 έως p
- Οι υποπληθυσμοί κατανέμονται μεταξύ των διαφόρων σταθμών εργασίας και εξελίσσονται τοπικά.

Ο αλγόριθμος που παρατίθεται στο σχήμα 5.2 περιγράφει τον καθορισμό των νησίδων στην υλοποίηση μας και τον καθορισμό της μεθόδου εξέλιξης των νησίδων.

Στις επόμενες υποενότητες θα περιγράψουμε 3 περιπτώσεις υλοποιήσης που βασίζονται στο μοντέλο συντονιστής - εργαζόμενος. Η πρώτη αφορά παραλληλη υλοποίηση των

Καθόρισε k αρχικές νησίδες επεξεργαστών $P^0, ..., P^{k-1}$ και διένειμε ένα αρχιπέλαγος σε k αρχικές νησίδες $pop^0, ..., pop^{k-1}$
Γενεά = 0
Για κάθε νησίδα επεξεργαστών P^i διαίρεσε την νησίδα pop^i σε g υποπληθυσμούς $subpop^j, ..., subpop^{g-1}$
Γενεά = Γενεά + 1
Εφάρμοσε μια από τις μεθόδους (βλέπε 5.3.2, 5.3.3, 5.3.4) σε κάθε pop^i.
Αν έχει σχηματιστεί ο απαραίτητος αριθμός καλών λύσεων της νησίδας pop^i αυτές μεταναστεύουν στο $pop^{(i+1)modk}$
Αν τουλάχιστον μία λύση παρελήφθη από το $pop^{(i-1)modk}$
τότε η λυση που παραλήφθηκε τοποθετείται στο pop^i
Ελέγχεται αν κάποια νησίδα ικανοποιεί τη συνθήκη τερματισμού και αν όχι τότε επιστρέφουμε στο βήμα 3.

Σχήμα 5.2: Μοντέλο Νησίδων Μιμητικών Αλγορίθμων σε Νησίδες επεξεργαστών

ΜΑ με προσομοιουμενη ανόπτηση, η δεύτερη ΜΑ με αποτρεπτική αναζήτηση και η τρίτη ΜΑ με κατευθυνόμενη τοπική αναζήτηση. Τις υλοποιησεις αυτές θα τις ονομάζουμε από εδω και στο εξης PMA1, PMA2, PMA3. Ενώ την τεταρτη περίπτωση κατα την οποία έχουμε μια υβριδικη των παραπάνω υλοποίηση την ονομάζουμε PMA4.

Με βάση την ταξινόμηση του Talbi η περιγραφή του μοντέλου ορίζεται για κάθε μια από τις υλοποιήσεις μας ως εξής:

1. Παραλληλοι ΜΑ με Προσωμοιούμενη Ανόπτηση PMA1 :
   HCH(HRH(EA+LCH(EA(SA)) (het,par,gen)) (het, par, gen) (hom,par, gen) (hom, glo,gen)
2. Παραλληλοι ΜΑ με Αποτρεπτική Αναζήτηση PMA2 :
   HCH(HRH(EA+LCH(EA(TS)) (het,par,gen)) (het, par, gen) (hom,par, gen) (hom, glo,gen)
3. Παραλληλοι ΜΑ με Κατευθυνόμενη Τοπική Αναζήτηση PMA3 :
   HCH(HRH(EA+LCH(EA(GLS)) (het,par,gen)) (het, par, gen) (hom,par, gen) (hom, glo,gen)
4. Παραλληλοι ΜΑ Υβριδικη Περίπτωση PMA4 :
   HCH(HRH(EA+LCH(EA(SA)+HRH(EA+LCH(EA(TS)+HRH(EA+LCH(EA(GLS))

(het,par,gen)) (het, par, gen) (hom, par, gen) (het, glo,gen).

5.3.2 Παράλληλη Υλοποίηση MA με Αποτρεπτική Αναζήτηση - PMA1

1. ***(ΣΥΝΤΟΝΙΣΤΗΣ)*** Υπολογίζουμε τη συνάρτηση καταλληλότητας $f(m)$ για κάθε άτομο σε ένα πληθυσμό.
2. ***(ΣΥΝΤΟΝΙΣΤΗΣ)*** Τα σημεία ταξινομούνται κατά αύξουσα τιμή της αντικειμενικής συνάρτησης και αποθηκεύονται σε ένα διάνυσμα $D = \{m_i, f_i\}$.
3. ***(ΔΙΑΝΟΜΗ ΑΠΟ ΣΥΝΤΟΝΙΣΤΗ ΣΕ ΕΡΓΑΖΟΜΕΝΟΥΣ)*** Τα στοιχεία του D χωρίζονται σε p πληθυσμούς $pop^{[1]}, pop^{[2]}, ..., , pop^{[p]}$ τα οποία και διανέμονται στους εργαζόμενους. Το ιδιο και η αποτρεπτική λίστα Z που στην πρωτη εκτελεση ειναι μηδενική.
4. ***(ΕΡΓΑΖΟΜΕΝΟΙ)*** Για κάθε ομάδα $pop^{[1...p]}$ εφαρμόζεται η διαδικασία της αναπαραγωγής σε κάθε ένα από τους εργαζόμενους.
5. ***(ΕΡΓΑΖΟΜΕΝΟΙ)*** Σε κάθε ομάδα ξεκινάμε από ένα σύνολο λύσεων $m^*_{0...t}$ και παράγουμε $z = 3$ σύνολα γειτονικών λύσεων $M_{0...3}$ και το καλύτερο ποιοτικά εξ αυτών αποτελεί το νέο σύνολο υποψηφιων λύσεων s^*_1, γύρω από το οποίο γεννώνται οι επόμενες γειτονικές λύσεις.
6. ***(ΣΥΝΤΟΝΙΣΤΗΣ Ενημέρωση)*** Μετά την αντικατάσταση του s από το $s+1$ το s μπαίνει στην τοπική απαγορευμένη λίστα του εργαζομένου. Μόλις εξαντληθει

η χωρητικότητα της απαγορευμένης λίστας Z_i, αφαιρούμε το πρώτο σύνολο λύσεων το οποίο είχε αποθηκευτεί πρώτο. Κάθε εργαζόμενος αποστέλει την απαγορευμένη του λίστα Z_i στον συντονιστή για ενημέρωση της ολικής απαγορευμένης λίστας Z η οποια και θα διανεμηθεί στους εργαζομενους στον επομενη γενιά.

7. ***(ΣΥΝΤΟΝΙΣΤΗΣ Ενημέρωση)*** Οι εξελιγμένες ομάδες $pop^{[1]}, pop^{[2]}, pop^{[3]}$, ..., $pop^{[p]}$ επανατοποθετούνται στο διάνυσμα D. Έτσι επιλέγεται η πλέον υποσχόμενη περιοχή, η περιοχή δηλαδή εκείνη στην οποία υπάρχει αυξημένη πιθανότητα να κείται η βέλτιστη λύση. Αυτη η περιοχή γίνεται ο νέος χώρος εφικτών λύσεων.

8. ***(ΕΡΓΑΖΟΜΕΝΟΙ)*** Στην συνέχεια εφαρμοζουμε ένα τελεστή ανασυνδυασμού. Η συχνότητα ανασυνδυασμού είναι p_c=0.6 για μέγεθος πληθυσμού μεγαλύτερο του 300 και p_c=0.9 για μέγεθος πληθυσμού μικρότερο του 100.

9. ***(ΕΡΓΑΖΟΜΕΝΟΙ)*** Μέσω ενός τελεστή μετάλλαξης ορισμένα άτομα του νέου χώρου εφικτών λύσεων αλλάζουν τιμή από 0 σε 1. Αντί για ένα συνολικό ποσοστό μετάλλαξης, μπορούν να διατηρηθούν πιθανότητες μετάλλαξης για κάθε μεταβλητή κάθε ατόμου. Έται κάθε μεταβλητή μπορεί να έχει διαφορετική πιθανότητα μετάλλαξης. Αυτή η πιθανότητα μετάλλαξης μπορεί να κωδικοποιηθεί σε κάθε άτομο σαν επιπλέον πληροφορία και να εξελιχθεί μαζί με το άτομο. Έτσι επιτυγχάνεται η αυτο-προσαρμογή των παραμέτρων μετάλλαξης, ταυτόχρονα με την διερεύνηση του χώρου. Οι απόγονοι που προκύπτουν αντικαθιστούν τα χειρότερα άτομα του πληθυσμού αν η ποιότητα τους είναι τουλάχιστον τόσο καλή όσο του χειρότερου ατόμου του πληθυσμού (εξάλειψη του χειρότερου - elimination of the worst).

10. ***(ΣΥΝΤΟΝΙΣΤΗΣ Ενημέρωση)*** Ο συντονιστής συλλέγει τους υποπληθυσμούς τους επανατοποθετεί στο διάνυσμα D.Η καλύτερη λύση στέλνεται στον συντονιστή $P^{(i+1)modk}$. Αν τουλάχιστον μία λύση παρελήφθη από τον συντονιστή $P^{(i-1)modk}$ τότε η λύση που παραλήφθηκε τοποθετείται στο P^i.

11. ***(ΣΥΝΤΟΝΙΣΤΗΣ)*** Ελέγχεται αν ο τρέχον πληθυσμός ή αν κάποιος από τους πληθυσμούς των άλλων συντονιστών ικανοποιεί ένα τουλάχιστον από τα κριτήρια τερματισμού, δηλαδή η σύγκλιση του αλγορίθμου και το συνολικό πλήθος δοκιμών. Αν κανένα από τα κριτήρια δεν ικανοποιείται, τότε ο αλγόριθμος επαναλαμβάνεται από το 4.

5.3.3 Παράλληλη Υλοποίηση MA με Προσομοιούμενη Ανόπτηση - PMA2

1. ***(ΣΥΝΤΟΝΙΣΤΗΣ)*** Υπολογίζουμε τη συνάρτηση καταλληλότητας $f(m)$ για κάθε άτομο στον πληθυσμό.

2. ***(ΣΥΝΤΟΝΙΣΤΗΣ)*** Επιλέγονται με καποια μεθοδο επιλογής τα σημεία με την καλύτερη και χειρότερη τιμή της αντικειμενικής συνάρτησης και ορίζεται η αρχική θερμοκρασία: $T^0 = f^0_{max} - f^0_{min}$.

   Τα σημεία ταξινομούνται κατά αύξουσα τιμή της αντικειμενικής συνάρτησης και αποθηκεύονται σε ένα διάνυσμα $D = \{m_i, f_i\}$.

3. ***(ΣΥΝΤΟΝΙΣΤΗΣ)*** Τα στοιχεία του D χωρίζονται σε p ομάδες $pop^{[1]}, pop^{[2]}$, ..., $pop^{[p]}$, κάθε μία από τις οποίες περιέχει o σημεία, έτσι ώστε $m^k_i = m_{k+p_{i-1}}$ με

$k = 1, 2, ..., p$ και $i = 1, 2, ..., o$.

4. ***(ΔΙΑΝΟΜΗ ΑΠΟ ΣΥΝΤΟΝΙΣΤΗ ΣΕ ΕΡΓΑΖΟΜΕΝΟΥΣ)*** Οι υποπληθυσμοί $pop^{[1]}, pop^{[2]}, ..., , pop^{[p]}$ διανέμονται στους εργαζόμενους. Το ιδιο και η θερμοκρασία T.

5. ***(ΕΡΓΑΖΟΜΕΝΟΙ)*** Τίθεται $k = 1$ σηματοδοτώντας την έναρξη νέου κύκλου θερμικής ισορροπίας.

6. ***(ΕΡΓΑΖΟΜΕΝΟΙ)*** Προσδιορίζονται το καλύτερο και το χειρότερο σημείο του pop^k , f^m_{min} και f^m_{max} αντίστοιχα.

7. ***(ΕΡΓΑΖΟΜΕΝΟΙ)*** Ελέγχεται αν η τρέχουσα θερμοκρασία του συστήματος ικανοποιεί τη συνθήκη :

   $T^m < \xi[f^m_{max} - f^m_{min}]$ όπου ξ παράμετρος του χρονοδιαγράμματος ανόπτησης.

8. ***(ΕΡΓΑΖΟΜΕΝΟΙ)*** Από το σύνολο των σημείων αυτών $m_1, m_2, ..., m_p$ επιλέγεται μια κορυφή w βάσει του κριτηρίου Metropolis, η οποία και θεωρείται ως χειρότερη.

9. ***(ΕΡΓΑΖΟΜΕΝΟΙ)*** Εφόσον $k < k_{max}$, τίθεται $k \rightarrow k + 1$ και επιστρέφουμε στο βήμα 6. Το μέγιστο πλήθος επαναλήψεων σε κάθε κύκλο θερμικής ισορροπίας λαμβάνει μια μεγάλη τιμή, της τάξης του 1000 ως το 100000.

10. ***(ΕΡΓΑΖΟΜΕΝΟΙ)*** Εφόσον η τρέχουσα θερμοκρασία T είναι μεγαλύτερη από το ελάχιστο όριο T_{min}, μειώνεται με βάση την εξίσωση $T^{k+1} = \lambda T^k$ όπου λ συντελεστής που λαμβάνει τιμές στο διάστημα (0,1).

11. ***(ΕΡΓΑΖΟΜΕΝΟΙ)*** Για κάθε ομάδα $pop^{[1]}, pop^{[2]}, pop^{[3]}, ..., pop^{[p]}$ εφαρμόζεται

ενας τελεστής ανασυνδυασμού και ένας τελεστής μετάλλαξης με συχνότητα p_c=0.3 και p_m=0.05 αντίστοιχα.Η μετάλλαξη έχει σκοπό να εμποδίσει τον αλγόριθμο από το να παγιδευτεί μέσα σε τοπικά ακρότατα του προβλήματος.

12. ***(ΣΥΝΤΟΝΙΣΤΗΣ Ενημέρωση)*** Τα εξελιγμένα δείγματα $pop^{[1]}, pop^{[2]}, pop^{[3]}$, ..., $pop^{[p]}$ επανατοποθετούνται στο διάνυσμα D και στη συνέχεια ταξινομείται το σύνολο των σημείων κατά αύξουσα τιμή της αντικειμενικής συνάρτησης.

13. ***(ΕΡΓΑΖΟΜΕΝΟΙ)*** Στην συνέχεια εφαρμοζουμε ένα τελεστή ανασυνδυασμού στο νέο πληθυσμό. Η συχνότητα ανασυνδυασμού είναι p_c=0.6 για μέγεθος πληθυσμού μεγαλύτερο του 300 και p_c=0.9 για μέγεθος πληθυσμού μικρότερο του 100.

14. ***(ΕΡΓΑΖΟΜΕΝΟΙ)*** Μέσω ενός τελεστή μετάλλαξης ορισμένα άτομα του νέου χώρου εφικτών λύσεων αλλάζουν τιμή από 0 σε 1. Η συχνότητα μετάλλαξης είναι ένας μικρός αριθμός p_m, ο οποίος λαμβάνει τιμές στο διάστημα $[0, ..., 0.2]$ με βήμα 0.005.

15. ***(ΣΥΝΤΟΝΙΣΤΗΣ Ενημέρωση)*** Ο συντονιστής συλλέγει τους υποπληθυσμούς τους επανατοποθετεί στο διάνυσμα D. Η καλύτερη λύση στέλνεται στον συντονιστή $P^{(i+1)modk}$. Αν τουλάχιστον μία λύση παρελήφθη από τον συντονιστή $P^{(i-1)modk}$ τότε η λύση που παραλήφθηκε τοποθετείται στο P^i.

16. ***(ΣΥΝΤΟΝΙΣΤΗΣ)*** Ελέγχεται αν ο τρέχον πληθυσμός ή αν κάποιος από τους πληθυσμούς των άλλων συντονιστών ικανοποιεί ένα τουλάχιστον από τα κριτήρια τερματισμού, δηλαδή η σύγκλιση του αλγορίθμου και το συνολικό πλήθος δοκιμών.

Αν κανένα από τα κριτήρια δεν ικανοποιείται, τότε ο αλγόριθμος επαναλαμβάνεται από το βήμα 6.

5.3.4 Παράλληλη Υλοποίηση MA με Κατευθυνόμενη Τοπική Αναζήτηση - PMA3

1. ***(ΣΥΝΤΟΝΙΣΤΗΣ)*** Υπολογίζουμε τη συνάρτηση καταλληλότητας $f(m)$ για κάθε άτομο σε ένα πληθυσμό.

2. ***(ΣΥΝΤΟΝΙΣΤΗΣ)*** Επιλέγονται με κάποια μεθοδο επιλογής τα σημεία με την καλύτερη και χειρότερη τιμή της αντικειμενικής συνάρτησης. Τα σημεία ταξινομούνται κατά αύξουσα τιμή της αντικειμενικής συνάρτησης και αποθηκεύονται σε ένα διάνυσμα $D = \{m_i, f_i\}$, τέτοιο ώστε το πρώτο στοιχείο του να αντιστοιχεί στο σημείο με την ελάχιστη τιμή της συνάρτησης.

3. ***(ΣΥΝΤΟΝΙΣΤΗΣ)***Τα στοιχεία του D χωρίζονται σε p ομάδες $pop^{[1]}, pop^{[2]}$, $..., pop^{[p]}$, κάθε μία από τις οποίες περιέχει o σημεία, έτσι ώστε $m_i^k = m_{k+p_{i-1}}$ με $k = 1, 2, ..., p$ και $i = 1, 2, ..., o$.

4. ***(ΔΙΑΝΟΜΗ ΑΠΟ ΣΥΝΤΟΝΙΣΤΗ ΣΕ ΕΡΓΑΖΟΜΕΝΟΥΣ)*** Οι υποπληθυσμοί $pop^{[1]}, pop^{[2]}, ..., , pop^{[p]}$ διανέμονται στους εργαζόμενους.

5. ***(ΕΡΓΑΖΟΜΕΝΟΙ)*** Για κάθε μια ομάδα $pop^{[1...p]}$ εφαρμόζουμε τον αλγόριθμο της κατευθυνόμενης τοπικής αναζήτησης GLS

6. ***(ΕΡΓΑΖΟΜΕΝΟΙ)*** Για κάθε ομάδα $pop^{[1]}, pop^{[2]}, pop^{[3]}, ..., pop^{[p]}$ εφαρμόζεται

ενας τελεστής ανασυνδυασμού και ένας τελεστής μετάλλαξης με συχνότητα p_c=0.2 και p_m=0.01 αντίστοιχα.Η μετάλλαξη έχει σκοπό να ενισχύσει τον αλγόριθμο στην προπάθεια να αποφευχθει ο εγκλωβισμός σε τοπικά ακρότατα του προβλήματος.

7. ***(ΕΡΓΑΖΟΜΕΝΟΙ Ενημέρωση ΣΥΝΤΟΝΙΣΤΗ)*** Τα εξελιγμένα δείγματα $pop^{[1]}, pop^{[2]}, pop^{[3]}, ..., pop^{[p]}$ επανατοποθετούνται στο διάνυσμα D και στη συνέχεια ταξινομείται το σύνολο των σημείων κατά αύξουσα τιμή της αντικειμενικής συνάρτησης.

8. ***(ΕΡΓΑΖΟΜΕΝΟΙ)*** Στην συνέχεια εφαρμοζουμε ένα τελεστή ανασυνδυασμού. Η συχνότητα ανασυνδυασμού είναι p_c=0.6 για μέγεθος πληθυσμού μεγαλύτερο του 300 και p_c=0.9 για μέγεθος πληθυσμού μικρότερο του 100.

9. ***(ΕΡΓΑΖΟΜΕΝΟΙ)*** Μέσω ενός τελεστή μετάλλαξης ορισμένα άτομα του νέου χώρου εφικτών λύσεων αλλάζουν τιμή από 0 σε 1. Η συχνότητα μετάλλαξης είναι ένας μικρός αριθμός p_m, ο οποίος λαμβάνει τιμές στο διάστημα $[0, ..., 0.2]$ με βήμα 0.005.

10. ***(ΣΥΝΤΟΝΙΣΤΗΣ Ενημέρωση)*** Ο συντονιστής συλλέγει τους υποπληθυσμούς τους επανατοποθετεί στο διάνυσμα D.Η καλύτερη λύση στέλνεται στον συντονιστή $P^{(i+1)modk}$. Αν τουλάχιστον μία λύση παρελήφθη από τον συντονιστή $P^{(i-1)modk}$ τότε η λύση που παραλήφθηκε τοποθετείται στο P^i.

11. ***(ΣΥΝΤΟΝΙΣΤΗΣ)*** Ελέγχεται αν ο τρέχον πληθυσμός ή αν κάποιος από τους πληθυσμούς των άλλων συντονιστών ικανοποιεί ένα τουλάχιστον από τα κριτήρια τερματισμού, δηλαδή η σύγκλιση του αλγορίθμου και το συνολικό πλήθος δοκιμών.

Αν κανένα από τα κριτήρια δεν ικανοποιείται, τότε ο αλγόριθμος επαναλαμβάνεται από το 4.

5.4 Πειραματική Μεθοδολογία

Για την αξιολόγηση των μεθόδων μας χρησιμοποιήθηκαν οι 14 μαθηματικές συναρτήσεις ελέγχου, τα χαρακτηριστικά των οποίων συνοψίζονται στο Κεφάλαιο 4, καθώς και το πρόβλημα του περιοδεύοντος πωλητή TSP.
Στα πλαίσια της παρούσας εργασίας υλοποιήθηκαν και αξιολογήθηκαν από την κατηγορία των εξελικτικών μεθόδων οι παρακάτω αλγόριθμοι:

- Παράλληλοι Εξελικτικοί Αλγόριθμοι
- Παράλληλοι Γενετικοί Αλγόριθμοι
- Παράλληλοι Μιμητικοί Αλγόριθμοι με Αποτρεπτική Αναζήτηση
- Παράλληλοι Μιμητικός Αλγόριθμοι με Προσομοιούμενη Ανόπτηση
- Παράλληλοι Μιμητικοί Αλγόριθμοι με Καθοδηγούμενη Τοπική Αναζήτηση

Επίσης παρατίθενται προς σύγκριση και τα αποτελέσματα που έδωσαν 2 παράλληλες υλοποιήσεις από την κατηγορία των ευρετικών μεθόδων. Αυτές οι μέθοδοι ήταν οι:

- Παράλληλη Προσομοιούμενη Ανόπτηση [121]
- Παράλληλη Αποτρεπτική Αναζήτηση [9]

Η πλατφόρμα για την πειραματική μας μελέτη είναι μια συστοιχία από 18 σταθμούς εργασίας συνδεδεμένη με δίκτυο 100 Mb/s Fast Ethernet. Συγκεκριμένα, η συστοιχία αποτελείται από έξι ομάδες υπολογιστών. Υπάρχουν 6 επεξεργαστές Pentium 166 MHz με 32 MB μνήμη ένας για κάθε ομάδα και χρησιμοποιουνται σαν συντονιστές των τριών ομάδων αντίστοιχα. Οι υπολοιποι επεξεργαστές είναι Pentium 100MHz με 64 MB μνήμη, οι οποίοι αποτελούν τους εργαζομένους. Οι υλοποιήσεις που αφορούσαν τις μιμητικές προσεγγίσεις έγιναν με την βοήθεια της βιβλιοθήκης PARAMENOAS ή οποία βασίζεται στο κώδικα της βιβλιοθήκης PARsa Lib από το πανεπιστήμιο του Panderborn και της βιβλιοθήκης PGAPack από το Argonne National Laboratory. Πληροφορίες σχετικά με την βιβλιοθήκη PARAMENOAS δίνονται στο Παράρτημα Α της διατριβής. Για την εκτέλεση των παράλληλων εξελικτικών αλγορίθμων χρησιμοποιηθηκε η βιβλιοθήκη NETSTREAM http://neo.lcc.uma.es/ Software/ codezip/ netstream v1.5.tar . Για τους παράλληλους γενετικούς αλγορίθμους χρησιμοποιήσαμε την βιβλιοθήκη PGAPAck http://www-fp.mcs.anl.gov /CCST /research / reports pre1998 /compbio/ stalk/ pgapack.html, για την παράλληλη προσομοιούμενη ανόπτηση την βιβλιοθήκη PARSA http://wwwcs.unipaderborn.de /fachbereich/ AG/ monien/ SOFTWARE/ PARSA/ και για την παράλληλη αποτρεπτική αναζήτηση την βιβλιοθήκη MALLBA http:// www.lsi.upc.es / mallba /information.html

Για κάθε ένα απο τα 14 μαθηματικά προβλήματα ελέγχου απόδοσης οι κατάλληλες τιμές για τον αρχικό πληθυσμό και τα κριτήρια σύγκλισης καθορίστηκαν μετά από ένα σύνολο 10.321.920 εκτελέσεων για κάθε μία από τις παραπάνω 4 υλοποιήσεις (4x2x8x8x8x6x14x5x2x3). Αναλυτικά οι κύριες παράμετροι που εξετάστηκαν ήταν

- Μέγεθος Πληθυσμού : 50,100,150,200,250,300,350,400
- Μέθοδος Αντικατάστασης: Ολική Αντικατάσταση, Σταθερή Αντικατάσταση
- Είδη Ανασυνδυασμού: Ενός σημείου, Δύο σημείων, Ευρετικός Ανασυνδυασμός.
- Πιθανότητα Ανασυνδυασμού: [0.6 - 0.9] με βήμα 0.05
- Πιθανότητα Μετάλλαξης: [0.001,0.02,0.04,0.06,0.08,0.10,0.18,0.20]
- Πιθανότητα Μετανάστευσης:[0.5 - 0.7] με βήμα 0.05
- Αριθμός Νησίδων-Επεξεργαστών: 3,6,9,12,15,18
- Χρήση Ιστορικού

Μέτρο της αποτελεσματικότητας είναι η μέση απόκλιση από τη θεωρητικά βέλτιστη λύση για ένα πλήθος N στοχαστικά ανεξάρτητων εκτελέσεων του αλγορίθμου. Η στοχαστική ανεξαρτησία έγκειται στην εκκίνηση της διαδικασίας βελτιστοποίησης από διαφορετικές τυχαίες αρχικές συνθήκες (αρχική λύση ή πληθυσμό λύσεων). Ο δείκτης αποτελεσματικότητας ορίζεται ως ο λόγος των επιτυχών προς τις συνολικές εκτελέσεις του αλγορίθμου. Μια εκτέλεση κρίνεται επιτυχής εφόσον η τιμή που επιστρέφει βρίσκεται κοντά στη βέλτιστη του εκάστοτε προβλήματος βελτιστοποίησης, δηλαδή:

$$|f_k^{[i]} - f_k^*| < \alpha_k$$

όπου $f_k^{[i]}$ Η η λύση του k προβλήματος κατά την i εκτέλεση του αλγορίθμου,f_k^* η θεωρητικά βέλτιστη τιμή της συνάρτησης και α_k μια αυθαίρετη τιμή ανοχής, εξαρτώμενη από το βαθμό δυσκολίας του εκάστοτε προβλήματος. Στην περίπτωση κατά την οποία όλες

οι δοκιμές συγκλίνουν στο ολικό ακρότατο, ο δείκτης αποτελεσματικότητας είναι 100%. Εφόσον επιλύονται K προβλήματα βελτιστοποίησης, εισάγεται ο μέσος δείκτης αποτελεσματικότητας, ο οποίος ορίζεται ως η μέση τιμή των επιμέρους δεικτών. Οι τιμές ανοχής (βάσει τον οποίων ελέγχθηκε αν η εκτέλεση του αλγορίθμου ήταν επιτυχής) ορίστηκαν ανάλογα με το βαθμό δυσκολίας του εκάστοτε προβλήματος και ήταν οι εξής:

Τιμές ανοχής των συναρτησεων		
f1 $\rightarrow \alpha = 0.1$	f2 $\rightarrow \alpha = 1$	f3 $\rightarrow \alpha = 0.0$
f4 $\rightarrow \alpha = 0.04$	f5 $\rightarrow \alpha = 0.6$	f6 $\rightarrow \alpha = 0.4$
f7 $\rightarrow \alpha = 0.32$	f8 $\rightarrow \alpha = 0.7$	f9 $\rightarrow \alpha = 0.2$
f10 $\rightarrow \alpha = 1.0$	f11 $\rightarrow \alpha = 1$	f12 $\rightarrow \alpha = 0.0$
f13 $\rightarrow \alpha = 0.2$	f14 $\rightarrow \alpha = 0.4$	

Σημειώνεται ότι, με το παραπάνω κριτήριο, ελέγχεται η τιμή και όχι η θέση του ολικού ακρότατου. Με άλλα λόγια, η εκτέλεση του αλγορίθμου θεωρείται επιτυχής εφόσον συγκλίνει σε οποιοδήποτε σημείο, η τιμή της συνάρτησης στο οποίο απέχει από τη θεωρητικά βέλτιστη λιγότερο από την ανοχή a_k. Το κριτήριο αυτό αντιπροσωπεύει καλύτερα την πραγματικότητα, όπου δεν είναι γνωστή η θέση του βέλτιστου αλλά μπορεί να είναι γνωστή, έστω και κατ' εκτίμηση, η βέλτιστη τιμή της αντικειμενικής συνάρτησης.

Ως μέτρο της αποδοτικότητας ενός αλγορίθμου θεωρείται ο χρόνος επίλυσης του προβλήματος βελτιστοποίησης, ανεξάρτητα αν η λύση που προκύπτει είναι ολικά βέλτιστη ή όχι. Ο χρόνος είναι έννοια σχετική διότι εξαρτάται από εξωγενείς παράγοντες, όπως η ταχύτητα του επεξεργαστή και ο χρόνος υπολογισμού της τιμής της αντικειμενικής συνάρτησης. Η επίδραση της πολυπλοκότητας στο συνολικό χρόνο υπολογισμών είναι

αξιόλογη μόνο όταν το πλήθος των μεταβλητών του προβλήματος είναι αρκετά μεγάλο, οπότε οι διαδικασίες προσπέλασης της μνήμης του υπολογιστή απαιτούν σχετκά πολύ χρόνο. Όταν ένα βήμα ενός ΜΑ έχει πολυπλοκότητα $\Theta(n^2)$, ο αριθμός των λειτουργιών που επιτυγχάνονται κατά την διάρκεια του βήματος αυτού είναι ανάλογος του τετραγώνου του αριθμού των ατόμων στη νησίδα[106]. Πράγματι, αν διπλασιασουμε τον αριθμό των νησίδων, ο αριθμός των ατόμων ανά νησί διαιρείται με το δύο και το φορτίο υπολογισμού του βήματος της επιλογής διαιρείται με $2^2 = 4$. Συνεπώς η εκτέλεση ενός μιμητικού αλγορίθμου που αποτελείται από 2 νησίδες των n ατόμων είναι γρηγορότερη από αυτήν ενός ΜΑ με πληθυσμό που αποτελείται από $2 * n$ άτομα. Ο παράγοντας χρόνος είναι δύο φορές καλύτερος.

Πληθυσμός: Το μέγεθος του πληθυσμού αποτελεί ένα από τα βασικά χαρακτηριστικά των παράλληλων ΜΑ που εξετάστηκαν. Το μέγεθος πληθυσμών είναι σταθερό κατά τη διάρκεια της εξέλιξης. Κάθε άτομο του πληθυσμού αντιπροσωπευει ένα σημείο του χώρου των πιθανών λύσεων ενός προβλήματος. Οι τιμές μεγεθών του πληθυσμού που εξετάστηκαν ήταν (50,100,150,200,250,300,350,400).

Μέθοδος Αντικατάστασης: Η εξέλιξη ενός πληθυσμού επιτυγχάνεται με μια σειρά γενεών (επαναλήψεων). Στα συγκεκριμένα πειράματα χρησιμοποιείται το μοντέλο ολικής αντικατάστασης και το μοντέλο σταθερής αντικατάστασης που περιγραφονται στο κεφάλαιο 3. Κάθε άτομο αλλάζει συνεχώς χωρίς να ελέγχει εάν και τα υπόλοιπα ατομα αλλάζουν επίσης. Αρχικά παράγεται ένα πλήθος σημείων μέσα από τον εφικτό χώρο, το οποίο χωρίζεται ανά ομάδες, οι οποίες αναπτύσσονται ανεξάρτητα. Από κάθε ομάδα παράγεται ένας βελτιωμένος πληθυσμός. Σε τακτά διαστήματα ο συνολικός πληθυσμός

χωρίζεται σε νέες ομάδες, εξασφαλίζοντας τη διάδοση των πληροφοριών που έχουν συλλεγεί. Σταδιακά, όλα τα σημεία τείνουν προς το ολικό βέλτιστο του προβλήματος, υπό την προυπόθεση ότι το μέγεθος του αρχικού πληθυσμου είναι αρκετά μεγάλο.

Ανασυνδυασμός: Ο μηχανισμός του ανασυνδυασμού δεν εφαρμόζεται συνήθως σε όλα τα ζευγάρια των ατόμων που έχουν επιλεγεί για αναπαραγωγή. Εάν δεν εφαρμοστεί η μέθοδος αυτή, οι απόγονοι γεννιουνται με απλή αντιγραφή των γονιών. Τρία είδη ανασυνδυασμού χρησιμοποιηθηκαν στα πειράματα

Μετάλλαξη: Η τιμή της συχνότητας μετάλλαξης είναι αντιστρόφως ανάλογη του μεγέθους πληθυσμού. Ο μηχανισμός της μετάλλαξης βοηθά τον μιμητικό αλγόριθμο να διαφεύγει από τα τοπικά ακρότατα, παρέχοντας μια επιπλέον συνιστώσα τυχαιότητας στη διαδικασία της εξέλιξης.

Ιστορικό Εξέλιξης : Σε γενικές γραμμές τα στοιχεία εκείνα που καταγράφουμε στο ιστορικό είναι τα :

- Η καλύτερη και η χειρότερη λύση.
- Το ποσοστό του πληθυσμού που συγκλίνει προς την τρέχουσα καλύτερη λύση.
- Η μέση τιμή της ποιότητας του πληθυσμού.
- Το πλήθος των ανασυνδυασμών και των μεταλλάξεων που έχουν γίνει στο πέρας μιας γενιάς.
- Το πλήθος των γενεών κατα τις οποίες η καλυτερη λύση παραμένει στάσιμη.

Μετανάστευση: Ένας σημαντικός παράγοντας που επηρεάζει τη σχεδίαση και την υλοποίηση μας είναι και η μετανάστευση και εξαρτάται από:

Την συχνότητα μετανάστευσης . Η μετανάστευση θα πρέπει να γίνει αφού έχει σχηματιστεί ο απαραίτητος αριθμός λύσεων που θα ανασυνδυαστούν. Μεταναστεύοντας νωρίτερα είναι συνηθως σπατάλη χρόνου. Η μετανάστευση δεν πρέπει να γίνεται πολύ

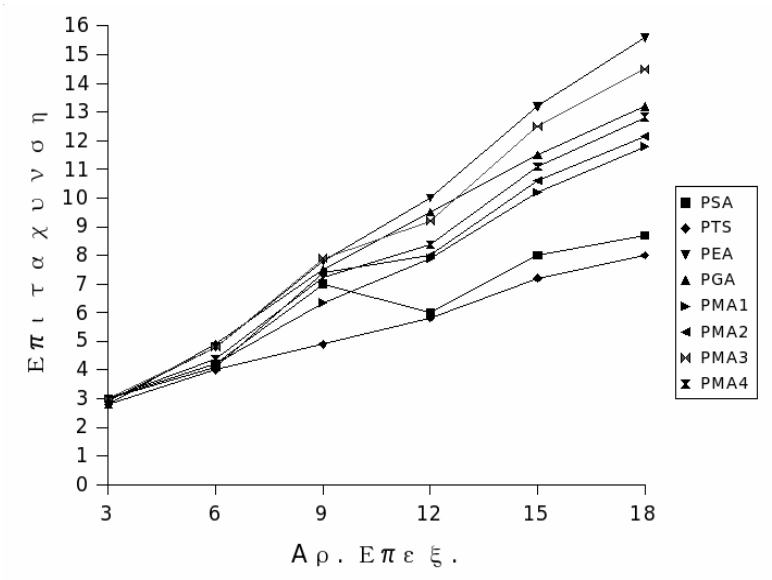
νωρίς πριν σχηματιστουν άτομα με κάποια καλά τμήματα για παραπέρα εξέλιξη. Ετσι η τοπική αναζήτηση πρέπει να προηγείται της μετανάστευσης.

Το ποσοστό μετανάστευσης. Ο αριθμός των γόνων που μεταναστεύουν είναι στενά συνδεδεμένος με την συχνότητα μετανάστευσης. Εάν οι γόνοι είναι καλοί ποιοτικά τότε μπορούμε να πάρουμε καλά αποτελέσματα με την μετανάστευση λίγων.

Την τοπολογία. Ποια είναι η καλυτερη μέθοδος σύνδεσης των υποπληθυσμών: Εάν επιλεγεί τοπολογία με μεγάλη διάμετρο, οι καλές λύσεις θα αργήσουν να διαδοθουν σε ολόκληρο τον πληθυσμό. Εάν η τοπολογία επιτρέπει την ταχεία διάδοση των καλών λύσεων αυτό είναι δυνατό να οδηγήσει στην ταχεία σύγκλιση του αλγορίθμου σε κάποιο τοπικό βέλτιστο.

Αριθμός Νησίδων - Επεξεργαστών : Στα πειράματα μας μελετήσαμε την επίδραση που έχει στη βελτίωση της αποδοτικότητας η χρήση διαφορετικού αριθμού επεξεργαστών (3,6,9,12,15,18) της πειραματικής συστοιχίας του Πανεπιστημίου Μακεδονίας. Κάθε νησίδα απαρτίζεται από ένα σταθμό εργασίας συντονιστή και μια συλλογή από σταθμούς εργασίας εργαζομένους. Ο συντονιστής καθε νησίδας στη γενική περίπτωση χρησιμοποιείται για να διαμερίσει μια ομάδα πληθυσμών σε ένα σύνολο από διάφορους μικρότερους υποπληθυσμούς. Οι εργαζόμενοι κυρίως εκτελούν ένα ακολουθιακό μιμητικό αλγόριθμο πάνω στις αντίστοιχες ομάδες υποπληθυσμών τους.

Κάθε νησίδα επεξεργαστών εκτελεί το ίδιο αλγοριθμικό μοντέλο αν πρόκειται για τις μεθοδους PMA1-PMA3 ή ένα διαφορετικό από τις υπόλοιπες αν πρόκειται για την PMA4. Στην πρώτη περίπτωση έχουμε ομοιογένεια ως προς τα είδη των αλγορίθμων.



Σχήμα 5.3: Συγκριτικά οι επιταχύνσεις για 7 μεθόδους για τις 14 συναρτήσεις

5.5 Πειραματικά αποτελέσματα

5.5.1 Αποτελέσματα για τις 14 συναρτήσεις ελέγχου

Τα καλύτερα αποτελέσματα απόδοσης προκύπτουν για τις μεθόδους PMA1, PMA3, PMA4 ενώ λιγότερο καλά είναι τα αποτελέσματα για την PMA2. Λιγότερο καλά ως προς την ποιότητα ήταν τα αποτελέσματα για τις υπόλοιπες μεθόδους που εξετάσαμε.

Η επιτάχυνση που πραγματοποιείται οφείλεται όχι μόνο στον παραλληλισμό αλλά και στον χρόνο που επιτυγχάνεται με την αύξηση των αριθμών των νησίδων. Από τα πειράματα αυτά διαπιστώνουμε ότι οι χειρότερες επιδόσεις προκύπτουν για πολυ μικρές και μεγάλες νησίδες κάτι το οποίο μπορεί να εξηγηθεί με δύο τρόπους. Πρώτον, μόνο κάποια μέρη του

αλγορίθμου έχουν πολυπλοκότητα $\Theta(n^2)$ ενώ το υπόλοιπο έχει χαμηλότερη πολυπλοκότητα και κατα συνέπεια μόνο ορισμένα τμήματα του αλγορίθμου επηρεάζουν τις τιμές του χρόνου. Δεύτερον ο χρόνος που απαιτείται από την διαδικασία της ανταλλαγής ατόμων εξαρτάται από τον αριθμό των νησίδων και οι υπολογισμοί χρειάζονται λιγότερο χρόνο. Σε όλα τα αποτελέσματα (βλ. σχήματα 5.3,5.10,5.12) παρατηρούμε μια απότομη πτώση των καμπυλών επιτάχυνσης που οφείλεται στην ετερογένεια του περιβάλλοντος μας. Η συμπεριφορά των απλών παράλληλων εξελικτικών αλγορίθμων δεν μπορεί να χαρακτηριστεί πολύ ικανοποιητική, ούτε ως προς την αποτελεσματικότητα ούτε (κυρίως) ως προς την αποδοτικότητα. Σε σχέση όμως με τις σειριακές εκτελέσεις ειδαμε ότι δεν έχουμε καθολική αποτυχία εντοπισμού του ολικού ακρότατου στην 10-διάστατη Rozenbrock όπως και άλλων προβλημάτων στα οποία με τη σειριακή εκτέλεση είχαμε αποτύχει.

Η συμπεριφορά των παράλληλων γενετικών αλγορίθμων είναι λιγότερο ικανοποιητική από αυτήν των παράλληλων εξελικτικών αλγορίθμων. Χαρακτηριστικό είναι το γεγονός της καθολικής αποτυχίας εντοπισμού αρκετών προβλημάτων και στην παράλληλη εκδοχή.

Η παράλληλη προσομοιούμενη ανόπτηση PSA είναι λιγοτερο αποτελεσματική από τις υπόλοιπες υλοποιήσεις. Ποιοτικά τα αποτελέσματα ηταν λίγο καλύτερα από ότι στις σειριακές εκτελέσεις κατι το οποίο οφείλεται στην λειτουργία της μετανάστευσης. Ο καθορισμός των γειτονικών λυσεων απαιτεί μια διαδικασία διακριτοποίησης του εφικτού χώρου κάτι το οποίο κάνει τον αργό σταθμό εργασίας να τελειώνει πάντα τελευταίος και να υπάρχει ένας σημαντικός χρόνος αδράνειας για τους ταχύτερους σταθμούς εργασίας.Ο χρόνος όμως επικοινωνίας είναι με μικρές αποκλίσεις περίπου ο ίδιος εφόσον διατηρούμε το μέγεθος πληθυσμού σταθερό. Το σχήμα 5.3 παρουσιάζει τις επιταχύνσεις

σε σχέση με τον αριθμό των σταθμών εργασιών χρησιμοποιώντας τους αλγορίθμους της προσομοιούμενης ανόπτησης, της αποτρεπτικής αναζήτησης και ενός απλού εξελικτικου αλγορίθμου και συγκρινει αυτές με τις αντιστοιχες επιταχύνσεις των δικών μας μεθόδων για τις 14 συναρτήσεις ελέγχου απόδοσης. Απο αυτά τα αποτελέσματα μπορούμε να δούμε ότι και στις αλλες περιπτώσεις αλγορίθμων είχαμε σημαντική βελτιωση αλλά όχι τόσο μεγάλη όσο στην περίπτωση των μιμητικών αλγορίθμων. Οι πίνακες 5.13 ως 5.20 δείχνουν τους χρόνους εκτέλεσης και τα διαγράμματα του σχήματος 5.5 τους μέσους χρόνους εκτέλεσης των τεσσάρων υλοποιήσεων μας συναρτήση του μεγέθους πληθυσμού σε διαφορετικό αριθμό επεξεργαστών για τα 14 προβλήματα ελέγχου απόδοσης. Τα αποτελέσματα δείχνουν ότι η PMA2 είναι λιγότερο αποτελεσματική καθώς υπάρχει μια πτώση στις καμπυλες των επιταχύνσεων. Το φαινόμενο αυτό οφείλεται στο γεγονός ότι υπάρχει μια ανισορροπία του φορτίου εξαιτίας της φτωχής τεχνικής διαμερισμού των νησίδων σε σχέση με τις υπόλοιπες υλοποιήσεις.

Διατηρήσαμε τον αριθμό των γενεών σταθερό κατά τη διάρκεια της μέτρησης της επιτάχυνσης. Ο αριθμός των γενεών δεν επηρεάζει ως παράμετρος την ακρίβεια της μέτρησης της επιτάχυνσης αλλά κυρίως την ποιότητα των λύσεων για τα διάφορα προβλήματα που ελέγξαμε. Ως εκ τουτου στην διαρκεια των μετρήσεων για την επιτάχυνση την παράμετρο αυτή την διατηρήσαμε σταθερό.

Στη διαρκεια των πειραμάτων μας διαπιστώσαμε ότι η μη συχνή μετανάστευση βοηθά στην αποφυγή της πρόωρης σύγκλισης του συνόλου των πληθυσμών. Επιπλέον η μετανάστευση σε συνδυασμό με την χρήση μιας μεθόδου τοπικής αναζήτησης βοηθά στο να μην χάνεται η ποικιλία του γενετικού υλικού και να μην γίνεται πρόωρη σύγκλιση

σε κάποιο τοπικό βέλτιστο. Η χρήση μιας μεθόδου τοπικής αναζήτησης δημιουργεί νέες ελπιδοφόρες περιοχές αναζήτησης λύσεων (βλ. σχήματα 5.11). Αυτό είναι αρκετά σημαντικό γενικά, καθότι αυτές οι λύσεις αρχικά είναι τοπικές, που είναι και ο συνήθης κανόνας με τις ποικιλίες στη φύση, έτσι ώστε όμοια τροποποιημένα άτομα συχνά αναπαράγονται μεταξύ τους. Εάν ο νέος πληθυσμός είναι επιτυχής στον αγώνα για επιβίωση , θα διαδοθεί αργά, ανταγωνιζόμενος τους υπόλοιπους.

Ένα σημαντικό χαρακτηριστικό των μεθόδων μας ήταν η μη ομαλή σχέση των συντελεστών του χρονοδιαγράμματος ανόπτησης - αποτελεσματικότητας στην μεθοδολογία PMA2. Θεωρητικά, βελτιώνοντας τις τιμές αυτών θα έπρεπε η αποτελεσματικότητα και η αποδοτικότητα να αυξάνει, όπως παρατηρήθηκε σε αλλα θεωρητικά προβλήματα που παρουσιάζονται στην εργασία του Merz [139]. Ωστόσο στα προβλήματα αυτά δεν φάνηκε να υπάρχει μια βέλτιστη τιμή για τους συντελεστές αυτούς με τις οποίες θα επιτυγχάνετο η μέγιστη αποτελεσματικότητα. Το θέμα αυτό χρήζει περαιτέρω διερεύνησης, καθώς είναι άμεσα συναρτώμενο με το ζήτημα της βέλτιστης ρύθμισης των παραμέτρων εισόδου των αλγορίθμων, με στόχο τη μεγιστοποίηση της αξιοπιστίας ενός αλγορίθμου, σε συνδυασμό πάντα με την ελαχιστοποίηση του υπολογιστικού φορτίου.

Στο σχήμα 5.9 βλέπουμε την επίδραση που έχει στην αποδοτικότητα και στην αποτελεσματικότητα της μεθοδου PMA4 ή μη καταγραφή του ιστορικού εξέλιξης σε σχέση με την επίδραση που εχει στην αποδοτικότητα και στην αποτελεσματικότητα η χρήση του ιστορικού εξέλιξης. Στα διαγράμματα του σχήματος 5.10 παρουσιάζουμε τις επιταχύνσεις σε σχέση με τη καταγραφή του ιστορικού εξέλιξης χρησιμοποιώντας της μεθόδους PMA1, PMA2, PMA3, PMA4 για τις 14 συναρτήσεις. Παρατηρούμε ότι υπάρχει μια

σημαντική επίδραση στην απόδοση των υλοποίησεων μας. Συγκεκριμένα βλέπουμε ότι με την καταγραφή του ιστορικού εξέλιξης ο χρόνος επικοινωνίας αυξάνεται. Η επιβάρυνση αυτή όμως δεν φαίνεται να επηρεάζει στην συνολική απόδοση των παράλληλων υλοποίησεων. Παρατηρούμε ότι η PMA2 είναι λιγότερο αποτελεσματική από τις υπόλοιπες τρεις υλοποιήσεις ενώ παρατηρείται και μια απότομη πτώση στους 12 σταθμούς εργασίας στην καμπύλη επιτάχυνσης της υλοποίησης PMA3 με χρήση ιστορικού εξέλιξης κάτι το οποίο οφείλεται στο ότι υπάρχει μια ανισορροπία του φόρτιου εξαιτίας της τεχνικής υλοποίησης της απαγορευμένης λίστας. Ο καθορισμός των γειτονικών λυσεων απαιτεί μια διαδικασία διακριτοποίησης του εφικτού χώρου κάτι το οποίο κάνει τον αργό σταθμό εργασίας να τελειώνει πάντα τελευταίος και να υπάρχει ένας σημαντικός χρόνος αδράνειας για τους ταχύτερους σταθμούς εργασίας σε ένα ετερογενές περιβάλλον. Συνεπώς ο λόγος του χρόνου υπολογισμού προς τον χρόνο επικοινωνίας είναι αρκετά υψηλός.

Παρατηρώντας τα αποτελέσματα βλέπουμε ότι υπάρχει μια αντίστροφη σχέση ανάμεσα στον παράλληλο χρόνο εκτέλεσης και στον αριθμό των σταθμών εργασίας. Ο λόγος του χρόνου υπολογισμού προς το χρόνο επικοινωνίας εξαρτάται ως ένα βαθμό απο την ομοιογένεια των αλγορίθμων που εκτελούνται στις διάφορες νησίδες επεξεργαστών. Χρησιμοποιώντας ετερογενείς αλγορίθμους στην PMA4 παρατηρούμε ότι ο παράλληλος χρόνος εκτέλεσης των υλοποιήσεων μας βελτιώνεται σε σχέση με τον παράλληλο χρόνο εκτέλεσης με χρήση ομοιογενών αλγορίθμων. Ο χρόνος όμως επικοινωνίας είναι με μικρές αποκλίσεις περίπου ο ίδιος εφόσον διατηρούμε το μέγεθος πληθυσμού σταθερό.

Είναι προφανές ότι με την παράλληλη υλοποίηση των μεθοδολογιών του κεφαλαίου 4 μειώθηκε το κόστος υπολογισμου και πραγματοποιηθηκε ενα σχετικά μεγάλο πλήθος

δοκιμών για τον εντοπισμό της ολικά βέλτιστης λύσης με ικανοποιητικη αξιοπιστία.

Οι εκτελέσεις των μεθόδων μας έδειξαν μια ένδειξη σχετικά με το ποια δομή της γειτονιάς του πληθυσμού ενδείκνυται ως η καταλληλότερη για την πραγματοποίηση της επιλογής. Παρόλο όμως που η τοπική επιλογή σε μια γειτονια με δομή δακτυλίου έδωσε καλύτερα αποτελέσματα σε σχέση με τις άλλες δύο περιπτώσεις δεν μπορούμε να συνάγουμε με ασφάλεια το συμπέρασμα ότι σε άλλου είδους προβλήματα μια άλλη δομή δεν θα έδινε καλυτερα ή εφάμιλλα αποτελέσματα. Στις επόμενες παραγράφους θα δούμε την επίδραση που είχε στην αποδοτικότητα και στην αποτελεσματικότητα κάθε μια από τις εξεταζόμενες παραμέτρους και στην συνέχεια θα αναφερθούμε στα αποτελέσματα για κάθεμια από τις εξεταζόμενες μεθόδους.

Επίδραση Πληθυσμού

Η μελέτη αυτής της παραμέτρου έγινε διαφοροποιώντας το μέγεθος του πληθυσμού, το οποίο ορίζεται βάσει του αριθμού των πληθυσμιακών ομάδων και του μεγέθους κάθε νησίδας[167]. Οι πίνακες 5.13 ως 5.20 δείχνουν τους χρόνους εκτέλεσης και τα διαγράμματα του σχήματος 5.5 τους μέσους χρόνους εκτέλεσης των τεσσάρων υλοποιήσεων μας συναρτήση του μεγέθους πληθυσμού σε διαφορετικό αριθμό επεξεργαστών για τα 14 προβλήματα ελέγχου απόδοσης.

Αρχικά η απόδοση των μοντέλων με μέγεθος πληθυσμού 100 είναι καλυτερη από την απόδοση άλλων μεγεθών πληθυσμού, αλλά συνήθως καθηλώνεται ο αλγόριθμος σε τοπικά ελάχιστα. Η χρήση μεγάλων τιμών για το μέγεθος πληθυσμού δεν συνεπάγεται σε

όλες τις περιπτώσεις καλύτερα αποτελέσματα και εκτός τούτου έχει σαν αποτέλεσμα την αύξηση του υπολογιστικού χρόνου. Μια μεγάλη τιμή για το μέγεθος πληθυσμού μπορεί να χρησιμοποιηθεί για να βελτιώσει την απόδοση, δηλαδή όταν υπάρχει διαθέσιμος χρόνος, ενώ μια μικρή τιμή απαιτείται όταν ο στόχος είναι η καλή απόδοση, δηλαδή όταν τα αποτελέσματα απαιτούνται σε σύντομο χρόνο.

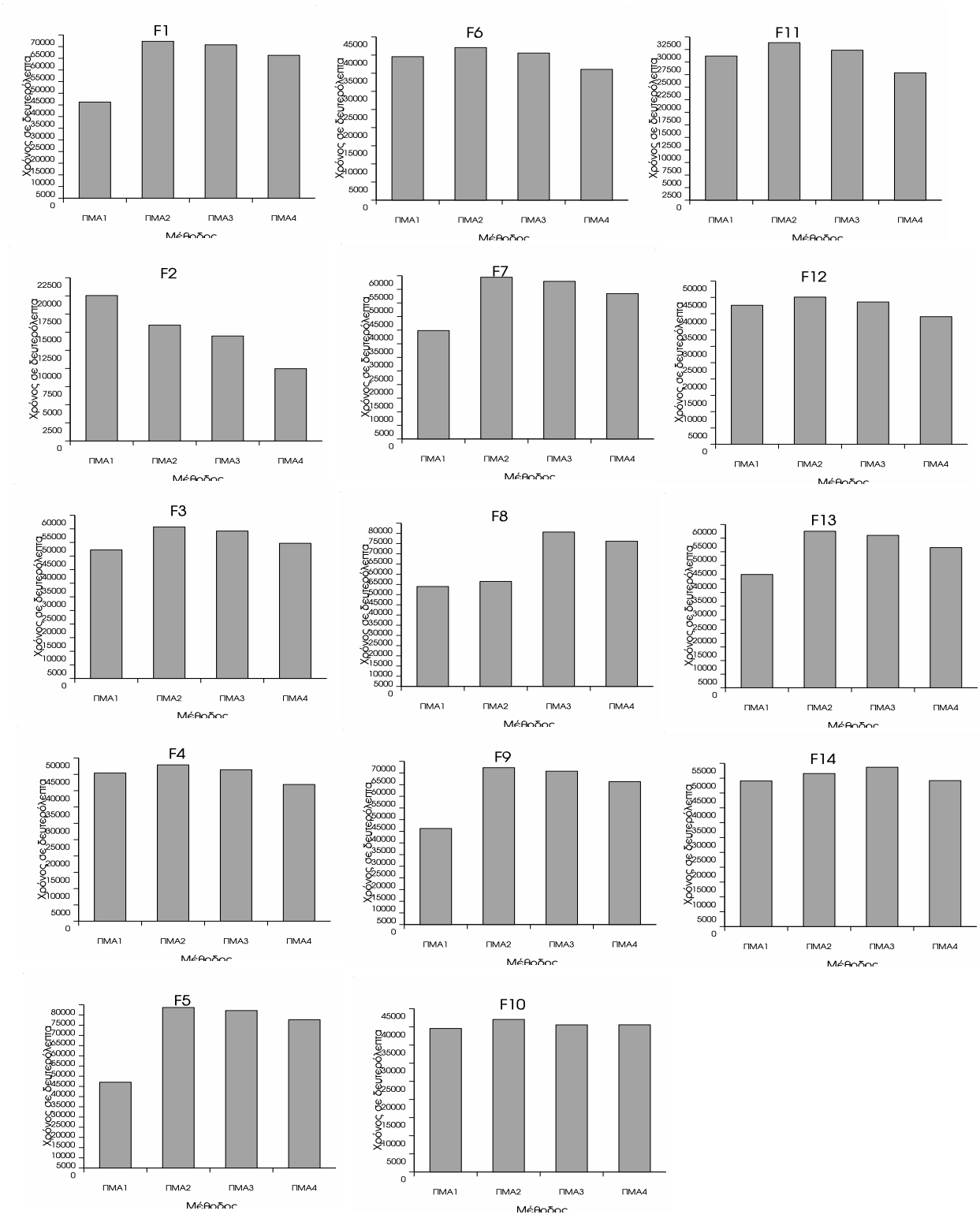
Με την παράλληλη υλοποίηση των μεθοδολογιών του κεφαλαίου 4 μειώθηκε το κόστος υπολογισμού και πραγματοποιήθηκε ενα σχετικά μεγάλο πλήθος δοκιμών για τον εντοπισμό της ολικά βέλτιστης λύσης με ικανοποιητικη αξιοπιστία. Αυξάνοντας το μέγεθος του πληθυσμού βελτιώθηκε η αξιοπιστία του αλγορίθμου, χωρίς η βελτίωση αυτή να είναι τέτοια που να δικαιολογεί τη μεγάλη αύξηση του πλήθους των απαιτούμενων δοκιμών. Παρατηρήθηκε βέβαια αύξηση της αποτελεσματικότητας με αύξηση του πλήθους εκκινήσεων, ωστόσο τα περιθώρια βελτίωσης ήταν πεπερασμένα και εξαρτώμενα τόσο από τις ιδιαιτερότητες του εκάστοτε προβλήματος.

Επίδραση Μεθόδου Αντικατάστασης

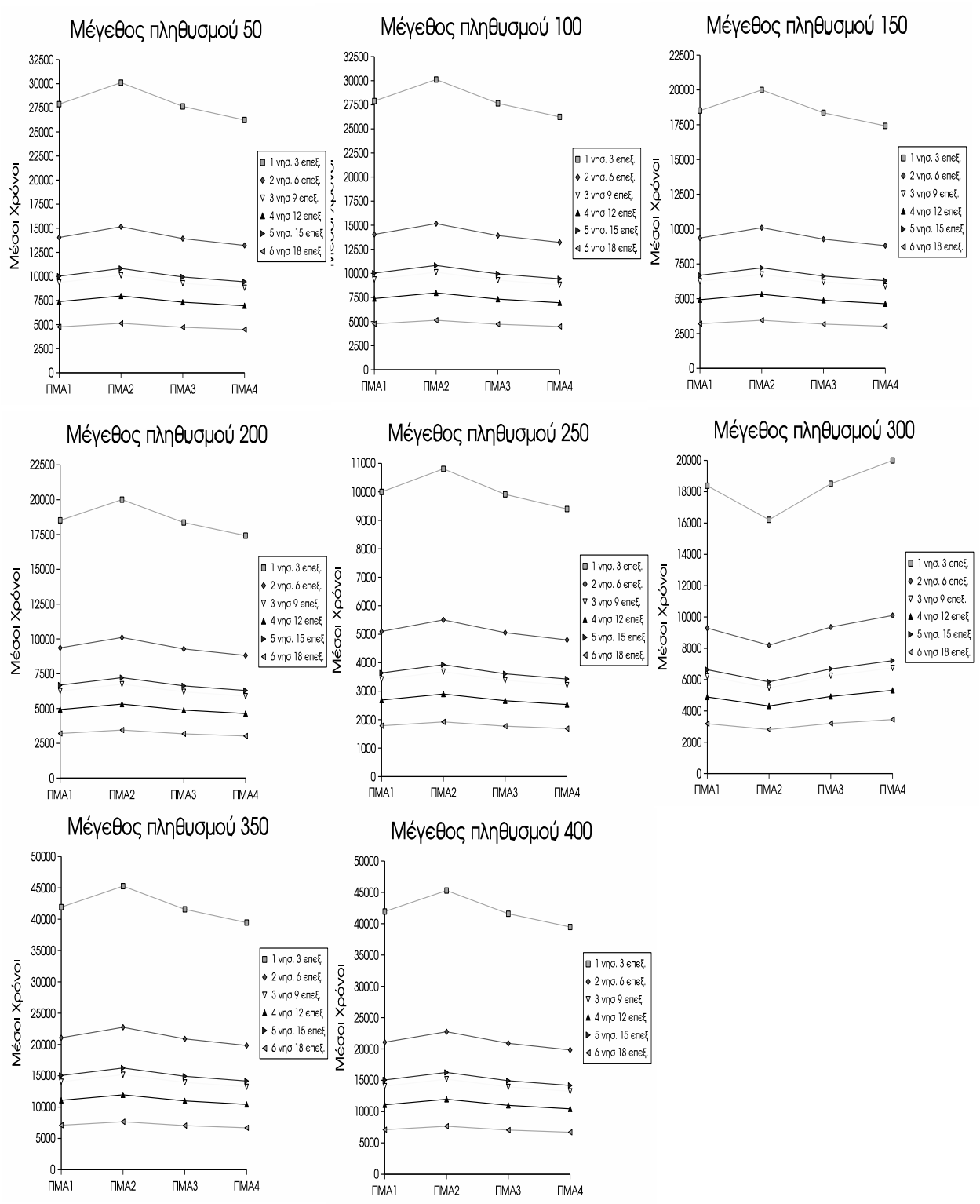
Ελέγξαμε δύο μοντέλα αντικατάστασης του πληθυσμού. Το μοντέλο ολικής αντικατάστασης και το μοντέλο σταθερής αντικατάστασης. Πιο συγκεκριμένα στη μέθοδο γενεαλογικής αντικατάστασης σε κάθε γενιά ο πληθυσμός αντικαθίσταται ολόκληρος ενώ στη δεύτερη περίπτωση μόνο ένα μέρος αυτού. Στην πλειονότητα των περιπτώσεων η μέθοδος γενεαλογικής αντικάταστασης απαιτουσε περισσότερο υπολογιστικό χρόνο από ότι η σταθερής αντικατάστασης (σχ. 5.6).

Επίδραση Ανασυνδυασμού

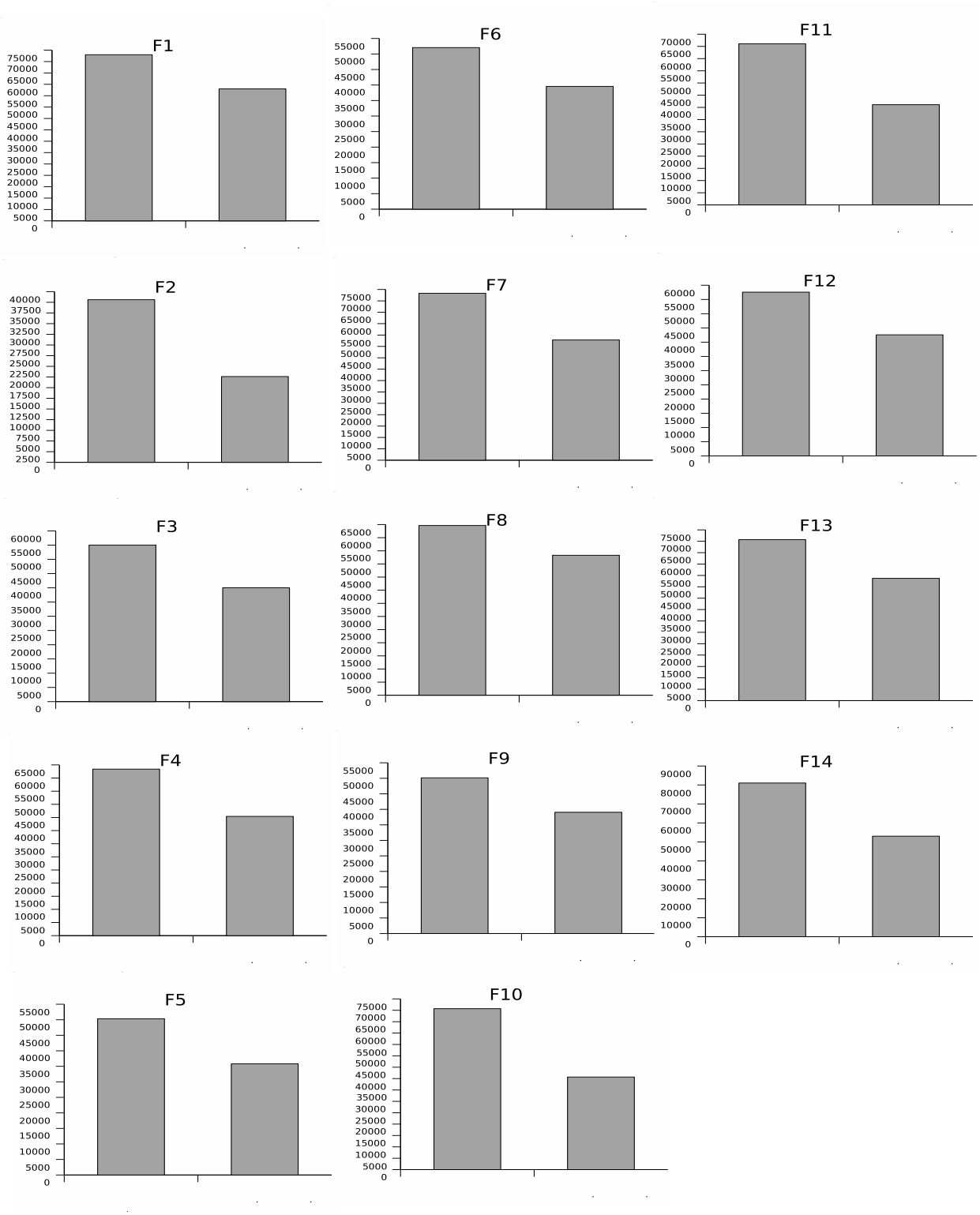
Τα ποσοστά ανασυνδυασμου τάξης $70\% \sim 75\ \%$ βοηθούν στην αποφυγή της πρόωρης σύγκλισης του πληθυσμού. Στις υλοποιήσεις μας έχουμε την δυνατότητα για διερεύνηση διαφορετικών περιοχών του χώρου λύσεων και δημιουργία γενετικής πληροφορίας η οποία καθώς εισάγεται στον πληθυσμό βοηθά στον να προχωρήσει η αναζήτηση του ολικού βέλτιστου. Με την χρήση του ευρετικού ανασυνδυασμού έχουμε καλύτερα ποιοτικά αποτελέσματα σε σχεση με τις περιπτώσεις του ανασυνδυασμού ενός σημείου και δύο σημείων αλλα απαιτήθηκε περισσότερος χρόνος (βλ. πίν. 5.11 και σχ. 5.7). Ο ευρετικός ανασυνδυασμός χρησιμοποιεί τις τιμές της αντικειμενικής συνάρτησης για να καθορίσει την κατεύθυνση της αναζήτησης, ενώ μπορεί να παράγει ένα μόνο απόγονο ή κανένα. Ο τελεστής δημιουργεί ένα μόνο απόγονο $x_3 = r(x_2 - x_1) + x_2$ από τα x_1, x_2 όπου $r \in [0, 1]$ Είναι πιθανό ο τελεστής αυτός να δημιουργήσει μια λύση που δεν ειναι πραγματοποιήσιμη. Σε μια τέτοια περίπτωση επιλέγεται νέο r και δημιουργείται ένας νέος απόγονος. Το τελικό συμπέρασμα που προκύπτει για αυτή τη παράμετρο είναι ότι πολύ μικρά ποσοστά ανασυνδυασμού των γονέων με μικρά μεγέθη πληθυσμού υποβαθμίζουν και άλλο την απόδοση των μεθόδων μας.Για όλες τις εξεταζόμενες εξελικτικές μεθόδους και ιδιαίτερα για τους απλούς παράλληλους εξελικτικούς και γενετικούς αλγορίθμους η πιθανότητα εφαρμογής του τελεστή ανασυνδυασμού απετέλεσε καθοριστικό παράγοντα στη διαδικασία σύγκλισης. Ειδικά στο μοντέλο σταθερής αντικατάστασης παρατηρήθηκε ότι και με μικρότερα ποσοστά ανασυνδυασμού (60% με $65\ \%$) είχαμε καλά αποτελέσματα.



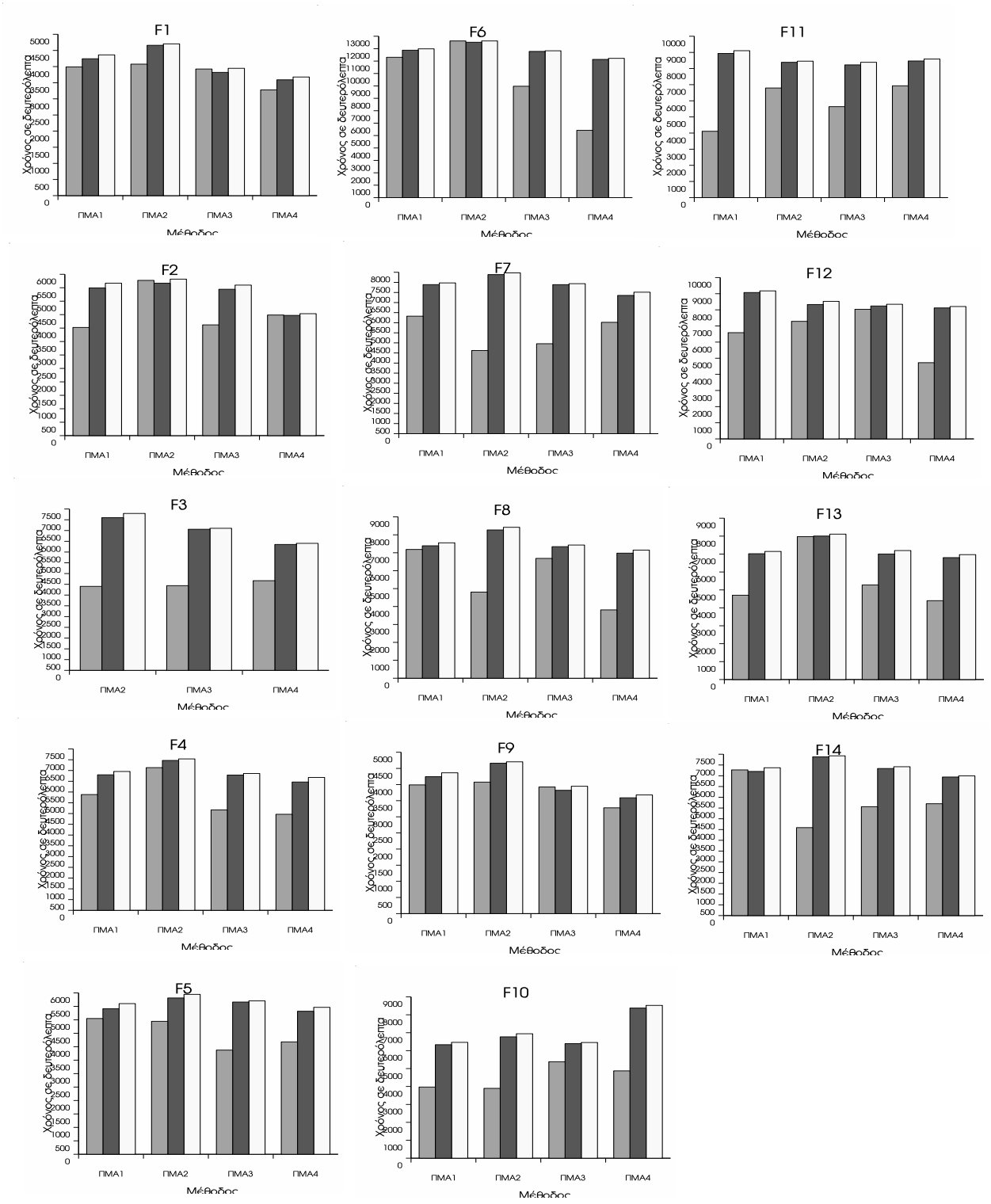
Σχήμα 5.4: Μέσοι χρόνοι εκτελέσεων για κάθε ένα από τα 14 προβλήματα



Σχήμα 5.5: Μέσοι χρόνοι εκτέλεσης για διαφορετικά μεγ. πληθυσμών σε διαφ. αριθμο επεξεργαστών



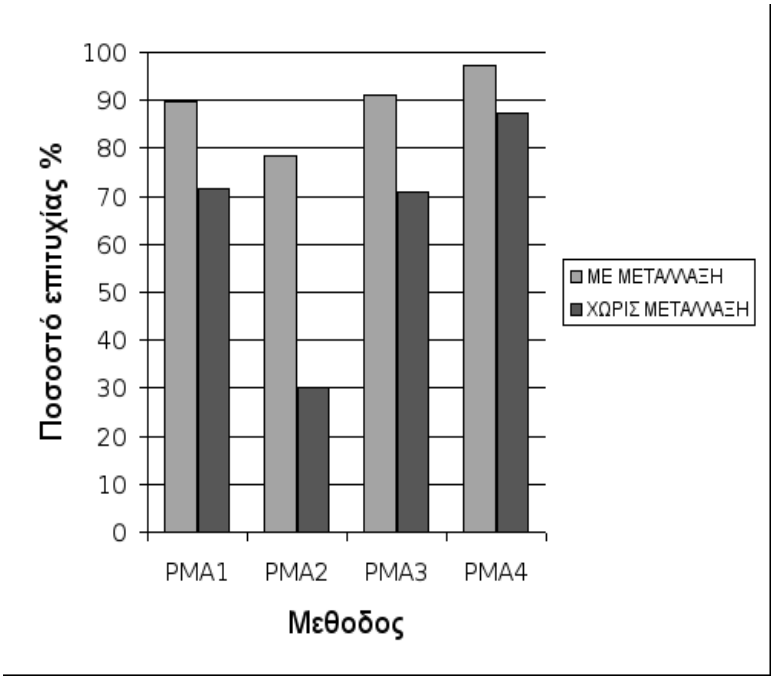
Σχήμα 5.6: Μέσοι χρόνοι εκτέλεσης για δυο μεθοδους αντικατάστασης , Αριστ. Ραβδογραμμ. : Γεν. Αντικατάσταση, Δεξιό Ραβδόγραμμ. : Σταθ. Αντικατάσταση



Σχήμα 5.7: Μέσοι χρόνοι εκτέλεσης για τρία είδη ανασυνδυασμού.Σε κάθε μέθοδο έχουμε τρείς ράβδους. Η πρώτη για ανασυνδ. δύο σημείων , η δεύτερη για ανασυνδ. πολλών σημείων, η τρίτη για ευρετικό ανασυνδ.

Επίδραση Μετάλλαξης

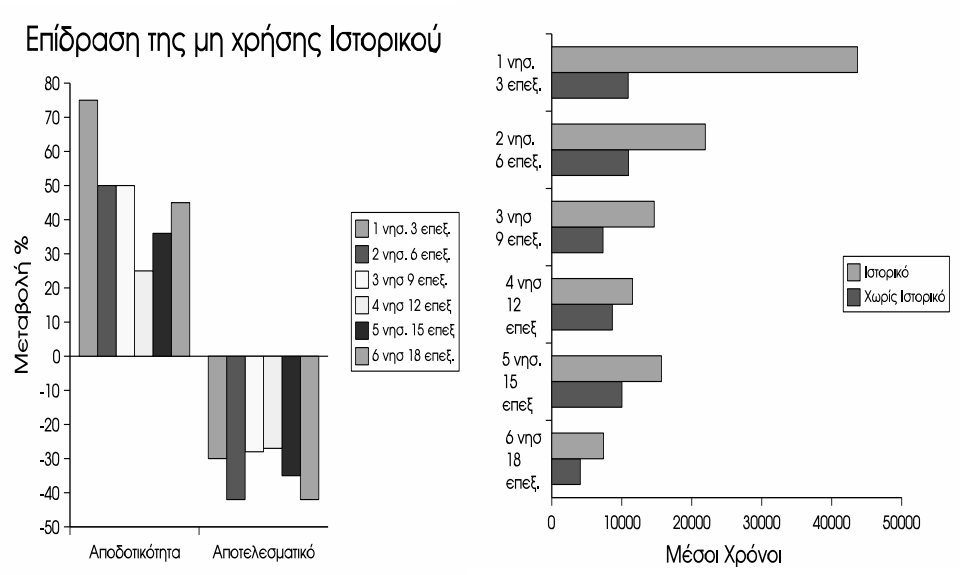
Μέσω του τελεστή μετάλλαξης ορισμένα άτομα του νέου χώρου εφικτών λύσεων αλλάζουν τιμή από 0 σε 1.Στο σχήμα 5.8 βλέπουμε τα ποσοστά επιτυχίας των αποτελεσμάτων με χρήση του τελεστή της μετάλλαξης. Αντί για ένα συνολικό ποσοστό μετάλλαξης διατηρούμε πιθανότητες μετάλλαξης για κάθε μεταβλητή κάθε ατόμου. Έτσι κάθε μεταβλητή έχει διαφορετική πιθανότητα μετάλλαξης. Αυτή η πιθανότητα μετάλλαξης μπορεί να κωδικοποιηθεί σε κάθε άτομο σαν επιπλέον πληροφορία και να εξελιχθεί μαζί με το άτομο. Έτσι επιτυγχάνεται η προσαρμογή των παραμέτρων μετάλλαξης, ταυτόχρονα με την διερεύνηση του χώρου.



Σχήμα 5.8: Ποσοστά επιτυχίας με χρήση μετάλλαξης

Επίδραση Ιστορικού Εξέλιξης

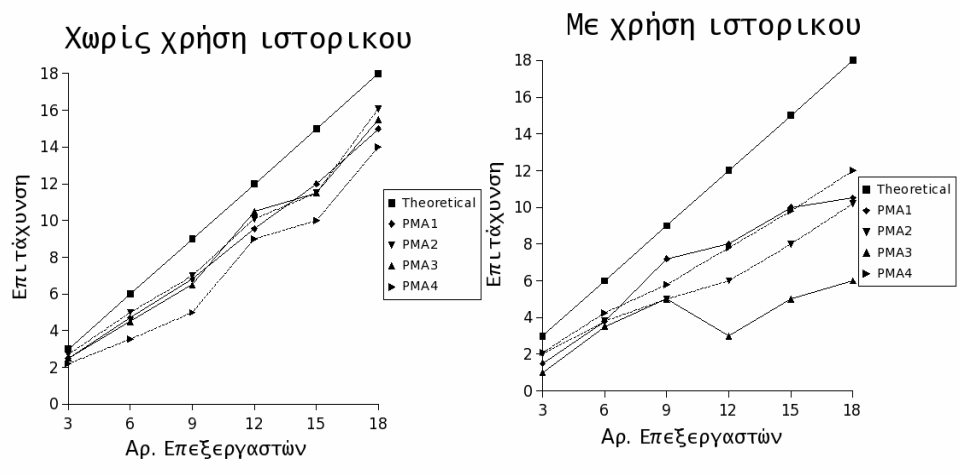
Όταν το ιστορικό εξέλιξης χρησιμοποιείται ως πηγή πληροφοριών, τότε το ιστορικό αυτό θα πρέπει να διατηρείται σταθερό και κάθε μονάδα επεξεργασίας πρέπει να μπορεί να έχει πρόσβαση σε αυτό, όταν είναι απαραίτητο. Το υπολογιστικό κόστος της πρόσβασης αυτής είναι υψηλό, όσον αφορά την επικοινωνία, όποια τεχνική και αν χρησιμοποιηθεί για να πραγματοποιηθεί. Επίσης, το ιστορικό εξέλιξης αλλάζει σε κάθε γενιά (ή ακόμα και



Σχήμα 5.9: Επίδραση της χρήσης ιστορικού εξέλιξης για την PMA4

πιο συχνά) αυξάνοντας το φορτίο επικοινωνίας. Η πιθανή επικοινωνία που απαιτείται για την ανεύρεση των απαιτούμενων πληροφοριών από τους απογόνους είναι ανάλογη προς τον αριθμό των γονέων, την ποσότητα των πληροφοριών που ανταλλάσσεται από τους γονείς και τη συχνότητα των ανταλλαγών αυτών. Το πιθανό αυτό φορτίο επικοινωνίας

καθορίζεται από τον αριθμό των γονέων και το ποσοστό ανταλλαγής.

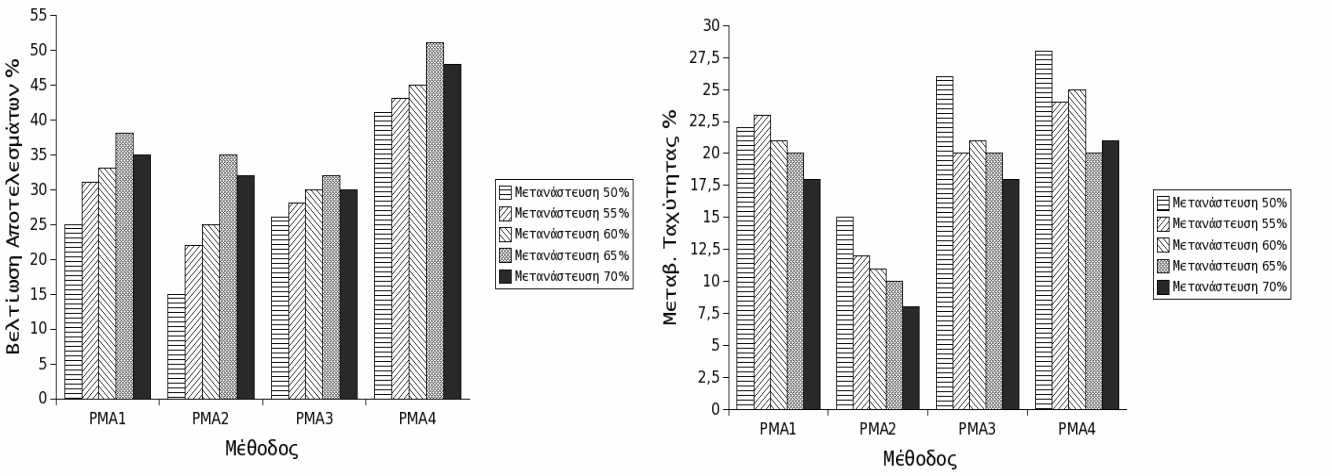


Σχήμα 5.10: Επιτάχυνση των παράλληλων υλοποιήσεων συναρτήση της χρήσης καταγραφης του ιστορικού εξέλιξης (Αριστερά : Χωρίς ιστορικό , Δεξιά : Με ιστορικό)

Επίδραση Μετανάστευσης

Διαπιστώσαμε ότι η μη συχνή μετανάστευση βοηθά στην αποφυγή της πρόωρης σύγκλισης του συνόλου των πληθυσμών. Η χρήση μιας μεθόδου τοπικής αναζήτησης δημιουργεί νέες ελπιδοφόρες περιοχές αναζήτησης λύσεων (βλ. σχήματα 5.11). Αυτό είναι αρκετά σημαντικό γενικά, καθότι αυτές οι λύσεις αρχικά είναι τοπικές. Εάν ο νέος πληθυσμός είναι επιτυχής στον αγώνα για επιβίωση , διαδίδεται αργά, ανταγωνιζόμενος τους υπόλοιπους και κατακτά άτομα στα όρια ενός διαρκώς αυξανόμενου κύκλου. Επιπλέον η μετανάστευση σε συνδυασμό με την χρήση μιας μεθόδου τοπικής αναζήτησης βοηθά στο να μην χάνεται η ποικιλία του γενετικού υλικού και να μην γίνεται πρόωρη σύγκλιση σε

κάποιο τοπικό βέλτιστο.

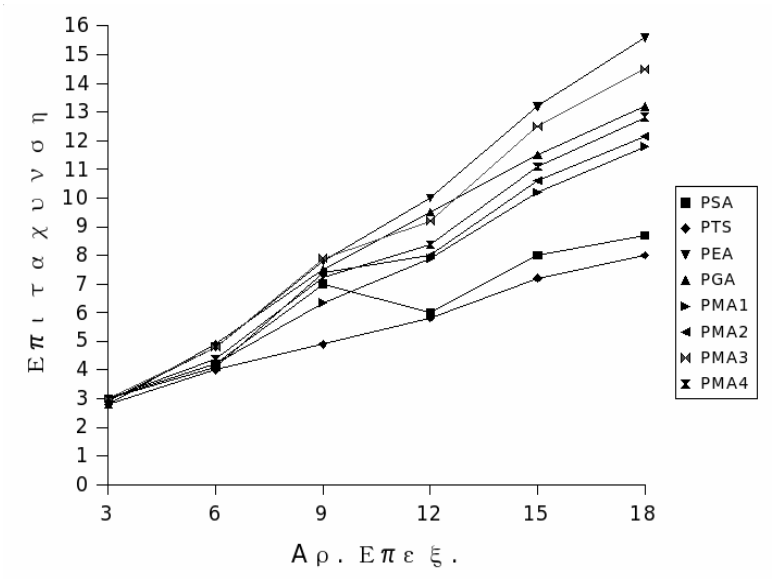


Σχήμα 5.11: Επίδραση της μεταβολής του ποσοστού μετανάστευσης στην ταχύτητα και στην ποιότητα των αποτελεσμάτων

Επίδραση Νησίδων-Επεξεργαστών

Διαπιστώνουμε ότι οι χειρότερες επιδόσεις προκύπτουν για πολυ μικρές και μεγάλες νησίδες κάτι το οποίο οφείλεται στο ότι ο χρόνος που απαιτείται από την διαδικασία της ανταλλαγής ατόμων εξαρτάται από τον αριθμό των νησίδων και οι υπολογισμοί χρειάζονται λιγότερο χρόνο (πιν. 5.21). Στο σχήμα 5.12 βλέπουμε τις επιταχύνσεις σε σχέση με τον αριθμό των σταθμών εργασιών χρησιμοποιώντας τους αλγορίθμους της προσομοιούμενης ανόπτησης, της αποτρεπτικής αναζήτησης και ενός απλού εξελικτικου αλγορίθμου και συγκρίνουμε αυτές με τις αντιστοιχες επιταχύνσεις των δικών μας μεθόδων για τις

14 συναρτήσεις ελέγχου απόδοσης. Παρατηρούμε ότι και στις αλλες περιπτώσεις αλγορίθμων είχαμε σημαντική βελτιωση αλλά όχι τόσο μεγάλη όσο στην περίπτωση των μιμητικών αλγορίθμων.



Σχήμα 5.12: Επιταχύνσεις

Αποτελέσματα για Παράλληλους Εξελικτικούς Αλγορίθμους PEA

Ο κώδικας για την υλοποίηση των Παράλληλων Εξελικτικών Αλγορίθμων πάρθηκε από την διεύθυνση http://neo.lcc.uma.es/ Software/ codezip/netstream v1.5.tar και βασίζεται στο μοντέλο συντονιστή-εργαζομένου. Η συμπεριφορά των απλών παράλληλων εξελικτικών αλγορίθμων δεν μπορεί να χαρακτηριστεί πολύ ικανοποιητική, ούτε ως προς την αποτελεσματικότητα ούτε (κυρίως) ως προς την αποδοτικότητα. Σε σχέση όμως με

τις σειριακές εκτελέσεις ειδαμε ότι δεν έχουμε καθολική αποτυχία εντοπισμού του ολικού ακρότατου στην 10-διάστατη Rozenbrock όπως και άλλων προβλημάτων στα οποία με τη σειριακή εκτέλεση είχαμε αποτύχει.Η αύξηση της τιμής της συχνότητας διασταύρωσης από 60% σε 100% δεν διαφοροποίησε τα αποτελέσματα, ενώ αντίθετα η αύξηση της συχνότητας μετάλλαξης από 0% σε 20% τα διαφοροποίησε σημαντικά, είτε προς τη θετική (όπως στην περίπτωση των συναρτήσεων Step, Quatric, Foxholes, Schwefel, Rastrigin, Powel badly scaled) είτε προς την αρνητική κατεύθυνση (όπως στην περίπτωση της σφαιροειδούς συνάρτησης). Δηλαδή, αυξάνοντας την τυχαιότητα του αλγορίθμου αυξήθηκε η πιθανότητα εντοπισμού της ολικά βέλτιστης λύσης μόνο στην περίπτωση που η αντικειμενική συνάρτηση ήταν έντονα μη γραμμική, διαφορετικά μειώθηκε η αποτελεσματικότητα του αλγορίθμου.

Γενικά, αυξάνοντας το μέγεθος του πληθυσμού βελτιώθηκε η αξιοπιστία του αλγορίθμου, χωρίς η βελτίωση αυτή να είναι τέτοια που να δικαιολογεί τη μεγάλη αύξηση του πλήθους των απαιτούμενων δοκιμών, με εξαίρεση τις περίπτωσεις των συναρτήσεων Griewank και Gulf research and development. Στο πίνακα 5.3 έχουμε για λόγους σύγκρισης και τα απότελέσματα που εδωσε η υλοποίηση του Salomon [186] για 7 από τα προβλήματα που εξετάσαμε. Η αύξηση τόσο της συχνότητας διασταύρωσης όσο και της συχνότητας μετάλλαξης είχε ως συνέπεια μεγαλύτερο υπολογιστικό φόρτο και στα πειράματα του Salomon. Η επίδραση των παραμέτρων εισόδου, δηλαδή του μεγέθους του πληθυσμού και των συχνοτήτων διασταύρωσης και μετάλλαξης, ήταν αρκετά σημαντική, και σε ορισμένες περιπτώσεις καθοριστική.

Συνάρτηση	Τα δικά μας πειράματα	Τα πειράματα του Salomon	Σειριακή Εκτ.
F1 Sphere model	90	92	67
F2 Rosenbrock	70	71	65
F3 Step	73	73	62
F4 Quatric	77	72	65
F5 Foxholes	71	75	61
F6 Rastrigin	38	40	35
F7 Schwefel	40	39	36
Μεση Αποτελεσματικότητα	65.6	66	55.9
F8 Griewang	40		37
F9 Watson	29		23
F10 Ext. Rosenbrock	68		68
F11 Penalty II	64		64
F12 Power badly scaled	85		80
F13 Gulf research and dev.	70		68
F14 Extended Powel singular	55		53
Μεση Αποτελεσματικότητα	62.2		57.8

Πίνακας 5.3: Ποσοστά Επιτυχίας Παράλληλων Εξελικτικών Αλγορίθμων για 1000 δοκιμές. Τα αποτελέσματα της τρίτης στήλης πάρθηκαν από την εργασία [186]

Αποτελέσματα για Παράλληλους Γενετικούς Αλγορίθμους PGA

Ο κώδικας για την υλοποίηση των Παράλληλων Εξελικτικών Αλγορίθμων πάρθηκε από την διεύθυνση http://www fp.mcs.anl.gov/ CCST/research/reports pre1998/comp bio/stalk/ pgapack.html και βασίζεται στο μοντέλο συντονιστή-εργαζομένου. Από τα αποτελέσματα φαίνεται ότι η συμπεριφορά των παράλληλων γενετικών αλγορίθμων είναι λιγότερο ικανοποιητική από αυτήν των παράλληλων εξελικτικών αλγορίθμων. Χαρακτηριστικό είναι το γεγονός της καθολικής αποτυχίας εντοπισμού αρκετών προβλημάτων και στην παράλληλη εκδοχή.

Συνάρτηση	Τα πειράματα μας	Πειράματα του Levin	Σειρ. Εκτ.
F1 Sphere model	85	82	63
F2 Rosenbrock	76	78	67
F3 Step	71	73	62
F4 Qutric	80	77	63
F5 Foxholes	72	70	61
F6 Rastrigin	28	30	25
F7 Schwefel	38	42	36
F8 Griewang	42	40	32
F9 Watson	38	43	23
F10 Ext. Rosenbrock	69	70	68
Μεση Αποτελεσματικότητα	59.9	60.5	50
F11 Penalty II	68		64
F12 Powel bandly scaled	77		70
F13 Gulf research and development	66		59
F14 Ext. Powel Sing.	62		53
Μεση Αποτελεσματικότητα	62.3		53.3

Πίνακας 5.4: Ποσοστά Επιτυχίας Παράλληλων Γενετικών Αλγορίθμων για 1000 δοκιμές. Τα αποτελέσματα της τρίτης στήλης πάρθηκαν από την εργασία [132]

Η χρήση μεγάλων τιμών για το μέγεθος πληθυσμού δεν συνεπάγεται σε όλες τις περιπτώσεις καλύτερα αποτελέσματα και εκτός τούτου έχει σαν αποτέλεσμα την αύξηση του υπολογιστικού χρόνου. Αρχικά η απόδοση των μοντέλων με μέγεθος πληθυσμού 150 είναι καλυτερη από την απόδοση άλλων μεγεθών πληθυσμού, αλλά συνήθως καθηλώνεται ο αλγόριθμος σε τοπικά ελάχιστα. Η αύξηση τόσο της συχνότητας διασταύρωσης όσο και της συχνότητας μετάλλαξης είχε ως συνέπεια μεγαλύτερο υπολογιστικό φόρτο. Μια μεγάλη τιμή για το μέγεθος πληθυσμού μπορεί να χρησιμοποιηθεί για να βελτιώσει την απόδοση, δηλαδή όταν υπάρχει διαθέσιμος χρόνος, ενώ μια μικρή τιμή απαιτείται όταν ο στόχος είναι η καλή απόδοση , δηλαδή όταν τα αποτελέσματα απαιτούνται σε σύντομο

Συνάρτηση	Ποσοστά Επιτυχίας	Σειριακές Εκτ.
F1 Sphere model	55	51
F2 Rosenbrock	57	58
F3 Step	58	58
F4 Quatric	58	55
F5 Foxholes	72	71
F6 Rastrigin	60	56
F7 Schwefel	64	60
F8 Griewang	39	37
F9 Watson	78	75
F10 Ext. Rosenbrock	50	48
F11 Penalty II	52	51
F12 Power bandly scaled	54	51
F13 Gulf research and dev.	64	64
F14 Extended Powel singular	54	54
Μεση Αποτελεσματικότητα	58.2	56.3

Πίνακας 5.5: Ποσοστά Επιτυχίας Παράλληλης Προσομοιούμενης Ανόπτησης για 1000 δοκιμές.

χρόνο. Διαπιστώσαμε ότι η μη συχνή μετανάστευση βοηθά στην αποφυγή της πρόωρης σύγκλισης του συνόλου των πληθυσμών. Στο πίνακα 5.4 έχουμε για λόγους σύγκρισης και τα απότελέσματα που εδωσε η υλοποίηση του Levin [132] για 10 από τα προβλήματα που εξετάσαμε.

Αποτελέσματα για Παράλληλη Προσομοιούμενη Ανόπτηση PSA

Για την υλοποίηση της μεθόδου αυτής χρησιμοποιήθηκε το λογισμικό PARSA το οποίο πάρθηκε από την διεύθυνση http://wwwcs.unipaderborn.de / fachbereich/ AG/ monien/ SOFTWARE/ PARSA/ και βασίζεται στο μοντέλο συντονιστή-εργαζομένου. Η

θερμοκρασία του συστήματος τίθεται ίση με τη διαφορά μεταξύ της μέγιστης και ελάχιστης τιμής της αντικειμενικής συνάρτησης για κάθε υποπληθυσμό. Κατα συνέπεια, υπάρχει μη μηδενική πιθανότητα επιλογής οποιουδήποτε σημείου του πληθυσμού για αντικατάσταση με εξαίρεση βέβαια το αρχικό βέλτιστο. Η προσαρμογή της αρχικής θερμοκρασίας στα χαρακτηριστικά του εκάστοτε προβλήματος χωρίς να απαιτείται ο ορισμός της από τον χρήστη είναι και μια από τις καινοτομίες αυτής της υλοποίησης[121]. Η χρήση του ελάχιστου δυνατού μεγέθους πληθυσμού σε 4 περιπτώσεις συναρτήσεων Sphere model, 2-Rozenbrock, 2-Extenden Rozenbrock, Griewang αποδειχτηκε ανεπαρκής για την εξασφάλιση υψηλού σχετικά ποσοστού επιτυχίας του αλγορίθμου σε σχέση με τις υπόλοιπες μεθόδους. Τα αποτελέσματα δεν εμφάνισαν κάποια βελτίωση ποιοτικά σε σχέση με τις σειριακές εκτελέσεις. Στα αποτελέσματα παρατηρούμε ότι η PSA είναι λιγοτερο αποτελεσματική από τις υπόλοιπες υλοποιήσεις. Ποιοτικά τα αποτελέσματα ηταν λίγο καλύτερα από ότι στις σειριακές εκτελέσεις κατι το οποίο οφείλεται στην λειτουργία της μετανάστευσης. Ο καθορισμός των γειτονικών λυσεων απαιτεί μια διαδικασία διακριτοποίησης του εφικτού χώρου κάτι το οποίο κάνει τον αργό σταθμό εργασίας να τελειώνει πάντα τελευταίος και να υπάρχει ένας σημαντικός χρόνος αδράνειας για τους ταχύτερους σταθμούς εργασίας.Ο χρόνος όμως επικοινωνίας είναι με μικρές αποκλίσεις περίπου ο ίδιος εφόσον διατηρούμε το μέγεθος πληθυσμού σταθερό.

Αποτελέσματα για Παράλληλη Αποτρεπτική Αναζήτηση PTS

Για την υλοποίηση της μεθόδου αυτής χρησιμοποιήθηκε το λογισμικό MALLBA το οποίο πάρθηκε από την διεύθυνση http:// www.lsi.upc.es / mallba /information.html και βασίζεται στο μοντέλο συντονιστή-εργαζομένου. Η μέθοδος εντόπισε λύσεις εμφάνισε καλή επίδοση στη Schwefel και στη Griewang, σχετικά μέτρια επίδοση στα προβλήματα Rastrigin, Watson, Penalty II, Powel badly scaled και πιο χαμηλή επίδοση παρουσίασε στο 10-διάστατο πρόβλημα Rozenbrock, Gulf research and development και Extenden powel singular (βλ. πίνακα 5.6). Και σε αυτήν την μέθοδο είναι προφανές ότι με χρήση μικρού μεγέθους πληθυσμού απαιτείται μικρότερος αριθμός δοκιμών για τον εντοπισμό της βέλτιστης λύσης, και συνεπώς βελτιώνεται σημαντικά η αποδοτικότητα του αλγορίθμου. Στην παράλληλη εκδοχή της μεθοδου έχουμε σημαντική βελτίωση του χρόνου που απαιτήθηκε αλλά όχι σημαντική βελτίωση ως πρός την ποιότητα των αποτελεσμάτων.

Αποτελέσματα για την PMA1

Η μέθοδος PMA1 αντιμετώπισε με σχετική επιτυχία αρκετά από τα προβλήματα. Απαιτήθηκε η θεώρηση μεγαλύτερου πληθυσμού όσο αυξανόταν ο βαθμός δυσκολίας του προβλήματος. Έτσι, ενώ ήταν επαρκής μία και μόνο νησίδα για την εύρεση του ελαχίστου της διδιάστατης Rozenbrock και της Extended Rozenbrock, απαιτήθηκαν τέσσερις νησίδες για την επίλυση των ίδιων προβλήματων στις 10 διαστάσεις, ενώ οι 4 νησίδες δεν ήταν επαρκείς για την επίλυση των προβλήματων Penalty II, Rastrigin, Powel badly

Συνάρτηση	Ποσοστά Επιτυχίας	Σειριακές Εκτ.
F1 Sphere model	71	65
F2 Rosenbrock	75	62
F3 Step	72	63
F4 Quatric	72	61
F5 Foxholes	72	62
F6 Rastrigin	68	63
F7 Schwefel	50	51
F8 Griewang	52	52
F9 Watson	58	53
F10 Ext. Rosenbrock	52	51
F11 Penalty II	49	49
F12 Power bandly scaled	49	40
F13 Gulf research and dev.	46	41
F14 Extended Powel singular	50	48
Μεση Αποτελεσματικότητα	59.7	54.7

Πίνακας 5.6: Ποσοστά Επιτυχίας Παράλληλης Αποτρεπτικής Αναζήτησης για 1000 δοκιμές.

scaled με ικανοποιητική αξιοπιστία. Η ταχύτητα της μεθόδου παρουσίασε σημαντικές διαφορές, ανάλογα με το βαθμό δυσκολίας του εκάστοτε προβλήματος και ήταν προφανώς ανάλογη του αριθμού των επεξεργαστών. Ο εντοπισμός του ολικού ακρότατου των πιο εύκολων συναρτήσεων ελέγχου, όπως της 10-διάστατης σφαιροειδούς, απαίτησε μικρό αριθμό δοκιμών, σε αντίθεση με πολυπλοκότερες συναρτήσεις, όπως για παράδειγμα η Rastrigin και η διδιάστατη Rozenbrock, για τις οποίες χρειάστηκαν μία ως δύο τάξεις μεγέθους περισσότερες δοκιμές.

Όσον αφορά τις συνάρτησεις Griewang και τη Watson, το υπερβολικά μεγάλο πλήθος δοκιμών είχε ως αίτιο την αδυναμία διαφυγής του αλγορίθμου από κορυφογραμμές. Στην περίπτωση της διδιάστατης Rozenbrock, η PMA1 κατόρθωσε, έστω και με υπερβολικά

Συνάρτηση	Ποσοστά Επιτυχίας	Σειριακές Εκτ.
F1 Sphere model	89	80
F2 Rosenbrock	91	80
F3 Step	88	90
F4 Quatric	88	78
F5 Foxholes	90	61
F6 Rastrigin	88	79
F7 Schwefel	90	77
F8 Griewang	71	30
F9 Watson	90	66
F10 Extended Rosenbrock	89	95
F11 Penalty II	89	53
F12 Power Bandly scaled	90	53
F13 Gulf research and dev.	90	58
F14 Extended powel singular	91	66
Μεση Αποτελεσματικότητα	94.6	88.29

Πίνακας 5.7: Ποσοστά Επιτυχίας PMA1 για 1000 δοκιμές.

μεγάλο αριθμό βημάτων, να προχωρήσει κατά μήκος της παραβολοειδούς χαράδρας και να συγκλίνει κοντά στο σημείο ελαχίστου.

Πολύ καλή ήταν και η απόδοση της μεθόδου αυτής λαμβάνοντας υπόψη και το βαθμό δυσκολίας του προβλήματος στην περίπτωση της 10-διάστατης συνάρτησης Griewank.

Αποτελέσματα για την PMA2

Προφανώς, απαιτήθηκε η θεώρηση μεγαλύτερου πληθυσμού όσο αυξανόταν ο βαθμός δυσκολίας του προβλήματος.Η χρήση του ελάχιστο δυνατού μεγέθους πληθυσμού σε 4 περιπτώσεις συναρτήσεων Sphere model, 2-Rozenbrock, 2-Extenden Rozenbrock, Griewang αποδειχτηκε επαρκής για την εξασφάλιση υψηλού σχετικά ποσοστού επιτυχίας του αλγορίθμου. Είναι προφανές ότι με χρήση μικρου μεγέθους πληθυσμού απαιτείται

Συνάρτηση	Ποσοστά Επιτυχίας	Σειριακές Εκτελ.
F1 Sphere model	77	71
F2 Rosenbrock	78	78
F3 Step	78	78
F4 Quatric	80	76
F5 Foxholes	80	71
F6 Rastrigin	80	56
F7 Schwefel	77	60
F8 Griewang	80	37
F9 Watson	79	75
F10 Extended Rosenbrock	77	68
F11 Penalty II	79	71
F12 Powel bandly scaled	78	71
F13 Gulf research and dev.	79	64
F14 Extended Powel singular	77	74
Μεση Αποτελεσματικότητα	78.2	67.86

Πίνακας 5.8: Ποσοστά Επιτυχίας PMA2 για 1000 δοκιμές.

μικρότερος αριθμός δοκιμών για τον εντοπισμό της βέλτιστης λυσης, και συνεπώς βελτιώνεται σημαντικά η αποδοτικότητα του αλγορίθμου.

Στα αποτελέσματα παρατηρούμε ότι η PMA2 είναι λιγοτερο αποτελεσματική από τις υπόλοιπες τρεις υλοποιήσεις παράλληλων μιμητικών αλγορίθμων. Ο καθορισμός των γειτονικών λύσεων απαιτεί μια διαδικασία διακριτοποίησης του εφικτού χώρου κάτι το οποίο κάνει τον αργό σταθμό εργασίας να τελειώνει πάντα τελευταίος και να υπάρχει ένας σημαντικός χρόνος αδράνειας για τους ταχύτερους σταθμούς εργασίας.Ο χρόνος όμως επικοινωνίας είναι με μικρές αποκλίσεις περίπου ο ίδιος εφόσον διατηρούμε το μέγεθος πληθυσμού σταθερό.

Αποτελέσματα για την PMA3

Από τα αποτελέσματα γίνεται φανερό ότι η μέθοδος παρουσίασε πολύ καλή επίδοση, κυρίως ως προς την αποτελεσματικότητα αλλά και ως προς την αποδοτικότητα. Συγκεκριμένα, εντόπισε λύσεις της τάξης του 95% για 5 από τις συναρτήσεις ελέγχου, ενώ εμφάνισε πολύ υψηλή επίδοση στη Schwefel και στη Griewang, σχετικά καλή επίδοση στα προβλήματα Rastrigin, Watson, Penalty II, Powel badly scaled και σχετικά πιο χαμηλή επίδοση παρουσίασε στο 10-διάστατο πρόβλημα Rozenbrock, Gulf research and development και Extenden powel singular (βλ. πίνακα 5.9). Και σε αυτήν την μέθοδο είναι προφανές ότι με χρήση μικρού μεγέθους πληθυσμού απαιτείται μικρότερος αριθμός δοκιμών για τον εντοπισμό της βέλτιστης λύσης, και συνεπώς βελτιώνεται σημαντικά η αποδοτικότητα του αλγορίθμου.

Αποτελέσματα για την PMA4

Από τα αποτελέσματα φαίνεται ότι η PMA4 αντιμετώπισε με μεγαλη επιτυχία από 80% $\sim$ 100% τα περισσότερα από τα προβλήματα. Φαίνεται ότι η μεθοδος είναι πράγματι μια σχετικά καλυτερη μέθοδος από αυτή του απλού εξελικτικού αλγορίθμου, αφού αντιμετώπισε με καλύτερο ποσοστό σχεδόν τις περισσότερες από τις συναρτήσεις ελέγχου (με εξαίρεση την 10-διαστατη Extended Rozenbrock και την Powel badly scaled). Απαιτήθηκε η θεώρηση μεγαλύτερου πληθυσμού όσο αυξανόταν ο βαθμός δυσκολίας του προβλήματος.

Συνάρτηση	Ποσοστά Επιτυχίας	Σειριακές Εκτελ.
F1 Sphere model	95	95
F2 Rosenbrock	92	92
F3 Step	92	95
F4 Quatric	96	95
F5 Foxholes	95	96
F6 Rastrigin	96	90
F7 Schwefel	90	79
F8 Griewang	91	85
F9 Watson	90	85
F10 Ext. Rosenbrock	90	81
F11 Penalty II	92	80
F12 Powel badly scaled	91	82
F13 Gulf research and dev.	91	82
F14 Extended Powel singular	90	81
Μεση Αποτελεσματικότητα	91	85.2

Πίνακας 5.9: Ποσοστά Επιτυχίας PMA3 για 1000 δοκιμές.

Η καλύτερη απόδοση της μεθόδου PMA4, λαμβάνοντας υπόψη και το βαθμό δυσκολίας του προβλήματος, παρατηρήθηκε στην περίπτωση της 10-διάστατης συνάρτησης Griewank. Παρα πολύ καλά αποτελέσματα η μέθοδος εφερε και στην περίπτωση της συνάρτησης Watson όσο και της βηματικής συνάρτησης, μόνο που χρειαστηκε να διερεύνησει εξαιρετικά μεγάλο αριθμό λύσεων. Στον πίνακα 5.4 βλέπουμε πόσο επηρεάζει την ποιότητα των αποτελεσμάτων για τις 4 υλοποιησεις η χρήση διαφορετικών ειδών ανασυνδυασμού ενώ στο πίνακα 5.5 βλέπουμε την επίδραση που εχει η χρηση αυτών στους χρόνους.

Τα αποτελέσματα που είχαμε και για τις τρείς μεθόδους αλλά ιδιαίτερα για την PMA4 δείχνουν ότι η συζευξη όλων αυτών των τεχνικών μέσα σε ανταγωνιστικό πλαίσιο που

καθορίζεται σε ένα περιβάλλον παράλληλης αρχιτεκτονικής δίνουν μια αδρομερή διερεύνηση του χώρου εφικτών λύσεων και εγγυάται τη συγκλιση στο ολικό βέλτιστο μιας συνάρτησης αποφεύγοντας τον εγκλωβισμό σε τοπικά ακρότατα.

Η υβριδική μέθοδος PMA4 πραγματοποιεί αρχικά μια αδρή και στη συνέχεια μια πιο λεπτομερή διερεύνηση του εφικτού χώρου. Προφανώς, απαιτήθηκε η θεώρηση μεγαλύτερου πληθυσμού όσο αυξανόταν ο βαθμός δυσκολίας του προβλήματος. Έτσι, ενώ ήταν επαρκής μία και μόνο νησίδα για την εύρεση του ελαχίστου της διδιάστατης Rozenbrock και της Extended Rozenbrock, απαιτήθηκαν τέσσερις νησίδες για την επίλυση των ίδιων προβλήματων στις 10 διαστάσεις, ενώ οι τέσσερις νησίδες δεν ήταν επαρκείς για την επίλυση των προβλήματων Penalty II, Rastrigin, Powel badly scaled με ικανοποιητική αξιοπιστία. Καταλήξαμε στην χρήση έξι νησίδων μια λύση που ικανοποιούσε την πλειονοτητα των προς επίλυση προβλημάτων και παρατηρήσαμε ότι η ταχύτητα της μεθόδου παρουσίασε σημαντικές διαφορές, ανάλογα με το βαθμό δυσκολίας του εκάστοτε προβλήματος και ήταν προφανώς ανάλογη του πλήθους των επεξεργαστών αλλά και του αριθμού των νησίδων σε κάθε περίπτωση όπως φαίνεται και στον πίνακα 5.2. Έτσι, ο εντοπισμός του ολικού ακρότατου των πιο εύκολων συναρτήσεων ελέγχου, όπως της 10-διάστατης σφαιροειδούς, απαίτησε μικρό αριθμό δοκιμών με μικρό αριθμό επεξεργαστών , σε αντίθεση με πολυπλοκότερες συναρτήσεις, όπως για παράδειγμα η Rastrigin και η διδιάστατη Rozenbrock, για τις οποίες χρειάστηκαν μία ως δύο τάξεις μεγέθους περισσότερες δοκιμές και οι 18 επεξεργαστές της πειραματικής μας συστοιχίας.

Όσον αφορά τις συνάρτησεις Griewang και τη Watson, το υπερβολικά μεγάλο πλήθος δοκιμών είχε ως αίτιο την αδυναμία διαφυγής του αλγορίθμου από κορυφογραμμές. Μια

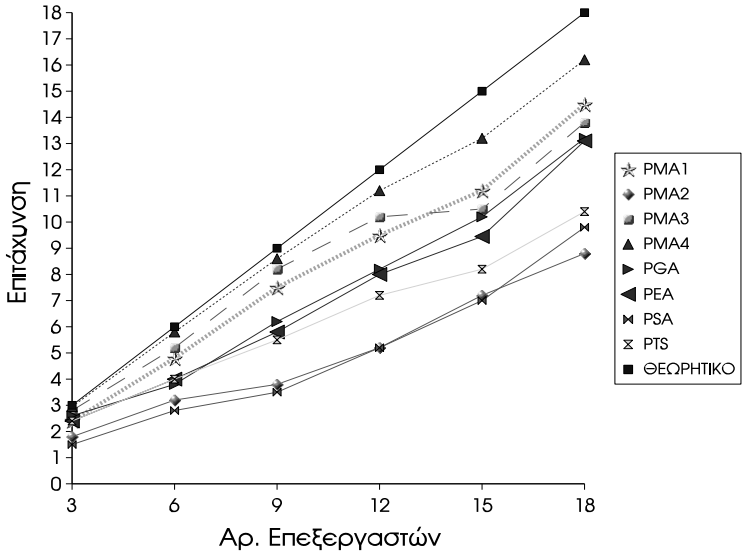
Συνάρτηση	Ποσοστά Επιτυχίας
F1 Sphere model	97
F2 Rosenbrock	98
F3 Step	96
F4 Quatric	97
F5 Foxholes	97
F6 Rastrigin	97
F7 Schwefel	98
F8 Griewang	97
F9 Watson	98
F10 Ext. Rosenbrock	98
F11 Penalty II	97
F12 Powel bandly scaled	97
F13 Gulf research and dev.	96
F14 Ext. Powel singular	98
Μεση Αποτελεσματικότητα	97

Πίνακας 5.10: Ποσοστά Επιτυχίας PMA4 για 1000 δοκιμές.

αντίστοιχη αδυναμία διαπιστώθηκε συγκρίνοντας τη συμπεριφορά του αλγορίθμου στις δύο συναρτήσεις Rozenbrock και Extended Rozenbrock. Στην περίπτωση της διδιάστατης Rozenbrock, ο αλγόριθμος κατόρθωσε με τη χρήση των εξι νησίδων που εξελίσσονται με διαφορετική μέθοδολογία να προχωρήσει κατά μήκος της παραβολοειδούς χαράδρας και να συγκλίνει κοντά στο σημείο ελαχίστου πολυ γρήγορα. Αυξάνοντας τη διάσταση του προβλήματος πετυχε μια πολυ καλή συγκλιση της τάξης του 75%.

5.5.2 Αποτελέσματα για το TSP

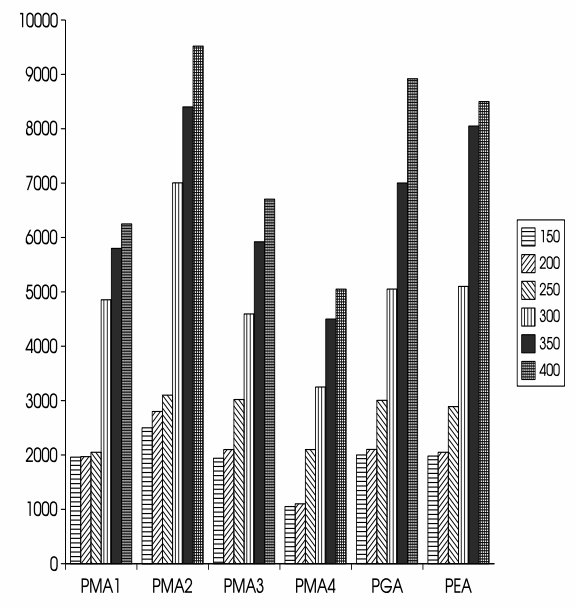
Για να μελετήσουμε την απόδοση των υλοποιήσεων μας και σε συνδυαστικά προβλήματα (combinatorial) χρησιμοποιήσαμε ένα από τα κλασσικά προβλήματα της περιοχής, το



Σχήμα 5.13: Συγκριτικά οι επιταχύνσεις για 8 μεθόδους για το TSP πρόβλημα

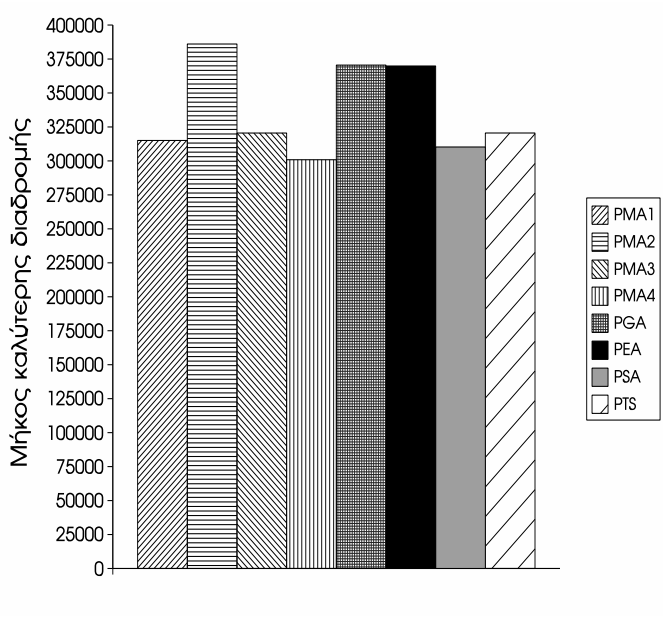
πρόβλημα του περιοδεύοντος πωλητή (Traveling Salesman Problem - TSP). Δεδομένων των εξόδων μεταφοράς μεταξύ των διαφόρων πόλεων, το TSP συνίσταται στην προσπάθεια του πλανόδιου πωλητή να επισκεφτεί όλες τις πόλεις της περιοχής του, μία μόνο φορά την καθεμιά, και να επιστρέψει στην πόλη από την οποία ξεκίνησε με το ελάχιστο δυνατό κόστος.

Σκοπός μας δεν είναι η επίλυση του συγκεκριμένου προβλήματος αλλά η επιβεβαίωση των αποτελεσμάτων και συμπερασμάτων που προκύπτουν για την απόδοση των υλοποιήσεων μας για τις 14 συναρτήσεις ελέγχου. Από τα διαθέσιμα προβλήματα επιλέχθηκε ένα πρόβλημα μεγέθους 9882 συνολικά τοποθεσιών στον Ελλαδικό χώρο. Η κατανομή



Σχήμα 5.14: Αριθμός γενεών που απαιτήθηκαν από 5 μεθόδους για το TSP πρόβλημα για διάφορα μεγέθη πληθυσμού

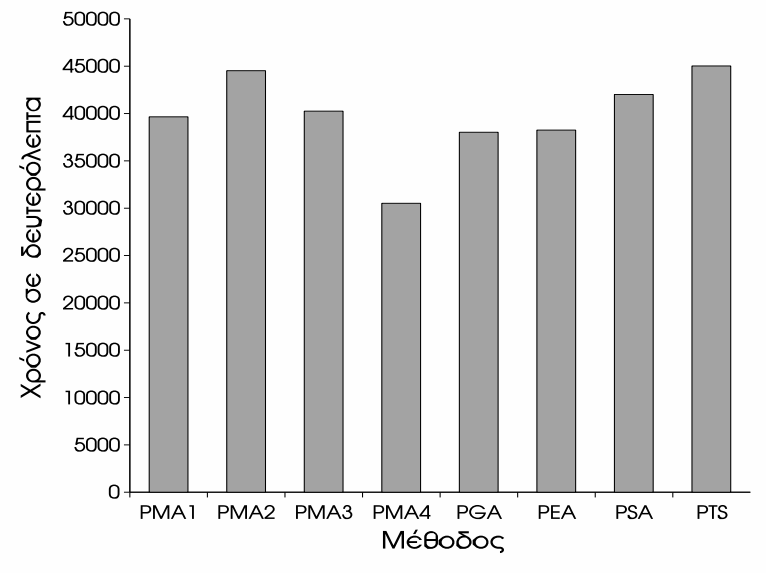
των 9882 τοποθεσιών στο χάρτη της Ελλάδας φαίνονται στο σχήμα 5.17. Τα στοιχεία παρέχονται στο gr9882.tsp (http://www.informatik.uni-heidelberg.de/groups/ comopt/software/TSPLIB95 / gr9882.tsp). Στο αρχείο αυτό αναγράφονται οι ακριβείς αποστάσεις μεταξύ όλων των τοποθεσιών. Το πρώτο βασικό ερώτημα που πρέπει να απαντηθεί είναι αυτό σχετικά με τον τρόπο αναπαράστασης του χρωμοσώματος που θα αναπαριστά τις πιθανές λύσεις. Μπορεί να είναι είτε ένας πίνακας ακεραίων είτε μια δυαδική συμβολοσειρά. Η δυαδική αναπαράσταση δεν ανταποκρίνεται στις απαιτήσεις του συγκεκριμένου προβλήματος του Περιοδεύοντος Πωλητή. Κι αυτό γιατί σε μια δυαδική αναπαράσταση για ένα TSP πόλεων, κάθε πόλη θα έπρεπε να αναπαρασταθεί ως μια δυαδική συμβολοσειρά δυαδικών ψηφίων. Αυτό σημαίνει ότι το χρωμόσωμα θα κατέληγε



Σχήμα 5.15: Καλύτερες τιμές για τα συνολικά μήκη των διαδρομών για το TSP πρόβλημα (Αποτελεσματα 8 μεθόδων)

να αποτελείται από δυαδικά ψηφία. Κατά τη διαδικασία της μετάλλαξης θα μπορούσε να εμφανιστεί πρόβλημα, όταν θα δημιουργείτο ακολουθία πόλεων, στην οποία μία τουλάχιστον πόλη θα εμφανιζόταν παραπάνω από μία φορά.

Επιπλέον, για ένα TSP με πόλεις (όπου για την αναπαράσταση κάθε πόλης θα χρειαζόντουσαν δυαδικά ψηφία), μερικές ακολουθίες των δυαδικών ψηφίων δεν θα αντιστοιχούν σε καμία πόλη. Παρόμοια προβλήματα μπορεί να εμφανιστούν και κατά την εφαρμογή των τελεστών διασταύρωσης. Απ΄ όλα αυτά συμπεραίνει κανείς, ότι εάν γινόταν χρήση των τελεστών μετάλλαξης και διασταύρωσης, όπως αυτοί ορίσθηκαν στα προηγούμενα παραδείγματα, θα ήταν απαραίτητη η ύπαρξη ενός διορθωτικού αλγορίθμου. Ένας τέτοιος αλγόριθμος θα επιδιόρθωνε κάθε χρωμόσωμα μεταφέροντάς το μέσα στο επιτρεπτό



Σχήμα 5.16: Μέσοι χρόνοι σε δευτερόλεπτα των υλοποιήσεων για το πρόβλημα TSP

σύνολο τιμών.

Φαίνεται λοιπόν από τα παραπάνω, ότι η αναπαράσταση με διάνυσμα ακεραίων αποτελεί καλύτερη επιλογή. Κι αυτό γιατί μ΄ αυτό τον τρόπο, αντί να εφαρμόζονται διορθωτικοί κανόνες μετά από τη χρήση των γενετικών τελεστών, μπορούν να ενσωματωθούν η γνώση και οι πληροφορίες που έχουμε για το συγκεκριμένο πρόβλημα από πριν στους τελεστές αυτούς. Έτσι, απομακρυνεται με έξυπνο τρόπο ο κίνδυνος για παραγωγή ατόμων εκτός των ορίων του διαστήματος τιμών. Στη συγκεκριμένη προσέγγιση χρησιμοποιείται αναπαράσταση με ακέραιους αριθμούς.

Τα καλύτερα αποτελέσματα απόδοσης προκύπτουν για τις μεθόδους PMA1, PMA3,

PMA4 ενώ λιγότερο καλά είναι τα αποτελέσματα για την PMA2. Για τη διαδικασία αρχικοποίησης χρησιμοποιούμε μια από τις μεθόδους (αποτρεπτική αναζήτηση, προσωμοιούμενη ανόπτηση, κατευθυνόμενη τοπική αναζήτηση) ανάλογα με την υλοποίηση (PMA1, PMA2, PMA3, PMA4). Η αποτίμηση των χρωμοσωμάτων γίνεται με άμεσο και ευθύ τρόπο: δεδομένων των εξόδων μεταφοράς μεταξύ των διαφόρων πόλεων, μπορούμε εύκολα να υπολογίσουμε το συνολικό κόστος ολόκληρης της διαδρομής.

Ο χώρος των λύσεων του προβλήματος έχει πολλές ασυνέχειες. Το πρόβλημα TSP χρησιμοποιηθηκε για να επιβεβαιωθεί αν τελικά η χρήση τεχνικών τοπικής αναζήτησης στις υλοποιησεις μας προσφέρουν καλύτερες δυνατότητες διερεύνησης του χώρου λύσεων έναντι των αντίστοιχων παράλληλων υλοποιήσεων άλλων τεχνικών όπως ενός απλού EA ή GA. Το πρόβλημα του περιοδεύοντος πωλητή μπορεί να ορισθεί ως εξής: Θεωρούμε n πόλεις με γνωστές αποστάσεις μεταξύ τους. Σκοπός είναι να βρεθεί η διαδρομή που καλύπτει όλες τις πόλεις και έχει την ελάχιστη συνολική απόσταση. Συνολικά έγιναν 380 πειράματα για το πρόβλημα του περιοδεύοντος πωλητή και τα αποτελέσματα είναι οι μέσοι όροι 204800 εκτελέσεων για κάθε πείραμα (4x2x8x8x8x5x5x2).

Στο σχήμα 5.14 βλέπουμε τον αριθμό γενεών που χρειαστηκε για 5 μεθόδους προκείμενου να βρουμε μια καλή λύση για το συγκεκριμένο πρόβλημα TSP ενώ το σχήμα 5.13 παρουσιάζει τις επιταχύνσεις για το πρόβλημα του περιοδεύοντος πωλήτη. Τα ιδια συμπεράσματα με τα αποτελέσματα των μεθοδολογιών μας στις 14 συναρτήσεις ελέγχου μπορεί να εξάγει κάποιος και για το πρόβλημα του περιοδεύοντος πωλητή. Σε όλα τα αποτελέσματα παρατηρούμε μια απότομη πτώση των καμπυλών επιτάχυνσης που οφείλεται στην ετερογένεια του περιβάλλοντος μας. Η απόδοση της μεθόδου PMA2 ήταν η χειρότερη

ενώ της PMA4 στην οποία τρέχουμε διαφορετικά είδη αλγορίθμων ήταν η καλύτερη αφού με αυτήν βρέθηκε το μικρότερο μήκος διαδρομής. Στο σχήμα 5.15 έχουμε τα συνολικά μήκη διαδρομών που επιτεύχθηκαν με κάθεμια από τις μεθόδους που εξετάσαμε. Η επιτάχυνση που πραγματοποιείται οφείλεται όχι μόνο στον παραλληλισμό αλλά και στον χρόνο που επιτυγχάνεται με την αύξηση των αριθμών των νησίδων. Αν και η συχνή επικοινωνία μεταξύ των πληθυσμών αυξάνει τις επικοινωνίες μεταξύ των επεξεργαστών επειδή ο χρόνος επεξεργασίας είναι συγκριτικά πολύ μεγάλος, η επίδραση των επικοινωνιών στην ταχύτητα των πειραμάτων είναι σχετικά μικρή. Στο σχήμα 5.16 βλέπουμε τους χρόνους των πειραμάτων για το πρόβλημα του περιοδεύοντος πωλητή. Παρατηρείται αύξηση της απόδοσης καθώς αυξάνεται το μέγεθος του πληθυσμού.

Για λόγους πληρότητας παραθέτουμε συγκριτικά αποτελέσματα (βλ. πίνακα 5.22) της μεθόδου μας για ακομα επτά προβλήματα TSP λιγότερο δυσκολα που πήραμε από την διεύθυνση http://www.iwr.uni-heidelberg.de/groups/comopt/software/TSPLIB95/index.html. Στον πίνακα αυτο εμφανιζονται οι μεσοι οροι ενος αριθμου εκτελεσεων καθε υλοποιηση. Στη στηλη Ἀβελτ' παρουσιαζονται οι βελτιστες λυσεις καθε προβληματος. Στο δευτερο τμημα του πινακα έχουμε τους χρονους εκτελεσης σε δευτερολεπτα.Από τα παραπάνω αποτελέσματα βλέπουμε ότι οι μεθοδολογίες μας εχουν πολύ καλή απόδοση και ποιότητα λύσης. Ιδιως σε μικρής και μέτριας δυσκολίας προβλήματα έχουμε βρεί λύσεις πολύ κοντά στη βέλτιστη τιμή και σε πολύ συντομο χρονικό διάστημα (βλ. πίνακα 5.14). Ο λόγος του χρόνου υπολογισμού προς το χρόνο επικοινωνίας εξαρτάται ως ένα βαθμό απο την ομοιογένεια των αλγορίθμων που εκτελούνται στις διάφορες νησίδες επεξεργαστών.

Χρησιμοποιώντας ετερογενείς αλγορίθμους στην PMA4 παρατηρούμε ότι ο παράλληλος

χρόνος εκτέλεσης των υλοποιήσεων μας βελτιώνεται σε σχέση με τον παράλληλο χρόνο εκτέλεσης με χρήση ομοιογενών αλγορίθμων. Ο χρόνος όμως επικοινωνίας είναι με μικρές αποκλίσεις περίπου ο ίδιος εφόσον διατηρούμε το μέγεθος πληθυσμού σταθερό.

Είναι προφανές ότι με την παράλληλη υλοποίηση των μεθοδολογιών του κεφαλαίου 4 μειώθηκε το κόστος υπολογισμου και πραγματοποιηθηκε ενα σχετικά μεγάλο πλήθος δοκιμών για τον εντοπισμό της ολικά βέλτιστης λύσης με ικανοποιητικη αξιοπιστία.

Τα αποτελέσματα του προβλήματος του περιοδεύοντος πωλητή συμφωνούν με τα αποτελέσματα των υπολοίπων προβλημάτων. Από τα πειραματικά αποτελέσματα παρατηρούμε ότι χρησιμοποιώντας διαφορετικές τιμές για το μέγεθος πληθυσμού καταλήγουμε σε διαφορετικές αποδόσεις. Η χειρότερη απόδοση προκύπτει για πολύ μικρές και μεγάλες τιμές του πληθυσμού. Αυτό συμβαίνει επειδή οι μικρές τιμές του πληθυσμού αυξάνουν την επιβάρυνση της διαεπεξεργαστικής επικοινωνίας, ενώ οι μεγάλες τιμές παρέχουν χαμηλή ισορροπία φορτίου.

5.6 Συμπεράσματα

Από τα αποτελέσματα φαίνεται ότι η PMA4 αντιμετώπισε με μεγαλη επιτυχία από 80% $\sim$ 100% τα περισσότερα από τα προβλήματα. Για τις υπόλοιπες τρείς μεθόδους παρατηρουμε ότι τα ποσοστά επιτυχίας ήταν υψηλότερα από αυτά των σειριακών υλοποιησεων όπως ήταν αναμενόμενο αλλα σε γενικές γραμμές ως προς τη σειριακή τους εκδοχή και σε σχέση πάντα με την αποτελεσματικότητα τους παρουσίασαν βελτίωση με αύξηση του

πλήθους των επεξεργαστών, χωρίς ωστόσο η βελτίωση αυτή να είναι ομοιόμορφη ποιοτικά.

Η PMA4 πλεονεκτεί και αυτό φαίνεται και στα δύσκολα προβλήματα. Επιτυγχάνουμε το υψηλότερα ποσοστα επιτυχίας τάξης $93 \sim 96\%$ στα περισσότερα από τα προβλήματα και σχετικά με μικρές απαιτήσεις. Και στις 4 περιπτώσεις όσο αυξάνουμε τις επαναλήψεις τόσο προσεγγίζουμε καλύτερα την συνάρτηση και άρα πετυχαίνουμε μεγαλύτερη ακρίβεια.

Η συντομότερη διαδρομή για το πρόβλημα του περιοδεύοντος πωλητή (σχήμα 5.18) έχει μήκος 310.876 και βρέθηκε με 3250 γενεές και μέγεθος πληθυσμού 300 από την μέθοδο PMA4 σε χρόνο 30520 δευτερόλεπτα ή αλλιώς σε 8,48 ώρες. Οι μεθοδολογίες μας με βαση και τα τεχνικά χαρακτηριστικά της πειραματικής μας συστοιχίας είχαν καλή απόδοση και καλή ποιότητα λύσης και στο πρόβλημα του περιοδεύοντος πωλητή TSP. Τα αποτελέσματα του προβλήματος του περιοδεύοντος πωλητή συμφωνούν με τα αποτελέσματα από τα υπόλοιπα 14 προβλήματα ελέγχου απόδοσης.

Οι μεγάλοι πληθυσμοί δημιουργούν προβλήματα καθώς κινούνται αργά προς τη βέλτιστη λύση και δημιουργούν πολλούς απογόνους σε κάθε γενιά με αποτέλεσμα να χρειάζονται πολλές επαναλήψεις. Συχνά με τη χρήση μεγάλου πληθυσμού ο αλγόριθμος παραβλέπει μία λύση με συνάρτηση κόστους άνω του μέσου όρου η οποια βρίσκεται σε μια στενή κορυφή του πεδίου αναζήτησης, καθώς έχει την τάση να οδηγείται σε πιο ευρείες κορυφές. Η απόδοση των μεθοδολογιών μας με μικρό μέγεθος πληθυσμού είναι καλυτερη από την απόδοση των άλλων μεγεθών πληθυσμού, αλλά συνήθως καθηλώνεται σε τοπικά σημεία. Μπορούμε να συμπεράνουμε ότι οι μικρές τιμές μεγέθους υποπληθυσμών είναι καλό να

χρησιμοποιούνται όταν ο στόχος είναι η ταχεία βελτιστοποίηση και όχι τόσο η εύρεση του ολικού ελάχιστου. Αντιθετα η χρήση μεγάλων τιμών υποπληθυσμών βοηθά στην ευρεση μιας πολύ καλής λύσης αλλά όχι και στην ταχεία βελτιστοποίηση.

Τα αποτελέσματα που είχαμε και για τις τρείς μεθόδους αλλά ιδιαίτερα για την PMA4 δείχνουν ότι η συζευξη όλων αυτών των τεχνικών μέσα σε ανταγωνιστικό πλαίσιο που καθορίζεται σε ένα περιβάλλον παράλληλης αρχιτεκτονικής δίνουν μια αδρομερή διερεύνηση του χώρου εφικτών λύσεων και εγγυάται τη συγκλιση στο ολικό βέλτιστο μιας συνάρτησης αποφεύγοντας τον εγκλωβισμό σε τοπικά ακρότατα.

Ερευνώντας ακόμα περισσότερο την PMA4, είναι δυνατόν να ελεγχθούν άλλες υλοποιήσεις του μοντέλου μέσω της επίδρασης διαφόρων άλλων στοιχείων στην απόδοση του όπως διαφορετικές κωδικοποιησεις ανά υποπληθυσμό, προσαρμογή της δομής της γειτονιάς στην αρχιτεκτονική του συστηματος, διαφορετικούς μηχανισμούς επιλογής ανά υποπληθυσμό. Έτσι προτείνεται για περαιτέρω διερεύνηση ο ανταγωνισμός ή η συνεργασία των πληθυσμών με διαφορετικές τεχνικές τοπικής αναζήτησης με απώτερο στόχο τον εντοπισμό με ακρίβεια και ταχύτητα του τοπικού ακροτάτου στην περιοχή έλξης του οποίου βρίσκεται το σημείο εκκίνησης τους χρησιμοποιώντας όμως εξελικτικές συμπεριφορές οι οποίες τους καθιστούν ικανούς στο να ξεφύγουν από αυτό. Ενδιαφέρον θα ήταν να δοκιμάσουμε ποια θα ήταν η επίδραση στην συμπεριφορά της μεθοδους PMA4 η χρήση σε καποια νησίδα επεξεργαστών αλγορίθμων βελτιστοποίησης με αποικίες ψηφιακών μυρμηγκιών στους οποίους θα γίνεται χρήση διαφορετικών τεχνικών τοπικής αναζήτησης τόσο κατά το στάδιο ομαλοποίησης όσο και κατά το στάδιο της επαναρχικοποίησης του μηχανισμού φερομόνης. Πληροφορίες για τις μεθόδους βελτιστοποίησης με αποικίες

ψηφιακών μυρμηγκιών τους μηχανισμούς τους και την παραλληλοποίηση αυτών μπορεί να δεί ο αναγνώστης στην πρόσφατη εργασία των Benkner et al [23].

Σχήμα 5.17: Η κατανομή των 9882 τοποθεσιών στον Ελλαδικό χώρο

Σχήμα 5.18: Η καλύτερη διαδρομή για την σύνδεση 9882 τοποθεσιών στον Ελλαδικό χώρο

Συνάρ.	Παρ. Υλοπ.	Ενός σημείου	Δυό Σημείων	Πολλ. Σημείων	Ευρετικός Ανασυνδ.
$F1$ (40520)	PMA1	49	57	67	89
	PMA2	28	44	61	76
	PMA3	45	58	68	91
	PMA4	43	53	61	98
$F2$ (25020)	PMA1	84	63	69	88
	PMA2	62	41	62	80
	PMA3	79	48	67	90
	PMA4	96	45	61	98
$F3$ (25601)	PMA1	62	55	65	90
	PMA2	65	35	62	76
	PMA3	74	49	60	91
	PMA4	96	46	63	96
$F4$ (34535)	PMA1	83	47	66	88
	PMA2	64	50	61	76
	PMA3	66	49	70	91
	PMA4	96	47	64	95
$F5$ (28005)	PMA1	74	42	69	90
	PMA2	63	51	61	76
	PMA3	85	48	62	91
	PMA4	97	50	69	95
$F6$ (42500)	PMA1	71	65	60	90
	PMA2	62	53	62	79
	PMA3	70	45	66	90
	PMA4	97	49	66	97
$F7$ (43050)	PMA1	62	63	62	88
	PMA2	65	41	60	79
	PMA3	66	47	64	90
	PMA4	95	49	64	96
$F8$ (48052)	PMA1	63	51	65	89
	PMA2	65	24	62	79
	PMA3	73	49	61	90
	PMA4	97	46	66	96
$F9$ (52140)	PMA1	64	68	65	90
	PMA2	65	45	62	79
	PMA3	72	45	67	91
	PMA4	96	48	64	96
$F10$ (51252)	PMA1	71	46	68	90
	PMA2	60	55	61	79
	PMA3	61	48	65	92
	PMA4	97	48	62	97
$F11$ (49250)	PMA1	61	49	64	88
	PMA2	63	47	61	77
	PMA3	71	48	65	91
	PMA4	96	48	61	97
$F12$ (52012)	PMA1	67	55	68	89
	PMA2	63	49	61	76
	PMA3	67	49	65	91
	PMA4	98	48	67	96
$F13$ (54201)	PMA1	65	51	68	88
	PMA2	63	23	61	76
	PMA3	86	47	64	92
	PMA4	98	46	65	97
$F14$ (48525)	PMA1	65	51	68	88
	PMA2	63	23	61	76
	PMA3	86	47	64	92
	PMA4	98	46	65	97

Πίνακας 5.11: Ποσοστά Επιτυχίας των 4 υλοποιήσεων για διαφορετικά είδη ανασυνδυασμού - Εντός παρενθέσεων αναγράφεται το μέσο πλήθος λύσεων που εξερευνήθηκαν

Συνάρ.	Παρ. Υλοπ.	Ενός σημείου	Δυό Σημείων	Πολλ. Σημείων	Ευρετικός Ανασυνδ.
$F1$	PMA1	4833	3989	4243	4363
	PMA2	4858	4076	4660	4704
	PMA3	4823	3925	3821	3946
	PMA4	4423	3278	3589	3677
$F2$	PMA1	6239	4024	5492	5667
	PMA2	4918	5774	5667	5821
	PMA3	5379	4119	5444	5598
	PMA4	5439	4485	4472	4535
$F3$	PMA1	7213	4217	6438	6497
	PMA2	4847	3904	7096	7294
	PMA3	4855	3938	6560	6605
	PMA4	6739	4169	5853	5905
$F4$	PMA1	6322	5382	6300	6455
	PMA2	5578	6634	6968	7039
	PMA3	7291	4669	6288	6361
	PMA4	5355	4466	5966	6178
$F5$	PMA1	5844	5048	5415	5602
	PMA2	6119	4944	5812	5946
	PMA3	6231	3873	5661	5703
	PMA4	5385	4179	5318	5464
$F6$	PMA1	9521	11306	11877	11986
	PMA2	13304	12623	12520	12623
	PMA3	9687	8973	11779	11831
	PMA4	8848	5419	11135	11219
$F7$	PMA1	6277	5826	7390	7474
	PMA2	8219	4123	7886	7973
	PMA3	6667	4459	7389	7436
	PMA4	6296	5520	6854	7014
$F8$	PMA1	5668	7183	7389	7555
	PMA2	4973	4804	8272	8420
	PMA3	7080	6686	7339	7431
	PMA4	7423	3814	6979	7149
$F9$	PMA1	5031	5196	6288	6356
	PMA2	6766	4092	6607	6784
	PMA3	6302	5896	6197	6256
	PMA4	6869	5294	6004	6047
$F10$	PMA1	6720	3969	6331	6462
	PMA2	7689	3896	6770	6942
	PMA3	7083	5382	6389	6455
	PMA4	7866	4872	8375	8524
$F11$	PMA1	9847	4111	8935	9106
	PMA2	8915	6793	8385	8456
	PMA3	7508	5636	8225	8386
	PMA4	7552	6931	8468	8593
$F12$	PMA1	5542	6583	9073	9175
	PMA2	7569	7286	8331	8525
	PMA3	8519	8034	8238	8345
	PMA4	8179	4724	8117	8205
$F13$	PMA1	5652	4705	7019	7147
	PMA2	7339	7971	8003	8112
	PMA3	5326	5280	7005	7196
	PMA4	7343	4400	6801	6968
$F14$	PMA1	6481	6773	6701	6873
	PMA2	7708	4089	7381	7427
	PMA3	6012	5063	6838	6920
	PMA4	7410	5200	6441	6497

Πίνακας 5.12: Οι καλύτεροι χρόνοι των 4 υλοποιήσεων για διαφορετικά είδη ανασυνδυασμού

Συνάρ.	Παρ. Υλοπ.	1 Νης. 3 επεξ.	2 Νης. 6 επεξ.	3 Νης. 9 επεξ.	4 Νης. 12 επεξ.	5 Νης. 15 επεξ.	6 Νης. 18 επεξ.
$f1$	PMA1	16894	8547	5711	4500	6104	2936
	PMA2	18256	9228	6165	4858	6590	3163
	PMA3	15229	7715	5156	4063	5509	2658
	PMA4	14151	7175	4797	3779	5124	2478
$f2$	PMA1	22109	11154	7450	5871	7966	3805
	PMA2	22722	11461	7654	6032	8185	3907
	PMA3	21833	11017	7358	5798	7868	3759
	PMA4	17582	8891	5941	4681	6349	3050
$f3$	PMA1	25424	12812	8555	6742	9150	4357
	PMA2	28611	14405	9617	7579	10288	4888
	PMA3	25859	13030	8700	6856	9305	4430
	PMA4	23058	11629	7766	6120	8305	3963
$f4$	PMA1	25259	12730	8500	6696	9091	4330
	PMA2	27593	13896	9278	7312	9625	4719
	PMA3	24882	12541	8374	6600	8957	4267
	PMA4	24150	12175	8130	6407	8695	4145
$f5$	PMA1	21847	11024	7362	5802	7873	3761
	PMA2	23223	11711	7821	6163	8364	3960
	PMA3	22252	11226	7497	5908	8017	3829
	PMA4	21297	10748	7179	5657	7676	3669
$f6$	PMA1	47369	23784	15870	12509	16987	8015
	PMA2	49614	25057	16718	13178	17897	8439
	PMA3	46750	23475	15663	12347	16766	7912
	PMA4	44304	22252	14848	11704	15893	7504
$f7$	PMA1	29332	14766	9857	7769	10546	5009
	PMA2	31327	15763	10522	8293	11258	5341
	PMA3	29178	14689	9806	7728	10491	4983
	PMA4	27494	13847	9245	7286	9889	4702
$f8$	PMA1	29654	14927	9665	7854	10661	5062
	PMA2	33113	16656	11118	8763	11896	5639
	PMA3	29158	14679	9796	7723	10484	4980
	PMA4	28030	14115	9423	7427	10081	4792
$f9$	PMA1	24860	12530	8367	6594	8949	4263
	PMA2	26572	13386	8937	7044	9560	4549
	PMA3	24464	12332	8235	6490	8807	4197
	PMA4	23627	11914	7956	6270	8508	4058
$f10$	PMA1	25284	12742	8508	6705	9100	4334
	PMA2	27205	13702	9148	7210	9786	4654
	PMA3	25259	12730	8500	6696	9091	4330
	PMA4	24943	12571	8394	6615	8978	4277
$f11$	PMA1	33528	16864	11256	8872	12044	5708
	PMA2	35856	18028	12032	9484	12876	6096
	PMA3	33258	16729	11166	8801	11948	5663
	PMA4	32978	16589	11073	8727	11848	5616
$f12$	PMA1	33803	17002	11348	8944	12143	5754
	PMA2	36131	18166	12124	9556	12974	6142
	PMA3	33534	16867	11258	8873	12046	5709
	PMA4	32813	16506	11018	8684	11789	5589
$f13$	PMA1	28025	14112	9422	7425	10079	4791
	PMA2	31883	16041	10708	8439	11457	5434
	PMA3	28231	14216	9490	7480	10153	4825
	PMA4	27430	13815	9223	7269	9867	4692
$f14$	PMA1	26929	13565	9056	7138	9688	4608
	PMA2	29142	14671	9794	7719	10478	4977
	PMA3	27117	13658	9119	7187	9754	4639
	PMA4	25424	12812	8555	6742	9150	4357

Πίνακας 5.13: Πειραματικοί χρόνοι εκτέλεσης (σε δευτερόλεπτα) για μέγεθος πληθυσμού 50 για τις 4 υλοποιήσεις

Συνάρ.	Παρ. Υλοπ.	1 Νης. 3 επεξ.	2 Νης. 6 επεξ.	3 Νης. 9 επεξ.	4 Νης. 12 επεξ.	5 Νης. 15 επεξ.	6 Νης. 18 επεξ.
$f1$	PMA1	16896	8550	5713	4501	6105	2937
	PMA2	18262	9231	6167	4860	6592	3164
	PMA3	15234	7717	5158	4064	5511	2659
	PMA4	14155	7178	4798	3780	5125	2479
$f2$	PMA1	22115	11158	7452	5872	7968	3806
	PMA2	22729	11465	7656	6034	8188	3908
	PMA3	21840	11020	7360	5800	7870	3760
	PMA4	17587	8894	5942	4682	6351	3051
$f3$	PMA1	25432	12816	8557	6744	9153	4359
	PMA2	28620	14410	9620	7582	10291	4890
	PMA3	25867	13034	8702	6858	9308	4431
	PMA4	23065	11632	7768	6122	8307	3964
$f4$	PMA1	25267	12733	8502	6701	9094	4331
	PMA2	27601	13901	9280	7314	9628	4720
	PMA3	24890	12545	8377	6602	8959	4268
	PMA4	24158	12179	8133	6409	8698	4146
$f5$	PMA1	21854	11027	7365	5804	7875	3762
	PMA2	23230	11715	7823	6165	8366	3962
	PMA3	22258	11229	7496	5910	8019	3830
	PMA4	21303	10752	7181	5659	7678	3671
$f6$	PMA1	47383	23792	15874	12513	16963	8017
	PMA2	49630	25065	16723	13182	17902	8442
	PMA3	46764	23482	15668	12351	16771	7914
	PMA4	44317	22259	14852	11707	15898	7506
$f7$	PMA1	29341	14770	9860	7771	10549	5010
	PMA2	31336	15768	10525	8296	11262	5343
	PMA3	29187	14693	9809	7731	10494	4984
	PMA4	27502	13851	9247	7288	9892	4704
$f8$	PMA1	29663	14931	9668	7856	10664	5064
	PMA2	33123	16661	11121	8765	11900	5640
	PMA3	29167	14684	9802	7726	10487	4981
	PMA4	28039	14119	9426	7429	10084	4793
$f9$	PMA1	24868	12534	8369	6596	8951	4265
	PMA2	26580	13390	8940	7046	9563	4550
	PMA3	24471	12336	8237	6492	8810	4196
	PMA4	23635	11917	7958	6272	8511	4059
$f10$	PMA1	25292	12746	8511	6707	9103	4335
	PMA2	27213	13706	9151	7212	9789	4655
	PMA3	25267	12733	8502	6701	9094	4331
	PMA4	24950	12575	8397	6617	8981	4278
$f11$	PMA1	33538	16869	11259	8874	12048	5710
	PMA2	35867	18033	12036	9487	12880	6098
	PMA3	33269	16734	11170	8804	11952	5665
	PMA4	32988	16594	11076	8730	11851	5618
$f12$	PMA1	33814	17007	11351	8947	12146	5756
	PMA2	36142	18171	12127	9559	12978	6144
	PMA3	33544	16872	11261	8876	12050	5711
	PMA4	32823	16511	11021	8686	11792	5590
$f13$	PMA1	28033	14117	9424	7428	10082	4792
	PMA2	31892	16046	10711	8442	11460	5435
	PMA3	28240	14220	9493	7482	10156	4827
	PMA4	27439	13819	9226	7271	9870	4693
$f14$	PMA1	26938	13569	9059	7140	9691	4610
	PMA2	29151	14675	9797	7721	10481	4978
	PMA3	27125	13662	9122	7189	9757	4641
	PMA4	25432	12816	8557	6744	9153	4359

Πίνακας 5.14: Πειραματικοί χρόνοι εκτέλεσης (σε δευτερόλεπτα) για μέγεθος πληθυσμού 100 για τις 4 υλοποιήσεις

Συνάρ.	Παρ. Υλοπ.	1 Νης. 3 επεξ.	2 Νης. 6 επεξ.	3 Νης. 9 επεξ.	4 Νης. 12 επεξ.	5 Νης. 15 επεξ.	6 Νης. 18 επεξ.
$f1$	PMA1	11193	5696	3811	3002	4067	1985
	PMA2	12100	6150	4113	3240	4392	2137
	PMA3	10083	5142	3441	2710	3671	1801
	PMA4	9364	4782	3201	2521	3414	1681
$f2$	PMA1	14668	7434	4969	3915	5309	2565
	PMA2	15077	7638	5106	4022	5455	2633
	PMA3	14484	7342	4908	3867	5243	2534
	PMA4	11651	5926	3964	3122	4231	2062
$f3$	PMA1	16878	8539	5706	4496	6098	2933
	PMA2	19001	9601	6414	5054	6856	3287
	PMA3	17167	8684	5802	4572	6201	2981
	PMA4	15301	7750	5180	4081	5534	2670
$f4$	PMA1	16768	8484	5669	4467	6058	2915
	PMA2	18323	9261	6188	4876	6614	3174
	PMA3	16516	8358	5585	4401	5969	2873
	PMA4	16029	8114	5423	4273	5794	2791
$f5$	PMA1	14494	7347	4911	3869	5246	2536
	PMA2	15411	7805	5217	4110	5574	2688
	PMA3	14763	7482	5001	3940	5343	2581
	PMA4	14127	7163	4789	3773	5115	2474
$f6$	PMA1	31503	15851	10581	8340	11321	5370
	PMA2	33196	16700	11146	8785	11927	5653
	PMA3	31090	15645	10443	8231	11174	5302
	PMA4	29460	14830	9600	7803	10591	5030
$f7$	PMA1	19482	9841	6574	5180	7028	3367
	PMA2	20811	10506	7017	5530	7503	3589
	PMA3	19379	9790	6540	5153	6961	3350
	PMA4	18257	9228	6166	4858	6590	3163
$f8$	PMA1	19696	9648	6645	5237	7104	3403
	PMA2	22002	11101	7414	5842	7928	3787
	PMA3	19366	9783	6535	5150	6987	3348
	PMA4	18614	9407	6285	4952	6718	3222
$f9$	PMA1	16502	8351	5581	4397	5963	2870
	PMA2	17642	8921	5961	4697	6371	3060
	PMA3	16238	8219	5493	4328	5869	2826
	PMA4	15680	7940	5307	4181	5670	2733
$f10$	PMA1	16784	8492	5675	4471	6064	2917
	PMA2	18064	9132	6101	4808	6522	3131
	PMA3	16768	8484	5669	4467	6058	2915
	PMA4	16557	8378	5596	4411	5983	2879
$f11$	PMA1	22279	11239	7506	5915	8027	3833
	PMA2	23830	12015	8023	6323	8581	4092
	PMA3	22096	11149	7446	5868	7962	3803
	PMA4	21912	11056	7384	5819	7896	3772
$f12$	PMA1	22462	11331	7567	5963	8092	3864
	PMA2	24013	12107	8084	6371	8646	4122
	PMA3	22282	11241	7507	5916	8028	3834
	PMA4	21802	11001	7347	5790	7856	3754
$f13$	PMA1	18611	9405	6284	4951	6717	3222
	PMA2	21182	10691	7141	5627	7635	3650
	PMA3	18748	9474	6329	4987	6766	3245
	PMA4	18215	9207	6152	4847	6575	3156
$f14$	PMA1	17881	9040	6040	4759	6456	3100
	PMA2	19355	9778	6532	5147	6983	3346
	PMA3	18006	9103	6082	4792	6501	3121
	PMA4	16878	8539	5706	4496	6098	2933

Πίνακας 5.15: Πειραματικοί χρόνοι εκτέλεσης (σε δευτερόλεπτα) για μέγεθος πληθυσμού 150 για τις 4 υλοποιήσεις

Συνάρ.	Παρ. Υλοπ.	1 Νης. 3 επεξ.	2 Νης. 6 επεξ.	3 Νης. 9 επεξ.	4 Νης. 12 επεξ.	5 Νης. 15 επεξ.	6 Νης. 18 επεξ.
$f1$	PMA1	11196	5698	3812	3003	4069	1986
	PMA2	12104	6152	4115	3241	4393	2137
	PMA3	10086	5143	3442	2711	3672	1801
	PMA4	9367	4784	3202	2522	3415	1681
$f2$	PMA1	14672	7436	4971	3916	5310	2565
	PMA2	15081	7641	5107	4024	5456	2634
	PMA3	14489	7344	4910	3868	5245	2535
	PMA4	11655	5927	3965	3123	4232	2062
$f3$	PMA1	16883	8541	5708	4497	6100	2934
	PMA2	19007	9604	6416	5055	6858	3288
	PMA3	17173	8686	5804	4573	6203	2982
	PMA4	15305	7753	5182	4083	5536	2671
$f4$	PMA1	16773	8486	5671	4468	6060	2915
	PMA2	18328	9264	6189	4877	6616	3175
	PMA3	16521	8361	5587	4402	5971	2874
	PMA4	16034	8117	5425	4274	5796	2792
$f5$	PMA1	14498	7349	4913	3870	5248	2536
	PMA2	15415	7808	5218	4111	5575	2689
	PMA3	14768	7484	5003	3941	5344	2581
	PMA4	14131	7166	4790	3774	5117	2475
$f6$	PMA1	31513	15856	10584	8342	11325	5372
	PMA2	33210	16705	11150	8788	11931	5655
	PMA3	31100	15650	10447	8234	11177	5303
	PMA4	29469	14835	9603	7805	10595	5032
$f7$	PMA1	19488	9844	6576	5182	7030	3368
	PMA2	20818	10509	7019	5531	7505	3590
	PMA3	19385	9793	6542	5155	6963	3351
	PMA4	18262	9231	6167	4860	6592	3164
$f8$	PMA1	19702	9651	6647	5238	7107	3404
	PMA2	22008	11104	7416	5844	7930	3788
	PMA3	19372	9786	6537	5151	6989	3349
	PMA4	18620	9410	6287	4954	6720	3223
$f9$	PMA1	16507	8353	5582	4398	5965	2871
	PMA2	17648	8924	5963	4698	6373	3061
	PMA3	16243	8221	5494	4329	5871	2827
	PMA4	15685	7942	5308	4182	5672	2734
$f10$	PMA1	16789	8495	5676	4473	6066	2918
	PMA2	18070	9135	6103	4809	6523	3132
	PMA3	16773	8486	5671	4468	6060	2915
	PMA4	16562	8381	5601	4413	5985	2880
$f11$	PMA1	22285	11243	7508	5917	8029	3834
	PMA2	23837	12019	8026	6325	8583	4093
	PMA3	22106	11153	7449	5870	7965	3804
	PMA4	21918	11059	7386	5821	7898	3773
$f12$	PMA1	22469	11334	7570	5965	8095	3865
	PMA2	24021	12110	8087	6373	8649	4123
	PMA3	22289	11245	7510	5918	8030	3835
	PMA4	21808	11004	7349	5792	7859	3755
$f13$	PMA1	18616	9408	6285	4953	6719	3223
	PMA2	21188	10694	7143	5629	7637	3651
	PMA3	18754	9477	6331	4989	6768	3246
	PMA4	18220	9210	6153	4849	6577	3157
$f14$	PMA1	17886	9043	6042	4761	6458	3101
	PMA2	19361	9781	6534	5148	6985	3347
	PMA3	18011	9106	6084	4794	6503	3122
	PMA4	16883	8541	5708	4497	6100	2934

Πίνακας 5.16: Πειραματικοί χρόνοι εκτέλεσης (σε δευτερόλεπτα) για μέγεθος πληθυσμού 200 για τις 4 υλοποιήσεις

Συνάρ.	Παρ. Υλοπ.	1 Νης. 3 επεξ.	2 Νης. 6 επεξ.	3 Νης. 9 επεξ.	4 Νης. 12 επεξ.	5 Νης. 15 επεξ.	6 Νης. 18 επεξ.
$f1$	PMA1	6005	3103	2082	1638	2215	1121
	PMA2	6500	3350	2247	1768	2391	1203
	PMA3	5400	2800	1880	1479	1968	1020
	PMA4	5008	2604	1749	1376	1858	954
$f2$	PMA1	7900	4050	2713	2136	2891	1436
	PMA2	8123	4161	2787	2194	2971	1473
	PMA3	7800	4000	2680	2109	2855	1420
	PMA4	6255	3227	2165	1703	2303	1162
$f3$	PMA1	9105	4652	3115	2452	3321	1637
	PMA2	10263	5231	3501	2757	3735	1830
	PMA3	9263	4731	3167	2494	3378	1663
	PMA4	8245	4222	2828	2226	3014	1494
$f4$	PMA1	9045	4622	3095	2437	3300	1627
	PMA2	9893	5046	3377	2660	3603	1768
	PMA3	8908	4554	3049	2401	3251	1604
	PMA4	8642	4421	2960	2331	3156	1560
$f5$	PMA1	7805	4002	2681	2111	2857	1420
	PMA2	8305	4252	2848	2242	3036	1504
	PMA3	7952	4076	2730	2149	2910	1445
	PMA4	7605	3902	2615	2058	2786	1387
$f6$	PMA1	17080	8640	5773	4548	6170	2966
	PMA2	18005	9102	6081	4792	6500	3120
	PMA3	16855	8527	5698	4489	6089	2929
	PMA4	15966	8083	5402	4256	5772	2781
$f7$	PMA1	10525	5362	3588	2826	3828	1874
	PMA2	11250	5725	3830	3017	4088	1965
	PMA3	10469	5335	3570	2811	3809	1865
	PMA4	9857	5029	3366	2651	3590	1763
$f8$	PMA1	10642	5421	3627	2857	3871	1894
	PMA2	11896	6050	4046	3187	4320	2103
	PMA3	10462	5331	3567	2809	3806	1863
	PMA4	10052	5126	3430	2701	3660	1795
$f9$	PMA1	8900	4550	3046	2396	3248	1603
	PMA2	9522	4861	3254	2562	3470	1707
	PMA3	8756	4478	2968	2361	3197	1579
	PMA4	8452	4326	2897	2281	3088	1528
$f10$	PMA1	9054	4627	3098	2439	3303	1629
	PMA2	9752	4976	3330	2623	3552	1745
	PMA3	9045	4622	3095	2437	3300	1627
	PMA4	8930	4565	3056	2406	3259	1608
$f11$	PMA1	12050	6125	4096	3226	4373	2128
	PMA2	12896	6548	4378	3449	4675	2269
	PMA3	11952	6076	4064	3201	4338	2112
	PMA4	11850	6025	4030	3174	4302	2095
$f12$	PMA1	12150	6175	4130	3253	4409	2145
	PMA2	12966	6598	4412	3475	4711	2286
	PMA3	12052	6126	4097	3227	4374	2128
	PMA4	11790	5965	4010	3158	4280	2085
$f13$	PMA1	10050	5125	3430	2701	3659	1795
	PMA2	11452	5826	3897	3069	4160	2028
	PMA3	10125	5162	3455	2721	3686	1807
	PMA4	9834	5017	3358	2644	3582	1759
$f14$	PMA1	9652	4926	3297	2596	3517	1728
	PMA2	10456	5328	3565	2808	3804	1862
	PMA3	9720	4960	3320	2614	3541	1740
	PMA4	9105	4652	3115	2452	3321	1637

Πίνακας 5.17: Πειραματικοί χρόνοι εκτέλεσης (σε δευτερόλεπτα) για μέγεθος πληθυσμού 250 για τις 4 υλοποιήσεις

Συνάρ.	Παρ. Υλοπ.	1 Νης. 3 επεξ.	2 Νης. 6 επεξ.	3 Νης. 9 επεξ.	4 Νης. 12 επεξ.	5 Νης. 15 επεξ.	6 Νης. 18 επεξ.
$f1$	PMA1	10180	6140	4140	2749	3670	2500
	PMA2	12040	7130	4340	2609	3879	2606
	PMA3	10080	5140	3440	2709	3670	1800
	PMA4	9361	4781	3200	2520	3413	1680
$f2$	ΠΜΑ'1	14663	7432	4968	3914	5307	2564
	PMA2	15072	7636	5104	4021	5453	2632
	PMA3	14480	7340	4907	3866	5241	2533
	PMA4	11648	5924	3963	3121	4230	2061
$f3$	PMA1	16873	8536	5704	4494	6096	2932
	PMA2	18966	9598	6412	5052	6854	3286
	PMA3	17162	8681	5801	4571	6196	2980
	PMA4	15296	7748	5179	4080	5533	2669
$f4$	PMA1	16763	8481	5668	4466	6057	2914
	PMA2	18317	9259	6186	4874	6612	3173
	PMA3	16511	8356	5584	4396	5967	2872
	PMA4	16024	8112	5421	4271	5793	2791
$f5$	PMA1	14489	7345	4910	3868	5245	2535
	PMA2	15406	7803	5215	4109	5572	2688
	PMA3	14759	7479	5000	3939	5341	2580
	PMA4	14123	7161	4788	3772	5114	2474
$f6$	PMA1	31493	15847	10578	8337	11318	5369
	PMA2	33189	16695	11143	8783	11923	5652
	PMA3	31081	15640	10440	8229	11170	5300
	PMA4	29451	14826	9897	7800	10588	5029
$f7$	PMA1	19476	9838	6572	5179	7026	3366
	PMA2	20805	10503	7015	5528	7500	3588
	PMA3	19373	9787	6538	5152	6989	3349
	PMA4	18251	9226	6164	4857	6588	3162
$f8$	PMA1	19690	9645	6643	5235	7102	3402
	PMA2	21965	11097	7412	5841	7925	3786
	PMA3	19360	9780	6533	5148	6984	3347
	PMA4	18609	9404	6283	4951	6716	3221
$f9$	PMA1	16497	8348	5579	4396	5962	2869
	PMA2	17637	8919	5959	4695	6369	3060
	PMA3	16233	8216	5491	4326	5867	2825
	PMA4	15675	7938	5305	4180	5668	2733
$f10$	PMA1	16779	8490	5673	4470	6063	2917
	PMA2	18059	9129	6100	4806	6520	3130
	PMA3	16763	8481	5668	4466	6057	2914
	PMA4	16552	8376	5597	4410	5981	2879
$f11$	PMA1	22272	11236	7504	5913	8024	3832
	PMA2	23823	12011	8021	6321	8578	4090
	PMA3	22092	11146	7444	5866	7960	3802
	PMA4	21905	11053	7382	5817	7893	3771
$f12$	PMA1	22455	11328	7565	5962	8090	3863
	PMA2	24006	12103	8082	6369	8644	4121
	PMA3	22275	11238	7505	5914	8025	3833
	PMA4	21795	10968	7345	5788	7854	3753
$f13$	PMA1	18605	9403	6282	4950	6715	3221
	PMA2	21175	10688	7138	5625	7633	3649
	PMA3	18743	9471	6328	4986	6764	3244
	PMA4	18209	9205	6150	4846	6573	3155
$f14$	PMA1	17875	9038	6038	4758	6454	3096
	PMA2	19349	9775	6530	5145	6980	3345
	PMA3	18000	9100	6080	4791	6496	3120
	PMA4	16873	8536	5704	4494	6096	2932

Πίνακας 5.18: Πειραματικοί χρόνοι εκτέλεσης (σε δευτερόλεπτα) για μέγεθος πληθυσμού 300 για τις 4 υλοποιήσεις

Συνάρ.	Παρ. Υλοπ.	1 Νης. 3 επεξ.	2 Νης. 6 επεξ.	3 Νης. 9 επεξ.	4 Νης. 12 επεξ.	5 Νης. 15 επεξ.	6 Νης. 18 επεξ.
$f1$	PMA1	25449	12824	8563	6748	9159	4361
	PMA2	27493	13846	9244	7286	9889	4702
	PMA3	22951	11576	7730	6092	8267	3945
	PMA4	21333	10766	7191	5667	7689	3675
$f2$	PMA1	33273	16736	11171	8805	11953	5665
	PMA2	34194	17197	11478	9047	12282	5819
	PMA3	32860	16530	11033	8696	11806	5597
	PMA4	26481	13341	8907	7020	9528	4534
$f3$	PMA1	38248	19224	12829	10112	13730	6495
	PMA2	43029	21615	14423	11369	15438	7292
	PMA3	38901	19550	13047	10284	13963	6603
	PMA4	34697	17449	11646	9179	12462	5903
$f4$	PMA1	38000	19100	12747	10047	13642	6453
	PMA2	41502	20851	13914	10967	14892	7037
	PMA3	37435	18817	12558	9896	13440	6359
	PMA4	36337	18268	12192	9610	13047	6176
$f5$	PMA1	32881	16540	11040	8702	11813	5600
	PMA2	34945	17573	11728	9244	12550	5944
	PMA3	33488	16844	11243	8861	12030	5701
	PMA4	32055	16127	10765	8485	11518	5462
$f6$	PMA1	71175	35688	23805	18766	25490	11983
	PMA2	74964	37597	25078	19770	26854	12619
	PMA3	70246	35223	23495	18522	25158	11828
	PMA4	66576	33388	22272	17557	23847	11216
$f7$	PMA1	44111	22156	14784	11653	15824	7472
	PMA2	47104	23652	15781	12440	16893	7971
	PMA3	43880	22040	14707	11592	15741	7433
	PMA4	41353	20777	13864	10928	14839	7012
$f8$	PMA1	44594	22397	14945	11780	15966	7552
	PMA2	49784	24962	16675	13144	17850	8417
	PMA3	43851	22025	14697	11585	15731	7428
	PMA4	42158	21179	14133	11140	15126	7146
$f9$	PMA1	37402	18801	12547	9890	13428	6354
	PMA2	39670	20085	13403	10565	14345	6782
	PMA3	36807	18504	12349	9734	13215	6255
	PMA4	35552	17876	11931	9404	12767	6045
$f10$	PMA1	38038	19119	12759	10057	13655	6460
	PMA2	40919	20560	13720	10814	14684	6940
	PMA3	38000	19100	12747	10047	13642	6453
	PMA4	37526	18863	12589	9622	13472	6374
$f11$	PMA1	50407	25304	16882	13308	18073	8521
	PMA2	53900	27050	18047	14226	19320	9103
	PMA3	50003	25101	16748	13202	17928	8454
	PMA4	49582	24891	16607	13091	17778	8384
$f12$	PMA1	50820	25510	17020	13417	18220	8590
	PMA2	54313	27257	18184	14335	19468	9172
	PMA3	50416	25308	16885	13310	18076	8523
	PMA4	49334	24767	16525	13026	17689	8342
$f13$	PMA1	42150	21175	14130	11138	15124	7145
	PMA2	47938	24069	16059	12659	17191	8110
	PMA3	42460	21330	14233	11219	15234	7197
	PMA4	41258	20729	13833	10903	14805	6966
$f14$	PMA1	40507	20353	13582	10706	14537	6871
	PMA2	43826	22013	14689	11578	15722	7424
	PMA3	40787	20494	13676	10780	14637	6918
	PMA4	38248	19224	12829	10112	13730	6495

Πίνακας 5.19: Πειραματικοί χρόνοι εκτέλεσης (σε δευτερόλεπτα) για μέγεθος πληθυσμού 350 για τις 4 υλοποιήσεις

Συνάρ.	Παρ. Υλοπ.	1 Νης. 3 επεξ.	2 Νης. 6 επεξ.	3 Νης. 9 επεξ.	4 Νης. 12 επεξ.	5 Νης. 15 επεξ.	6 Νης. 18 επεξ.
$f1$	PMA1	25457	12828	8566	6751	9162	4363
	PMA2	27501	13851	9247	7288	9892	4704
	PMA3	22958	11579	7733	6094	8269	3946
	PMA4	21339	10770	7193	5668	7691	3677
$f2$	PMA1	33283	16742	11174	8807	11957	5667
	PMA2	34204	17202	11481	9050	12286	5821
	PMA3	32870	16535	11037	8696	11809	5598
	PMA4	26489	13345	8910	7022	9530	4535
$f3$	PMA1	38260	19230	12833	10115	13734	6497
	PMA2	43042	21621	14427	11372	15442	7294
	PMA3	38912	19556	13051	10287	13967	6605
	PMA4	34708	17454	11649	9182	12466	5905
$f4$	PMA1	38012	19106	12751	10050	13646	6455
	PMA2	41514	20857	13918	10971	14897	7039
	PMA3	37446	18823	12562	9602	13444	6361
	PMA4	36348	18274	12196	9613	13051	6178
$f5$	PMA1	32891	16545	11044	8704	11817	5602
	PMA2	34956	17578	11732	9247	12554	5946
	PMA3	33498	16849	11246	8864	12034	5703
	PMA4	32065	16132	10768	8487	11522	5464
$f6$	PMA1	71197	35698	23812	18772	25497	11986
	PMA2	75017	37609	25086	19776	26862	12623
	PMA3	70268	35234	23503	18528	25166	11831
	PMA4	66596	33398	22279	17563	23854	11219
$f7$	PMA1	44125	22162	14788	11657	15829	7474
	PMA2	47119	23659	15786	12444	16898	7973
	PMA3	43893	22047	14711	11596	15746	7436
	PMA4	41366	20783	13869	10932	14843	7014
$f8$	PMA1	44608	22404	14949	11784	16001	7555
	PMA2	49796	25000	16680	13148	17855	8420
	PMA3	43864	22032	14701	11588	15736	7431
	PMA4	42171	21186	14137	11143	15131	7149
$f9$	PMA1	37413	18807	12551	9893	13432	6356
	PMA2	39682	20091	13407	10568	14349	6784
	PMA3	36818	18509	12353	9737	13219	6256
	PMA4	35563	17881	11934	9407	12771	6047
$f10$	PMA1	38049	19125	12763	10060	13659	6462
	PMA2	40932	20566	13724	10818	14689	6942
	PMA3	38012	19106	12751	10050	13646	6455
	PMA4	37537	18869	12592	9625	13476	6376
$f11$	PMA1	50423	25311	16888	13312	18078	8524
	PMA2	53917	27058	18052	14230	19326	9106
	PMA3	50018	25109	16753	13206	17934	8456
	PMA4	49597	24898	16612	13095	17783	8386
$f12$	PMA1	50836	25518	17025	13421	18226	8593
	PMA2	54330	27265	18190	14339	19474	9175
	PMA3	50431	25316	16890	13314	18081	8525
	PMA4	49349	24775	16530	13030	17695	8345
$f13$	PMA1	42163	21181	14134	11141	15128	7147
	PMA2	47953	24077	16064	12663	17196	8112
	PMA3	42473	21336	14238	11223	15239	7196
	PMA4	41271	20735	13837	10907	14810	6968
$f14$	PMA1	40519	20360	13586	10709	14541	6873
	PMA2	43840	22020	14693	11582	15727	7427
	PMA3	40800	20500	13680	10783	14641	6920
	PMA4	38260	19230	12833	10115	13734	6497

Πίνακας 5.20: Πειραματικοί χρόνοι εκτέλεσης (σε δευτερόλεπτα) για μέγεθος πληθυσμού 400 για τις 4 υλοποιήσεις

Συνάρ.	Παρ. Υλοπ.	1 Νης. 3 επεξ.	2 Νης. 6 επεξ.	3 Νης. 9 επεξ.	4 Νης. 12 επεξ.	5 Νης. 15 επεξ.	6 Νης. 18 επεξ.
$F1$	PMA1	62	65	75	75	84	89
	PMA2	62	62	68	70	74	77
	PMA3	74	64	76	83	83	92
	PMA4	95	95	97	96	96	97
$F2$	PMA1	82	70	74	76	81	91
	PMA2	63	63	67	71	74	78
	PMA3	72	65	74	79	84	90
	PMA4	97	98	98	97	95	98
$F3$	PMA1	64	76	73	80	80	88
	PMA2	63	63	66	70	72	75
	PMA3	68	71	82	80	82	91
	PMA4	96	97	97	96	96	96
$F4$	PMA1	74	64	73	85	85	88
	PMA2	62	61	69	69	73	80
	PMA3	86	64	83	79	83	91
	PMA4	96	96	98	96	98	97
$F5$	PMA1	68	84	72	85	85	90
	PMA2	65	65	67	70	74	80
	PMA3	66	68	86	79	82	92
	PMA4	98	95	97	96	96	97
$F6$	PMA1	64	78	72	81	80	88
	PMA2	62	63	68	72	74	80
	PMA3	78	69	75	85	85	91
	PMA4	97	97	96	98	98	97
$F7$	PMA1	68	61	74	84	85	90
	PMA2	62	62	66	70	75	77
	PMA3	85	67	80	85	85	92
	PMA4	96	97	97	95	97	98
$F8$	PMA1	71	61	70	79	84	91
	PMA2	64	62	68	70	74	80
	PMA3	84	82	74	81	83	91
	PMA4	98	96	98	97	97	97
$F9$	PMA1	75	81	72	78	84	90
	PMA2	65	65	66	72	73	79
	PMA3	62	77	82	80	83	90
	PMA4	96	96	96	96	95	98
$F10$	PMA1	72	75	73	80	85	89
	PMA2	62	64	68	70	72	77
	PMA3	84	87	87	85	83	91
	PMA4	98	96	96	95	96	98
$F11$	PMA1	62	68	70	82	81	89
	PMA2	61	62	68	70	73	79
	PMA3	86	82	85	85	82	90
	PMA4	98	97	97	98	95	98
$F12$	PMA1	72	75	73	79	83	90
	PMA2	64	65	67	69	73	78
	PMA3	77	81	77	82	84	90
	PMA4	98	97	96	97	97	97
$F13$	PMA1	81	82	75	81	81	90
	PMA2	64	60	66	71	75	79
	PMA3	73	78	79	80	83	92
	PMA4	96	97	97	96	96	96
$F14$	PMA1	65	78	75	82	83	91
	PMA2	61	65	68	70	73	77
	PMA3	73	65	83	79	85	91
	PMA4	97	98	98	97	98	98

Πίνακας 5.21: Ποσοστά Επιτυχίας των 4 υλοποιήσεων για διαφορετικό αριθμό επεξεργαστών

				Λύσεις				
Πρόβλημα	Βέλτιστη Λύση	PMA1	PMA2	PMA3	PMA4	PTS	PSA	PEA
ftv170	2755	2790	2810	2797	2770	2825	2820	2800
kro124p	36230	36240	36252	36242	36238	36280	36275	36240
ry48p	14422	14430	14442	14428	14426	14452	14448	14450
ft70	38673	38680	38694	38678	38675	38680	38690	38685
d198	15780	15792	15802	15783	15795	15805	15810	15795
eil51	426	430	442	435	429	450	440	435
kroA100	21282	21300	21350	21295	21289	24420	21765	21380
				Χρόνοι				
Πρόβλημα		PMA1	PMA2	PMA3	PMA4	PTS	PSA	PEA
ftv170		802	850	812	805	880	885	805
kro124p		780	820	750	760	850	860	802
ry48p		380	385	382	383	500	570	385
ft70		602	680	608	615	700	780	620
d198		950	980	908	960	1020	1084	975
eil51		420	460	418	425	520	530	435
kroA100		680	790	674	645	802	830	720

Πίνακας 5.22: Σύγκριση αλγορίθμων σε 7 προβλήματα TSP ΥΓ. Στήλη 2 Αναφερ. στην τρέχουσα βέλτιστη λύση

Κεφάλαιο 6

Χρονοπρογραμματισμός παραγωγής και συντήρησης θερμοηλ. σταθμών

Εκτενής έρευνα έχει πραγματοποιηθεί προκειμένου να κατανοηθεί η διαδικασία του χρονοπρογραμματισμού (scheduling) . Το πρόβλημα χρονοπρογραμματισμού παραγωγής και συντήρησης των σταθμών ηλ. ρεύματος είναι δε από τα δύσκολα σχετικά συνδυαστικά προβλήματα που έχουν παρουσιαστεί. Θα θεωρήσουμε ότι όλες οι δραστηριότητες υποβάλλονται σε επεξεργασία χωρίς διακοπή, δηλαδή μόλις μία δραστηριότητα αρχίσει, υποβάλλεται σε επεξεργασία μέσα σε ένα συνεχές χρονικό διάστημα.

Το κεφάλαιο αυτό είναι διαρθρωμένο ως εξής: στην ενότητα 6.1 κάνουμε μια σύντομη

περιγραφή των συστημάτων παραγωγής και του ρόλου του χρονοπρογραμματισμού σε αυτά. Στην ενότητα 6.2 γίνεται μια εκτενής περιγραφή του προβλήματος, στην ενότητα 6.3 περιγράφεται η υλοποίηση που έγινε στα πλαίσια της εργασίας αυτής και στην συνέχεια ακολουθούν τα αποτελέσματα και τα συμπεράσματα από τις εκτελέσεις των πειραμάτων μας.

Ένα μέρος των αποτελεσμάτων που παρουσιάζονται σε αυτό το κεφάλαιο έχουν δημοσιευτεί στο 5th World Multiconference on Systemics, Cybernetics and Informatics[73] και στο 7th International Conference on Information Systems Analysis and Synthesis (ISAS 2001) [70] στα πρακτικά του 5th Hellenic European Conference on Computer Mathematics and its Applications[68], στο Πανελλήνιο Συμπόσιο Αυτοματισμού, Ρομποτικής και Βιομηχανικής Παραγωγής το 2001 [67] καθώς και στο περιοδικό Mathematics and Computers in Simulation [76] το 2002.

6.1 Συστήματα Παραγωγής και Χρονοπρογραμματισμός

Οι εξελικτικοί αλγόριθμοι έχουν πια καθιερωθεί σήμερα ως μια επιτυχημένη μεθοδολογία για τη λύση πλήθους προβλημάτων χρονοπρογραμματισμού παράλληλα με τις συνήθεις τεχνικές της Επιχειρησιακής Έρευνας ενώ οι ΜΑ αποτελούν μια πολύ ενδιαφέρουσα υβριδική εξελικτική τεχνική πολλά υποσχόμενη στην επίλυση τέτοιων προβλημάτων[38, 111]. Στο γενικό πρόβλημα χρονοπρογραμματισμού παραγωγής υπάρχει ένα σύνολο εργασιών

και ένα σύνολο μηχανών. Κάθε εργασία αποτελείται από ένα σύνολο λειτουργιών (operations) που πρέπει πραγματοποιηθούν με μια καθορισμένη σειρά. Επιπλέον σε κάθε λειτουργία δίνεται ένας χρόνος επεξεργασίας και η συγκεκριμένη μηχανή στην οποία πρέπει να υποβληθεί σε επεξεργασία. Το πρόβλημα είναι να βρεθεί ένα πρόγραμμα, αν αυτό υπάρχει, στο οποίο καμία μηχανή δεν θα επεξεργάζεται περισσότερες από μία λειτουργίες την ίδια χρονική στιγμή και τέλος οι λειτουργίες να γίνονται με την απαιτούμενη σειρά. Για την επίλυση του γενικού προβλήματος χρονοπρογραμματισμού έχουν αναπτυχθεί πολλοί αλγόριθμοι, οι οποίοι χωρίζονται στους αλγορίθμους βελτιστοποίησης και στους προσεγγιστικούς αλγορίθμους (approximation algorithms) . Για την εύρεση όμως αυτής της λύσης μπορεί να χρειαστεί πολύ μεγάλος υπολογιστικός χρόνος. Αντιθέτως ένας προσεγγιστικός αλγόριθμος βρίσκει κάποια λύση σε αποδεκτά πλαίσια χρόνου. Η λύση αυτή ενδέχεται να είναι βέλτιστη, αλλά αυτό δεν είναι εγγυημένο.

Τα συστήματα παραγωγής μπορούν να διακριθούν σε τρεις βασικές κατηγορίες:

α) Συστήματα συνεχούς ροής (flow-shop) , όπου η παραγωγή εξειδικεύεται σε ένα περιορισμένο αριθμό τυποποιημένων προϊόντων, που παράγονται σε αντίστοιχες γραμμές παραγωγής και προορίζονται για ευρεία κατανάλωση. Στα συστήματα αυτά η ροή του προϊόντος σε κάθε γραμμή είναι ίδια για κάθε κομμάτι. Ο παραγωγικός εξοπλισμός οργανώνεται χωροταξικά σε γραμμική διάταξη και είναι ειδικής χρήσης, με μεγάλο βαθμό αυτοματοποίησης. Στην κατηγορία αυτή ανήκει και η περίπτωση όπου το σύστημα συμπεριφέρεται σαν μία μηχανή, όπου οι εισροές μετασχηματίζονται σε ένα ή περισσότερα προϊόντα (π.χ. ένα διυλιστήριο ή μία μονάδα παραγωγής τσιμέντου) [26, 32].

β) Συστήματα παραγωγής κατά παραγγελία (job-shop) , που παράγουν μεγάλη ποικιλία

προϊόντων σε μικρές ποσότητες και με προδιαγραφές που ορίζονται από τον πελάτη, ο οποίος αναθέτει στο σύστημα την παραγωγή ενός αριθμού ίδιων προϊόντων (παραγγελία ή εργασία). Η ροή του προϊόντος στα συστήματα αυτά είναι διαφορετική για κάθε παραγγελία (παρτίδα παραγωγής). Η χωροταξική διάταξη είναι λειτουργική, δηλαδή ο παραγωγικός εξοπλισμός, που είναι γενικής χρήσης με γενικά περιορισμένο βαθμό αυτοματοποίησης, διατάσσεται σε ομάδες παραγωγικών μονάδων που εκτελούν την ίδια λειτουργία (π.χ. χωριστά οι τόρνοι, χωριστά οι πρέσσες κ.ο.κ.) [7, 20]

γ) Συστήματα κατασκευής έργων, που παράγουν συνήθως ένα προϊόν μεγάλου μεγέθους και αξίας που προορίζεται για ένα πελάτη. Στα συστήματα αυτά ο παραγωγικός εξοπλισμός, που χαρακτηρίζεται από μικρό βαθμό αυτοματοποίησης, διατάσσεται γύρω από το προϊόν.[30, 94]

Με τον όρο χρονοπρογραμματισμό οριζεται η κατανομή δεδομένων πόρων στη διάρκεια του χρόνου με σκοπό την ολοκλήρωση ενός συνόλου εργασιών. Τα προβλήματα χρονοπρογραμματισμού παραγωγής συνδυάζουν επιστημονικές περιοχές τόσο διαφορετικές όπως είναι αυτές του σχεδιασμού παραγωγής (production planning) και του computer design . Η θεωρία και οι εφαρμογές του χρονοπρογραμματισμού παραγωγής έχουν αναπτυχθεί σε ένα σημαντικό πεδίο της έρευνας, τόσο από την πλευρά της επιχειρησιακής έρευνας όσο και από την πλευρά της τεχνητής νοημοσύνης.

Το πρόβλημα του χρονοπρογραμματισμού παραγωγής ρεύματος σε σταθμούς ηλεκτρικού ρεύματος παρουσιάζει ιδιαίτερο ενδιαφέρον για τις οικονομίες των διαφόρων χωρών αφού σχετίζεται με τον ενεργειακό τομέα. Ηλεκτρική ενέργεια μπορούμε να πάρουμε πρακτικά από κάθε άλλη μορφή ενέργειας. Παρ΄ όλα αυτά όμως δόθηκε ιδιαίτερο βάρος (όπως ήταν

φυσικό) στους τρόπους παραγωγής ηλεκτρισμού που συνέφεραν περισσότερο οικονομικά.

Το πρόβλημα του χρονοπρογραμματισμού των εργασιών των σταθμών ηλεκτρικού ρεύματος έχει μελετηθεί εκτένως στο παρελθόν και συνίσταται στην ελαχιστοποίηση του κόστους των εργασιών που γίνονται σε αυτούς και του συνολικού κόστους ζήτησης. Το 1973, ο Gruhl [101] παρουσίασε ένα γενικό πρόβλημα χρονοπρογραμματισμου που περιέλαβε τη συντήρηση σχεδιάζοντας ως υποπρόβλημα προτείνοντας μια τεχνική δυναμικού προγραμματισμού. Δύο έτη αργότερα το 1975, ο Dopazo και Merrill [80, 218] χρησιμοποίησαν μια τεχνική ακεραίου προγραμματισμού. Αυτές οι προσεγγίσεις έπασχαν από το μειονέκτημα ότι ήταν ανίκανες να αντιμετωπίσουν ακόμη και προβλήματα μέτριας κλίμακας. Το 1997 ο Burke [39] χρησιμοποιησε μια εξελικτική μέθοδο η οποία και έδωσε καλά σχετικά αποτελέσματα ενώ το 2003 ο Nara [165] προσέγγισε το πρόβλημα με μια παραλληλη υλοποίηση της μεθόδου της προσομοιούμενης ανόπτησης. Είχε προηγηθεί η σειριακή υλοποίηση από τον ίδιο παλιότερα η οποία όμως δεν απέδωσε καλά αποτελέσματα σε μεγάλης κλίμακα προβλήματα [188].

6.2 Περιγραφή του προβλήματος

Στην εργασία μας περιλαμβάνουμε μια γενική κατηγορία περιοριστικών πόρων δίνοντας ένα πρόβλημα στο οποίο οι δραστηριότητες μπορούν να έχουν περισσότερους από ένα τρόπους εκτέλεσης αφού υπάρχουν και ανανεώσιμοι αλλά και μη ανανεώσιμοι περιοριστικοί πόροι.

Το πρόβλημα αυτό είναι σχετικό και στα ευέλικτα συστήματα κατασκευής (flexible manufacturing systems) . Οι παραδοσιακές τεχνικές βελτιστοποίησης όπως ο ακέραιος προγραμματισμός δίνουν μια ακριβή βελτιστη λύση για μικρά προβλήματα.

Άλλες πιο πρόσφατες ευρετικές μέθοδοι που έχουν εφαρμοστεί για την επίλυση του προβλήματος είναι η προσομοιούμενη ανόπτηση , η στοχαστική αναζήτηση, οι μιμητικοί αλγόριθμοι, η αποτρεπτική αναζήτηση. Επιπρόσθετα η χρήση υβριδικών γενετικών αλγορίθμων με τεχνικές αναζήτησης (όπως της τοπικής αναζήτησης, της προσομοιούμενης ανόπτησης, και της αποτρεπτικής αναζήτησης) συνέβαλαν στην σημαντική βελτίωση της δυνατότητας αναζήτησης των GA και η υψηλή απόδοση που παρουσιάζουν αναφέρεται συχνά στη διεθνή βιβλιογραφία.

Εξετάζουμε i μονάδες παραγωγής ηλεκτρικού ρεύματος σε ένα χρονικό ορίζοντα M_i περιόδων. Κάθε μονάδα πρέπει να παράγει για M_i συνεχείς περιόδους. Η λειτουργική χωρητικότητα κάθε μονάδας περιγράφεται από το C_i. Κάτω από ορισμένες προϋποθέσεις είναι δυνατόν για μια μονάδα να ξεπεραστεί αυτό το όριο. Προκειμένου να αποφευχθούν οι τυχαίοι παράγοντες στο πρόβλημα, όπως για παράδειγμα οι τυχαίες διακοπές λειτουργίας των μονάδων, χρησιμοποιείται μια ακόμα μεταβλητή για τις εφεδρικές μονάδες λειτουργίας ανάλογη με προς τη ζήτηση στην περιγραφή του προβλήματος. Το πρόβλημα ταξινομείται σαν ένα πρόβλημα ελαχιστοποίησης κόστους το οποίο μπορεί να επιλυθεί με μια τεχνική βελτιστοποίησης.

Επομένως στην περίοδο j η προβλεπόμενη ζήτηση για το σύστημα καθορίζεται από το D_j και η χωρητικότητα των εφεδρικών μονάδων παραγωγής που απαιτούνται ορίζεται από το R_j. Οι δαπάνες για τα καύσιμα μπορούν επίσης να υπολογιστούν για κάθε περίοδο

σαν μια σταθερά f_j για κάθε μονάδα παραγωγής.

Τέλος με p_{ij} αναπαρίσταται η ηλεκτρική ενέργεια που παράγεται από μια μονάδα i σε μια περίοδο j, με c_{ij} συμβολίζεται το κόστος συντήρησης μιας μονάδας i αν αυτή συντηρείται στη διάρκεια της περιόδου j και τα y_{ij} παίρνουν την τιμή 1 αν η μονάδα i συντηρείται στην περίοδο j και 0 σε διαφορετική περίπτωση. Με I συμβολίζονται ο μέγιστος αριθμός μονάδων και με J ο μέγιστος αριθμός περιόδων. Ο στόχος του προβλήματος είναι η ελαχιστοποίηση του αθροίσματος του συνολικού κόστους και του συνολικού κόστους της ζήτησης.

$$\min \sum_{i=j}^{J} (f_i \cdot \sum_{i=1}^{I} p_{ij}) + \sum_{i=1}^{I} c_i(x_i)$$

Όταν ξεκινάει η μονάδα i την συντήρηση της, τότε η μονάδα αυτή πρέπει να βρίσκεται σε κατάσταση συντηρήσεως για M_i συνεχείς περιόδους.

$$\begin{gathered} y_{ij} = 0, j = 1, 2, ..., x_i - 1 \\ y_{ij} = 1, j = x_i, ..., x_i + M_i - 1 \\ y_{ij} = 0, j = x_i + M_i, ..., J \end{gathered}$$

Η συνολική ενέργεια που παράγεται διακρίνεται σε πρωτεύουσα και δευτερεύουσα. Πρωτεύουσα θεωρείται η εγγυημένη ενέργεια η οποία είναι διαθέσιμη από τις μονάδες σταθμούς. Όλη η υπόλοιπη ενέργεια που παράγεται από το σύστημα θεωρείται δευτερεύουσα και έχει μικρότερη οικονομική αξία. Αν $x_i + M_i > J$, τότε το υπόλοιπο της συντήρησης της μεταφέρεται στην επόμενη επανάληψη. Έτσι διασφαλίζεται το συνεχές της συντηρήσεως. Η παραγωγή των γεννητριών δεν πρέπει να υπερβαίνει ένα συγκεκριμένο όριο ενώ όταν η μονάδα τίθεται σε κατάσταση συντηρήσεως η παραγωγή παίρνει τιμή 0.

$$0 \leq p_{ij} \leq C_i(1 - y_{ij})$$

Η συνολική παραγωγή πρέπει να είναι ίση με τη ζήτηση σε κάθε περίοδο,

$$\sum_{i=1}^{I} p_{ij} = D_j, j = 1, 2, ..., J$$

και για την συνολική χωρητικότητα ισχύει ο παρακάτω περιορισμός.

$$\sum_{i=1}^{I} (1 - y_{ij}) C_i \geq (D_j + R_j)$$

Η περιγραφή του προβλήματος βασίζεται στην διατύπωση του προβλήματος που έγινε στην εργασία [188]. Χαρην απλοποίησεως της αλγοριθμικής διαδικασίας θεωρούμε ότι όλες οι λύσεις στον χώρο λύσεων είναι έγκυρες. Μία λύση θα μπορούσε να χαρακτηριστεί ως ανέφικτη αν οι περιορισμοί της ζήτησης και των αποθεμάτων δεν μπορούν να αντιμετωπιστούν. Σε αυτήν την περίπτωση η λύση εμποδίζεται από μια επιπρόσθετη συνάρτηση ποινής.

$$\alpha \sum_{j=1}^{J} u_j + \beta \sum_{j-1}^{J} u_j$$

όπου α και β είναι παράμετροι διορθωτικές και οι u_j y_j προέρχονται από το έλλειμα στην παραγωγή

$$(\sum_{i=1}^{I} p_{ij}) + u_j = D_j$$

και από το έλλειμμα στην χωρητικότητα.

$$(\sum_{i=1}^{I} C_i (1 - y_{ij})) + u_j = D_j + R_j$$

τα u_j δεν επιτρέπεται να είναι αρνητικά και κατά συνέπεια σε μια εφικτή λύση δεν εφαρμόζεται η συνάρτηση ποινής. Κατά συνέπεια οποιαδήποτε αρχική λύση μπορεί να επιλεχτεί και ο αλγόριθμος βελτιστοποίησης θα κατευθυνθεί προς τις εφικτές λύσεις μέσω της επιλογής αρκετά υψηλών α και β.

6.3 Υλοποίηση

Το πρόβλημα είναι η ελαχιστοποίηση του κόστους όλων των εργασιών ενός δικτύου σταθμών ηλ. ρεύματος. Για να γίνει πιο κατανοητή η ανάλυση που θα ακολουθήσει, έχουμε επιλέξει σαν παράδειγμα ένα σύστημα με έξι σταθμούς και έξι εργασίες. Τα δεδομένα μας τα έχουμε πάρει από την βιβλιογραφία και βασίζονται στην εργασία του Burke [39] και αφορούν ενα σύστημα με εξήντα σταθμούς και εξήντα εργασίες. Η υλοποίηση βασίζεται στην μοντελοποίηση του προβλήματος που περιγράφτηκε στην προηγούμενη ενότητα και με τις μεθόδους PMA1 - PMA4 μπορέσαμε να καταλήξουμε σε μία πολύ καλή με μικρό χρόνο υλοποίησης των λειτουργιών συντήρησης. Η πλατφόρμα για την πειραματική μας μελέτη ήταν μια συστοιχία από 18 σταθμούς εργασίας συνδεδεμένη με δίκτυο 100 Mb /s Fast Ethernet. Ο υπολογιστικός χρόνος που χρειάστηκε ήταν μόλις μερικά δευτερόλεπτα και για τις τέσσερις μεθοδους σε αντίθεση με άλλες μεθόδους οι οποίες για το ίδιο μέγεθος προβλήματος (60*60) χρειάζονται πολύ περισσότερο χρόνο. Κάθε εργασία αποτελείται από εξήντα λειτουργίες, όσοι δηλαδή και οι σταθμοί.

Οι PMA1 $\sim$ PMA4 χρησιμοποιούνται ώστε να εμπλέκουν τα χρωμοσώματα που κατασκευάζονται και να βρίσκουν αποδοτικότερη ακολουθία κανόνων προτεραιότητας υλοποίησης των λειτουργιών παραγωγής και συντήρησης. Στις επόμενες παραγράφους αναφερόμαστε στην αναπαραστάση που χρησιμοποιηθηκε στη δική μας υλοποίηση.

Η αναπαράσταση λειτουργιών (operation - based representation) [176] κωδικοποιεί μία λύση ως μία σειρά λειτουργιών παραγωγής και κάθε γονίδιο αντιπροσωπεύει την θερμική ενέργεια που παράγεται από την καύση πετρελαίου. Ένας απλός τρόπος να ονομάσουμε

τις λειτουργίες παραγωγής είναι να δώσουμε σε καθεμία ένα φυσικό αριθμό. Δυστυχώς όμως λόγω της ύπαρξης των περιορισμών προτεραιοτήτων συντήρησης (precedence constraints) και της χωρητικότητας των μονάδων δεν αντιστοιχούν όλες οι μεταθέσεις αυτών των αριθμών σε μια χρονικη περιοδο.

Μία εναλλακτική λύση είναι να ονομαστούν όλες οι λειτουργίες με δύο ψηφία. Το δεύτερο ψηφίο θα είναι το ίδιο για όλες τις λειτουργίες παραγωγής της ίδιας μονάδας παραγωγής. Το πρώτο ψηφίο θα δηλώνεται σύμφωνα με τη σειρά εμφάνισης της λειτουργίας παραγωγής στο χρωμόσωμα. Για παράδειγμα, η δεύτερη κατά σειρά λειτουργία της τρίτης εργασίας θα δηλώνεται ως 23 ενώ η τρίτη λειτουργία της ίδιας εργασίας ως 33. Για περισσότερη ευκολία βέβαια μπορούμε να παραλείψουμε το πρώτο ψηφίο και αυτό να υπονοείται κάθε φορά που συναντάμε την ίδια εργασία στο πρόγραμμα.

Αρχικά μεταφράζουμε το χρωμόσωμα σε μία λίστα ταξινομημένων λειτουργιών που περιλαμβάνουν και την μετάδοση κίνησης από τον ατμοστρόβιλο στην γεννήτρια, μετά παράγουμε το πρόγραμμα βήμα βήμα προγραμματίζοντας πρώτα την πρώτη λειτουργία παραγωγής της λίστας, έπειτα τη δεύτερη λειτουργία παραγωγής της λίστας κ.ο.κ. Κάθε λειτουργία παραγωγής τοποθετείται στον καλύτερο διαθέσιμο χρόνο επεξεργασίας της αντίστοιχης γεννήτριας που απαιτείται να εκτελέσει τη λειτουργία και δεν βρίσκεται σε συντηρηση. Η διαδικασία επαναλαμβάνεται έως ότου όλες οι λειτουργίες παραγωγής και συντήρησης προγραμματιστούν. Ένα πρόγραμμα που παράγεται με αυτή τη διαδικασία είναι εγγυημένο ότι αποτελεί ένα ενεργό πρόγραμμα (active schedule) .

Τα δεδομένα που έχουμε αρχικά και θα χρησιμοποιήσουμε για την εφαρμογή των μεθόδων μας είναι δύο πίνακες οι οποίοι περιγράφουν τους χρόνους παραγωγής και τη σειρά

100	350	650	700	340	600
80	50	110	110	130	40
55	64	88	99	10	87
45	45	45	63	88	99
49	43	45	44	43	10
350	305	95	100	40	10

Πίνακας 6.1: Χρόνοι επεξεργασίας

με την οποία οι λειτουργίες συντήρησης πρέπει να πραγματοποιηθούν στους σταθμούς του συστήματος. Ακόμα ως δεδομένο έχουμε και τις δαπάνες για τα καύσιμα. Ο πίνακας 6.1 περιγράφει τους χρόνους που χρειάζονται οι διάφορες λειτουργίες καύσης πετρελαιου και των άλλων διαδικασιών που απαιτούνται για την παραγωγή ηλεκτρικής ενέργειας σε ένα συστημα 6*6.

Η πρώτη γραμμή του πίνακα αναφέρεται στους χρόνους επεξεργασίας των λειτουργιών της πρώτης μονάδας (το στοιχείο 11 αναφέρεται στο χρόνο της πρώτης κατά σειρά λειτουργίας της πρώτης μονάδας, το 12 στη δεύτερη λειτουργία, το 13 στην τρίτη λειτουργία, το14 στην τέταρτη λειτουργία, το 15 στην πέμπτη λειτουργία και το 16 στην έκτη λειτουργία). Όμοια η δεύτερη γραμμή αναφέρεται στη δεύτερη εργασία και η τρίτη γραμμή στην τρίτη εργασία, η τέταρτη γραμμή στην τέταρτη εργασία η πέμπτη γραμμή στην πέμπτη εργασία και η έκτη γραμμή στην έκτη εργασία. Ο πίνακας 6.2 που περιγράφει τη σειρά με την οποία οι λειτουργίες συντήρησης πρέπει να πραγματοποιηθούν στους σταθμούς στο παράδειγμα μας.

3	1	2	4	6	5
2	3	5	6	1	4
3	4	6	1	2	5
2	1	3	4	5	6
3	2	5	6	1	4
2	4	6	1	5	3

Πίνακας 6.2: Σειρά λειτουργιών συντήρησης

Η πρώτη γραμμή αναφέρεται στη σειρά με την οποία πρέπει να γίνουν οι λειτουργίες συντήρησης του πρώτου σταθμού (το στοιχείο 11 δείχνει ότι η πρώτη λειτουργία της πρώτης μονάδας θα πρέπει να πραγματοποιηθεί στο σταθμό 1, το στοιχείο 12 δείχνει ότι η δεύτερη λειτουργία της πρώτης μονάδας θα πρέπει να πραγματοποιηθεί στο σταθμό 2 κ.ο.κ.). Για παράδειγμα, το στοιχείο 32 δείχνει ότι η δεύτερη λειτουργία της τρίτης μονάδας θα πρέπει να πραγματοποιηθεί στο σταθμό 1.

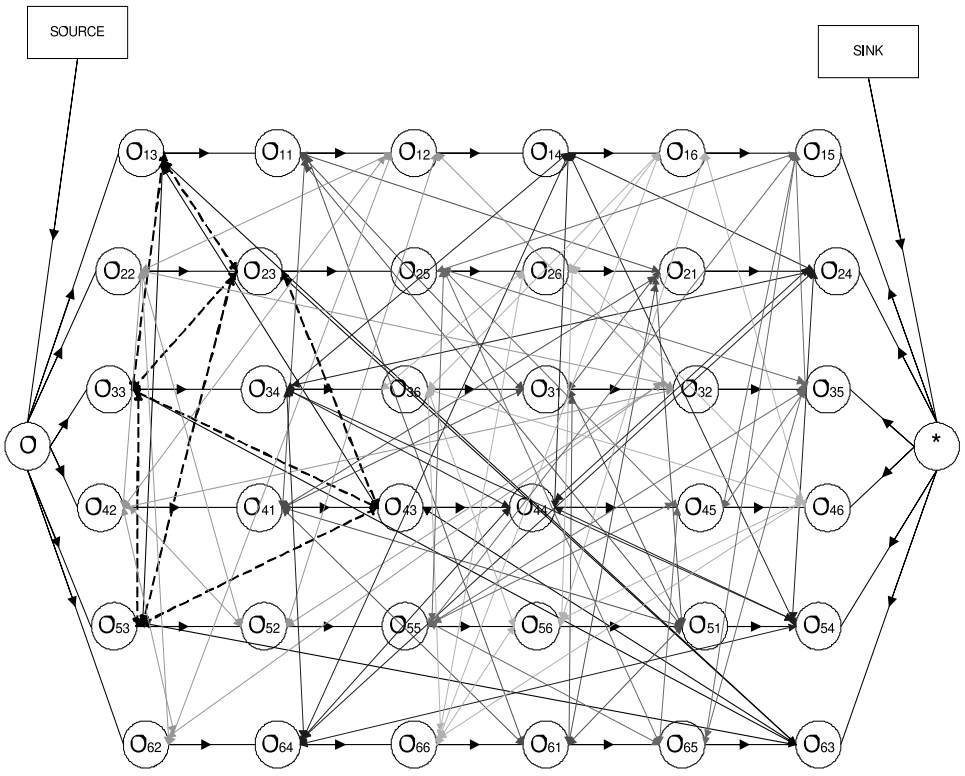
Γενικότερα είναι δύσκολο να προσδιοριστούν τυπικά τα κριτήρια σύγκλισης. Ενδέχεται η καταλληλότητα ενός πληθυσμού να παραμένει στάσιμη για ένα μεγάλο αριθμό επαναλήψεων προτού μία εξαιρετική λύση βρεθεί στη συνέχεια. Για αυτό το λόγο η εφαρμογή ενός συμβατικού κριτηρίου τερματισμού (για παράδειγμα να έχουμε τερματισμό όταν η λύση που θα βρεθεί θα είναι κάτω, για ένα πρόβλημα ελαχιστοποίησης, από κάποιο ποσοστό της τιμής της αρχικής λύσης) γίνεται προβληματική. Μία κοινή πρακτική είναι να τερματίζεται ο αλγόριθμος μετά από ένα προκαθορισμένο αριθμό επαναλήψεων και στη συνέχεια να ελέγχεται η ποιότητα των καλύτερων μελών του πληθυσμού (οι καλύτερες λύσεις) σε σχέση με τον ορισμό του εκάστοτε προβλήματος. Εάν καμία αποδεκτή λύση

δε βρεθεί τότε ο αλγόριθμος μπορεί να ξαναρχίσει μία καινούργια αναζήτηση.

Για την αρχικοποίηση του πληθυσμού δημιουργείται ένας πίνακας διαστάσεων m x n, όπου m είναι ο πληθυσμός των πιθανών λύσεων για το πρόβλημα και n το μέγεθος της κάθε λύσης. Το m καθορίζεται από το χρήστη και ένας καλός πληθυσμός έχει μέγεθος από 200 έως 350 άτομα, ανάλογα με το μέγεθος του κάθε προβλήματος. Το n επίσης καθορίζεται από το μέγεθος του προβλήματος. Ο πίνακας αυτός περιέχει τον αρχικό πληθυσμό από τον οποίο θα αρχίσει η εφαρμογή του αλγορίθμου. Η κάθε σειρά του αποτελεί τυχαία λύση για το πρόβλημα. Στη συγκεκριμένη εφαρμογή δημιουργούμε έναν πίνακα ο οποίος αποτελείται από 350 γραμμές (λύσεις) και 36 στήλες (συνολικά τα στοιχεία που αποτελούν τη κάθε λύση).

Η εισαγωγή του αρχικού πληθυσμού παρήγαγε τυχαία ακεραίους αριθμούς. Με αυτό τον τρόπο όμως ενδέχεται να δημιουργηθούν λύσεις που δεν ανταποκρίνονται σε πραγματοποιήσιμο πρόγραμμα. Για παράδειγμα στη δική μας περίπτωση θα μπορούσε να είχε δημιουργηθεί λύση 3 2 1 4 7 5 6 8 9.....35 36. Η συγκεκριμένη λύση δεν είναι πραγματοποιήσιμη γιατί έχει την τρίτη λειτουργία της πρώτης εργασίας (αριθμός 3) να γίνεται πρώτα από τις δύο προηγούμενές της, την 1 και 2. Γενικότερα, ένα πρόγραμμα (πιθανή λύση) είναι αποδεκτό μόνο εάν για κάθε εργασία οι λειτουργίες της ακολουθούν τη σωστή σειρά, δηλαδή για την πρώτη εργασία το 1 να προηγείται του 2 και το τελευταίο του 3.

Για να μπορέσουμε λοιπόν να διασφαλίσουμε ότι οι αρχικές λύσεις είναι αποδεκτές και πραγματοποιήσιμες με βάση και τη συνολική ζήτηση σε ενέργεια εισάγουμε με τυχαίο τρόπο τις ακολουθίες των λειτουργιών αλλά η απαιτούμενη σειρά είναι αποτέλεσμα μιας



Σχήμα 6.1: Γραφος αναπαράστασης συστηματος θερμοηλεκτρικών σταθμών

από τις τρείς ευρετικές μεδοδους (SA, TS, GLS ανάλογα με την υλοποίηση) για κάθε εργασία.

Στο επόμενο βήμα υπολογίζεται ο χρόνος ολοκλήρωσης υπολογίζεται με τη βοήθεια της απεικόνισης του συστήματος με τον γράφο του σχήματος 6.1. Οι κόμβοι του γράφου αντιπροσωπεύουν λειτουργίες εργασιών . Εκτός από τις 36 λειτουργίες (για ένα σύστημα πχ $6x6$) περιέχονται και άλλοι δύο κόμβοι, ο κόμβος έναρξης (source) και ο κόμβος τερματισμού (sink) . Για να δημιουργήσουμε μία λύση θα πρέπει τα διακεκομμένα τόξα να μετατραπούν σε κατευθυνόμενα που θα δείχνουν τη σειρά των εργασιών αυτών στον

κάθε σταθμό.

Στην υλοποίηση μας για κάθε χρωμόσωμα έχουμε ως έξοδο έναν πίνακα $Graphos(n)$, ο οποίος περιέχει πληροφορίες σχετικές προτεραιότητας και τους χρόνους ολοκλήρωσης των λειτουργιών για κάθε μία από τις μονάδες καθώς και πληροφορίες για τη σειρά που πρέπει να ακολουθήθει σχετικά με την διαδικασία συντήρησης των διάφορων σταθμούς ηλ. ρεύματος στην διάρκεια μια χρονικής περιόδου. Δέχεται για είσοδο τον πίνακα που περιέχει τον συνολικό αριθμό των χρωμοσωμάτων αφού πρώτα ορίσουμε τις διαστάσεις του συστήματος πχ $(6x6)$ τους κόμβους αρχή και τέλους. Στο επόμενο στάδιο ορίζουμε και αρχικοποιούμε τον πίνακα $Graphos$ με την τιμή -1, είναι ένας πίνακας διαστάσεων m^2+2xm^2+2 ο οποίος στο τέλος του προγράμματος θα μας δώσει πληροφορίες σχετικά με τους χρόνους ολοκλήρωσης των λειτουργιών παραγωγής για κάθε μία από τις μονάδες καθώς και πληροφορίες για τις δαπάνες σε καύσιμα. Στη συνέχεια δίνουμε την τιμή 0 στο μήκος τόξου από την πηγή κόμβος 0 στις πρώτες λειτουργίες της κάθε εργασίας. Επιπλέον ορίζουμε τον κόμβο 1 ως αρχή του χρόνου υπολογισμού των εργασιών και τον τελευταίο κόμβο (m^2+2) ως το τέρμα. Μετά από την ολοκλήρωση των παραπάνω για κάθε σταθμό αποθηκεύεται η λίστα με τη σειρά των λειτουργιών που πρέπει να πραγματοποιηθούν. Στην συνέχεια δημιουργείται ο πίνακας $Alloo(n)$ με (n=1,2,3,...,m), είναι ένας πίνακας γραμμή που ταξινομεί τους κόμβους του γράφου και επιστρέφει τη σειρά των κόμβων.

Για κάθε στοιχείο του πίνακα $Graphos(i,j)$ με (i=1,2,3,...,m) και (j=1,2,3,...,m) που είναι διάφορο του -1 προσθέτει σε κάθε στήλη τον αριθμό 1. Ο πίνακας $Alloo(n)$ έχει μία γραμμή με m^2+2 στοιχεία αφού εκτός από τους m^2 κόμβους λειτουργίες περιέχει

και τους δύο κόμβους έναρξης και τερματισμού.

Για την κατανόηση των παραπάνω παραθέτουμε το ακόλουθο παράδειγμα: Ως όρισμα έχουμε μία πιθανή ζήτηση σε ηλεκτρική ενέργεια. Έστω στο συγκεκριμένο παράδειγμα το όρισμα είναι η πρώτη γραμμή του πίνακα. Ώς έξοδο έχουμε τον πίνακα $Graphos1$, ο οποίος εμπεριέχει πληροφορίες σχετικές με τη σειρά προτεραιότητας των λειτουργιών ζήτησης και τους χρόνους ολοκλήρωσης της διαδικασίας συντηρησης για κάθε μία από τις μονάδες καθώς και πληροφορίες για τη σειρά που πρέπει να ακολουθήσει κάθε εργασία στους διάφορους σταθμούς. Ο $Graphos1$ είναι $(38x38)$ για ένα σύστημα $(6x6)$ και ένα τμήμα του είναι της μορφής

$$\begin{array}{cccccccccccc}
-1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 \\
-1 & -1 & -1 & 3 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 \\
-1 & -1 & -1 & -1 & 6 & -1 & -1 & 6 & -1 & -1 & -1 & -1 \\
-1 & -1 & -1 & -1 & -1 & 7 & -1 & -1 & -1 & -1 & -1 & -1 \\
-1 & -1 & -1 & -1 & -1 & -1 & -1 & 7 & -1 & -1 & -1 & -1
\end{array}$$

Στην συνέχεια κάνουμε ελαχιστοποίηση της αντικειμενικής συνάρτησης υπολογίζουμε το χρόνο ολοκλήρωσης των εργασιών παραγωγής για όλους τους κόμβους, δηλαδή χρησιμοποιωντας την τοπολογική σειρά των κόμβων. Οι χρόνοι ολοκλήρωσης είναι υπολογισμένοι για τους κόμβους σε αυτή τη σειρά με βάση τον υπολογισμό της μεγαλύτερης διαδρομής. Ο χρόνος ολοκλήρωσης ενός κόμβου υπολογίζεται μόνο μετά από το χρόνο ολοκλήρωσης όλων των προκατόχων κόμβων του έχουν υπολογιστεί. Αφού υπολογίσουμε τους χρόνους ολοκλήρωσης όλων των χρωμοσωμάτων του πληθυσμού, δημιουργούμε έναν πίνακα

Καθόρισε k αρχικές νησίδες επεξεργαστών $P^0, ..., P^{k-1}$ και διένειμε ένα αρχιπέλαγος σε k αρχικές νησίδες $pop^0, ..., pop^{k-1}$
Γενεά = 0
Για κάθε νησίδα επεξεργαστών P^i διαίρεσε την νησίδα pop^i σε g υποπληθυσμούς $subpop^j, ..., subpop^{g-1}$
Γενεά = Γενεά + 1
Εφάρμοσε μια από τις μεθόδους (βλέπε 5.3.2, 5.3.3, 5.3.4) σε κάθε pop^i.
Αν έχει σχηματιστεί ο απαραίτητος αριθμός καλών λύσεων της νησίδας pop^i αυτές μεταναστεύουν στο $pop^{(i+1)modk}$
Αν τουλάχιστον μία λύση παρελήφθη από το $pop^{(i-1)modk}$
τότε η λυση που παραλήφθηκε τοποθετείται στο pop^i
Ελέγχεται αν κάποια νησίδα ικανοποιεί τη συνθήκη τερματισμού και αν όχι τότε επιστρέφουμε στο βήμα 3.

Σχήμα 6.2: Μοντέλο Νησίδων Μιμητικών Αλγορίθμων σε Νησίδες επεξεργαστών

Objecto στήλη, που περιλαμβάνει αυτούς τους χρόνους. Ο πίνακας έχει τόσες γραμμές όσος είναι και ο πληθυσμός που χρησιμοποιείται.

Ο αλγόριθμος που παρατίθεται παρακάτω περιγράφει τον καθορισμό των νησίδων στην υλοποίηση μας και τον καθορισμό της μεθόδου εξέλιξης των νησίδων. Ειδικά για την PMA2 παρατηρήθηκε ότι κατά την διάρκεια της εξέλιξης των διαφόρων πληθυσμών είναι ανώφελη η διατήρηση μιας υπερβολικά μεγάλης θερμοκρασίας η οποία έχει ως αποτέλεσμα τη δραματική μείωση της ταχύτητας του αλγορίθμου. Για το λόγο αυτό, σε κάθε κύκλο ελέγχεται αν η θερμοκρασία ξεπερνά την απόκλιση μεταξύ μέγιστης και ελάχιστης τιμής κατά έναν παράγοντα ξ, ο οποίος ρυθμίζει τη διαδικασία ανόπτησης κατά τα αρχικά κυρίως στάδια του αλγορίθμου, οπότε ο ρυθμός εξέλιξης του πληθυσμού είναι ταχύτερος (αυτό συμβαίνει επειδή οι πολύ κακές λύσεις εύκολα εντοπίζονται και αντικαθίστανται).

Μια τυπική τιμή της παραμέτρου, η οποία προτείνεται μετά από διερεύνηση, είναι ξ= 6. Για την επιλογή της κατάλληλης συνάρτησης μετάλλαξης πραγματοποιήθηκε εκτεταμένη έρευνα. Κατάλήξαμε στη χρήση της σχέσης $x = c + d_{max}\frac{y}{|y|}$. Συνίσταται στη γέννηση μιας τυχαίας λύσης x στο όριο της υπερσφαίρας που ορίζεται από το κέντρο του πληθυσμού c και μια ακτίνα d_{maxb} παρουσιάζει το χαρακτηριστικό ότι το σημείο που κείται πάντοτε πάνω στο νοητό σύνορο του τρέχοντος πληθυσμού, το οποίο ορίζεται ως η ελάχιστη υπερσφαίρα που περικλείει όλα τα σημεία του πληθυσμού. Με τον τρόπο αυτό, στα αρχικά στάδια του αλγορίθμου, οπότε η διασπορά του πληθυσμού είναι πολύ μεγάλη, το σημείο μετάλλαξης γεννάται πρακτικά κοντά στα όρια του εφικτού χώρου. Αντίθετα κατά τα τελικά στάδια του αλγορίθμου, η μετάλλαξη δημιουργεί μικρές διαταραχές γύρω από την περιοχή της βέλτιστης λύσης όπου βρίσκονται συγκεντρωμένα όλα τα σημεία του πληθυσμού, συμβάλλοντας στην επιτάχυνση της διαδικασίας σύγκλισης.

6.4 Αποτελέσματα

Στην ενότητα αυτή παρατίθενται τα αποτελέσματα των πειραμάτων με την μορφή τεσσάρων γραφικών παραστάσεων για κάθε υλοποίηση. Στα σχήματα 6.3, 6.4, 6.5 και 6.6 δίνονται η καλύτερη, η χειρότερη και η μέση ποιότητα που βρέθηκε μέσα από ένα σύνολο εκατό πειραμάτων. Έχουμε χρησιμοποιήσει για δομή γειτονιάς του πληθυσμού δομή δακτυλίου. Επίσης σε διαγράμματα των σχημάτων 6.3, 6.4, 6.5 και 6.6 παρουσιάζονται και οι χρόνοι από το κάθε σύνολο πειραμάτων καθώς και το πλήθος των γενιών που

εξελίχθησαν συνολικά σε κάθε πείραμα. Τα αποτελέσματα αυτών των σχημάτων αφορούν εκτελέσεις που έγιναν και στους 18 επεξεργαστές της συστοιχίας. Στα σχήματα 6.9 έχουμε συγκριτικά αποτελέσματα για το ίδιο πρόβλημα αλλών δύο υλοποιήσεων (παράλληλης προσομοιούμενης ανόπτησης και παράλληλης αποτρεπτικής αναζήτησης) με τις δικές μας. Οι υλοποίησεις αυτές εγιναν από τους Nara [165] και Burke [39]. Τα δεδομένα που χρησιμοποιήθηκαν για την παρούσα μελέτη ήταν τα ίδια με αυτά της εργασίας του Burke [39]. Κατά την διάρκεια των πειραμάτων μας διαπιστώθηκε ότι οι μεγάλοι πληθυσμοί μπορεί να δημιουργήσουν προβλήματα καθώς κινούνται αργά προς τη βέλτιστη λύση και δημιουργούν πολλά παιδιά σε κάθε γενιά με αποτέλεσμα να χρειάζονται πολλές επαναλήψεις. Πολλές φορές οδηγούν σε πιο ακριβή λύση αλλά δεν δίνουν πάντα τη καλύτερη απάντηση στο πρόβλημα. Ο λόγος για αυτό είναι ότι συχνά με τη χρήση μεγάλου πληθυσμού ο αλγόριθμος παραβλέπει μία λύση με συνάρτηση κόστους άνω του μέσου όρου η οποία βρίσκεται σε μία στενή κορυφή του πεδίου αναζήτησης, καθώς έχει την τάση να οδηγείται σε πιο ευρείες κορυφές. Συνεπώς είναι λογικό να αντιμετωπίζουμε ένα πρόβλημα με πολλές επαναλήψεις της διαδικασίας βελτιστοποίησης με τη χρήση μικρού πληθυσμού παρά με μία επανάληψη με μεγάλο αφού και στις δύο περιπτώσεις θα χρειαστούν τον ίδιο χρόνο εξομοίωσης.

Από την άλλη πλευρά, πολύ μικροί πληθυσμοί οδηγούν σε πρώιμη σύγκλιση. Η μείωση του πληθυσμού οδηγεί σε μείωση της ποικιλότητας των ατόμων με αποτέλεσμα ο αλγόριθμος να μην έχει τη δυνατότητα να ψάξει σε μεγάλο τμήμα του πεδίου αναζήτησης και να παγιδευτεί. Στα αποτελέσματα (βλ. σχήμα 6.9) παρατηρούμε μια απότομη πτώση των καμπυλών επιτάχυνσης που οφείλεται στην ετερογένεια του περιβάλλοντος μας.

PMA1	1201282
PMA2	1201513
PMA3	1201279
PMA4	1197332
Burke	1201296
Nara	1202522
PSA	1201296
PTS	1201522
PEA	1201342

Πίνακας 6.3: Οι καλύτερες τιμές για 9 μεθόδους

Στο σχήμα 6.9 παρατηρούμε ότι η PMA2 είναι λιγοτερο αποτελεσματική από τις υπόλοιπες τρεις υλοποιήσεις ενώ παρατηρείται και μια απότομη πτώση στους 12 σταθμούς εργασίας στην καμπύλη επιτάχυνσης της υλοποίησης PMA3 κάτι το οποίο οφείλεται στο ότι υπάρχει μια ανισορροπία του φόρτιου εξαιτίας της τεχνικής υλοποίησης της απαγορευμένης λίστας. Ο καθορισμός των γειτονικών λύσεων απαιτεί μια διαδικασία διακριτοποίησης του εφικτού χώρου κάτι το οποίο κάνει τον αργό σταθμό εργασίας να τελειώνει πάντα τελευταίος και να υπάρχει ένας σημαντικός χρόνος αδράνειας για τους ταχύτερους σταθμούς εργασίας σε ένα ετερογενές περιβάλλον.

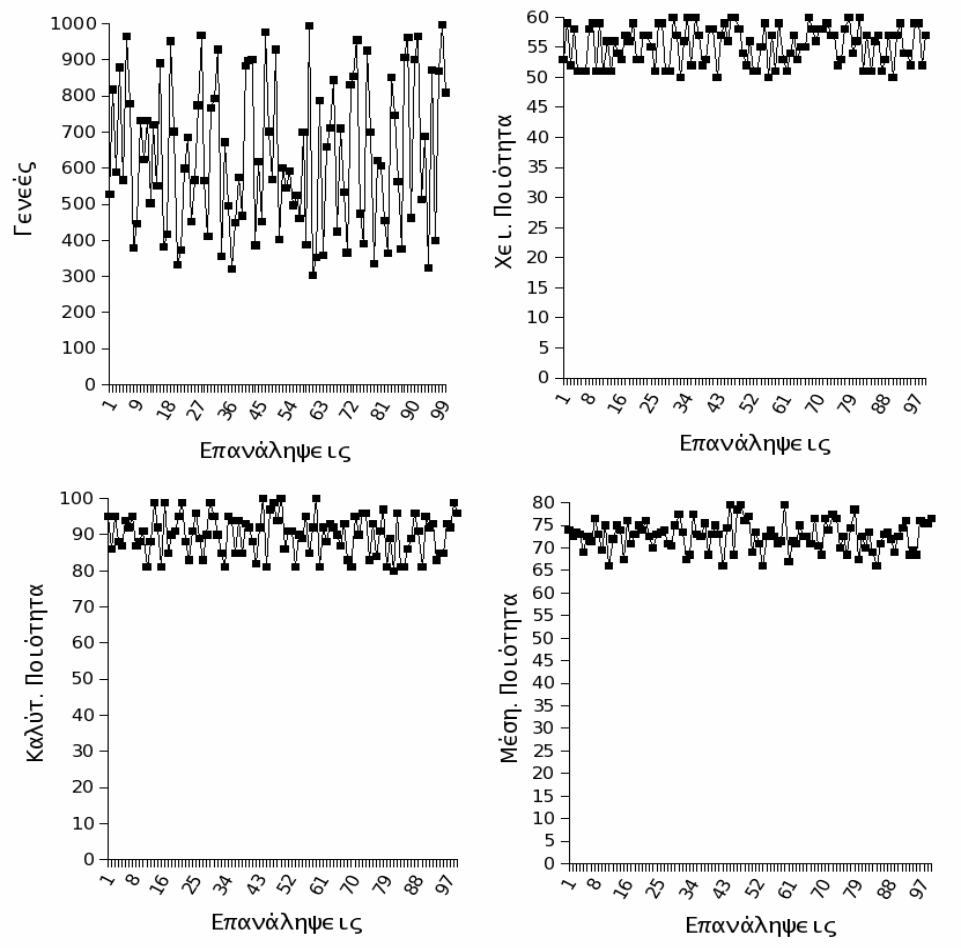
Οι καλύτερες τιμές που είχαμε για τις μεθόδους που δοκιμάσαμε φαίνονται στον πίνακα 6.3.

Παρατηρώντας τα αποτελέσματα βλέπουμε ότι υπάρχει μια αντίστροφη σχέση ανάμεσα στον παράλληλο χρόνο εκτέλεσης και στον αριθμό των σταθμών εργασίας. Ο λόγος του

χρόνου υπολογισμού προς το χρόνο επικοινωνίας εξαρτάται ως ένα βαθμό απο την ομοιογένεια των αλγορίθμων που εκτελούνται στις διάφορες νησίδες επεξεργαστών. Χρησιμοποιώντας ετερογενείς αλγορίθμους στην PMA4 παρατηρούμε ότι ο παράλληλος χρόνος εκτέλεσης των υλοποιήσεων μας βελτιώνεται σε σχέση με τον παράλληλο χρόνο εκτέλεσης με χρήση ομοιογενών αλγορίθμων. Ο χρόνος όμως επικοινωνίας είναι με μικρές αποκλίσεις περίπου ο ίδιος εφόσον διατηρούμε το μέγεθος πληθυσμού σταθερό. Στα διαγράμματα του σχήματος 6.8 παρουσιάζουμε τις επιταχύνσεις σε σχέση με τη καταγραφή του ιστορικού εξέλιξης χρησιμοποιώντας της μεθόδους PMA1, PMA2, PMA3, PMA4. Παρατηρούμε ότι υπάρχει μια σημαντική επίδραση στην απόδοση των υλοποίησεων μας. Συγκεκριμένα βλέπουμε ότι με την καταγραφή του ιστορικού εξέλιξης ο χρόνος επικοινωνίας αυξάνεται. Η επιβάρυνση αυτή όμως δεν φαίνεται να επηρεάζει στην συνολική απόδοση των παράλληλων υλοποίησεων. Συνεπώς ο λόγος του χρόνου υπολογισμού προς τον χρόνο επικοινωνίας είναι αρκετά υψηλός. Τα καλύτερα αποτελέσματα απόδοσης προκύπτουν για τις μεθόδους PMA1, PMA3, PMA4 ενώ λιγότερο καλά είναι τα αποτελέσματα για την PMA2.

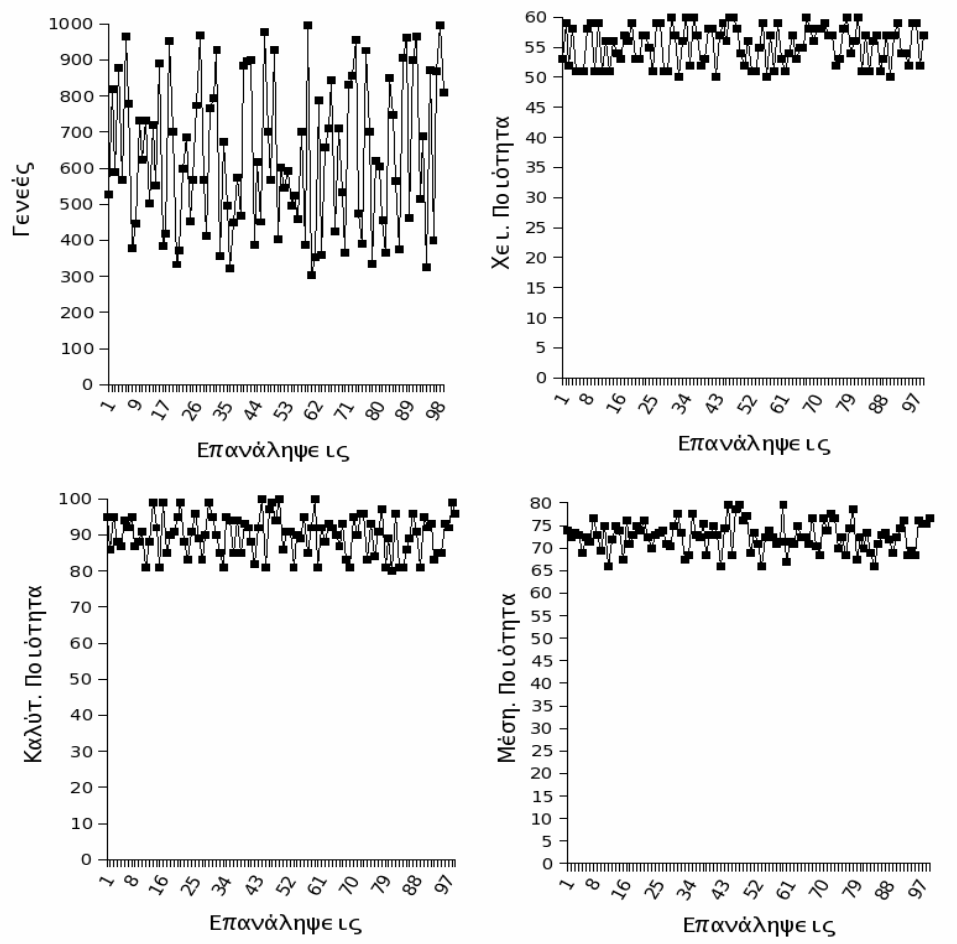
Οι επιταχύνσεις που πραγματοποιούνται οφείλονται όχι μόνο στον παραλληλισμό αλλά και στον χρόνο που επιτυγχάνεται με την αύξηση των αριθμών των νησίδων. Στην εργασία μας η εννοια της νησίδας διαφοροποιείται από την έννοια της παραλληλοποιηση ενός ΜΑ.

Στη διαρκεια των πειραμάτων μας διαπιστώσαμε ότι η μη συχνή μετανάστευση βοηθά στην αποφυγή της πρόωρης σύγκλισης του συνόλου των πληθυσμών. Επιπλέον η μετανάστευση σε συνδυασμό με την χρήση μιας μεθοδου τοπικής αναζήτησης βοηθά στο να μην χάνεται η ποικιλία του γενετικού υλικού και να μην γίνεται πρόωρη σύγκλιση σε



Σχήμα 6.3: Αποτελέσματα της PMA1 για 100 επαναλήψεις

κάποιο τοπικό βέλτιστο. Η χρήση μιας μεθόδου τοπικής αναζήτησης δημιουργεί νέες ελπιδοφόρες περιοχές αναζήτησης λύσεων (βλ. σχήμα 6.7α). Αυτό είναι αρκετά σημαντικό γενικά, καθότι αυτές οι λύσεις αρχικά είναι τοπικές, που είναι και ο συνήθης κανόνας με τις ποικιλίες στη φύση, έτσι ώστε όμοια τροποποιημένα άτομα συχνά αναπαράγονται μεταξύ τους. Εάν ο νέος πληθυσμός είναι επιτυχής στον αγώνα για επιβίωση , θα διαδοθεί αργά, ανταγωνιζόμενος τους υπόλοιπους και κατακτώντας άτομα στα όρια ενός διαρκώς

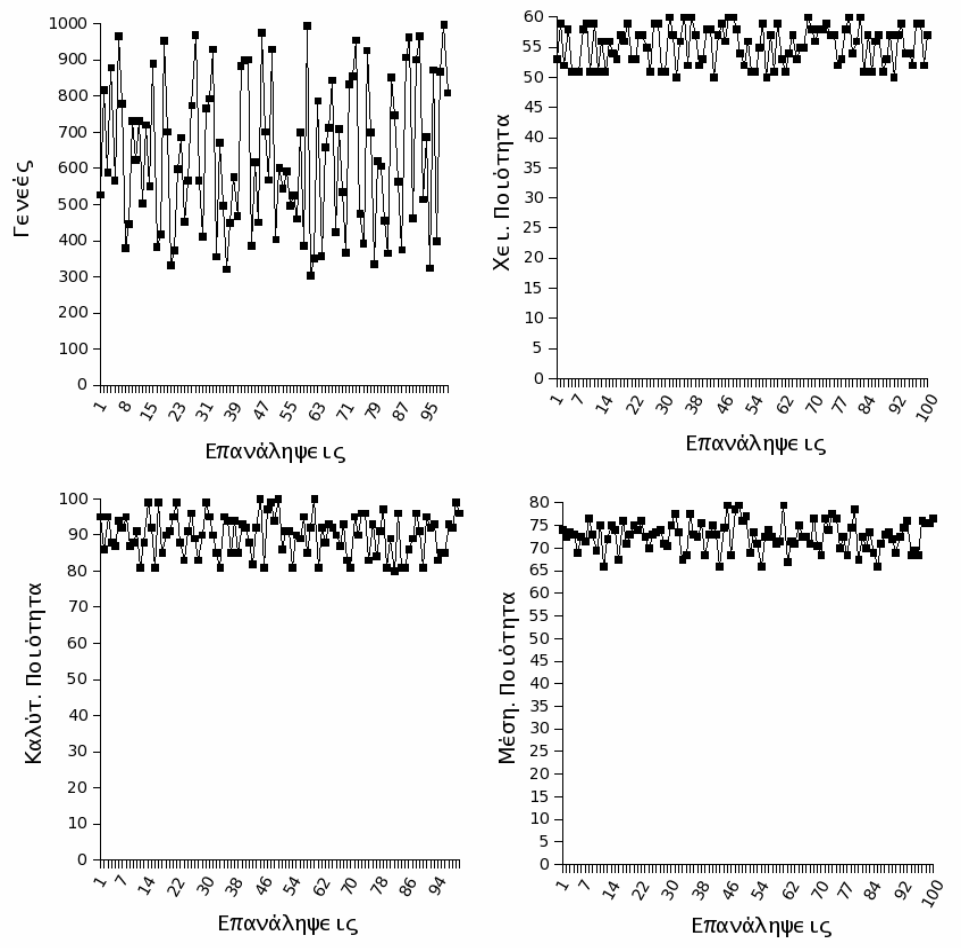


Σχήμα 6.4: Αποτελέσματα της PMA2 για 100 επαναλήψεις

αυξανόμενου κύκλου.

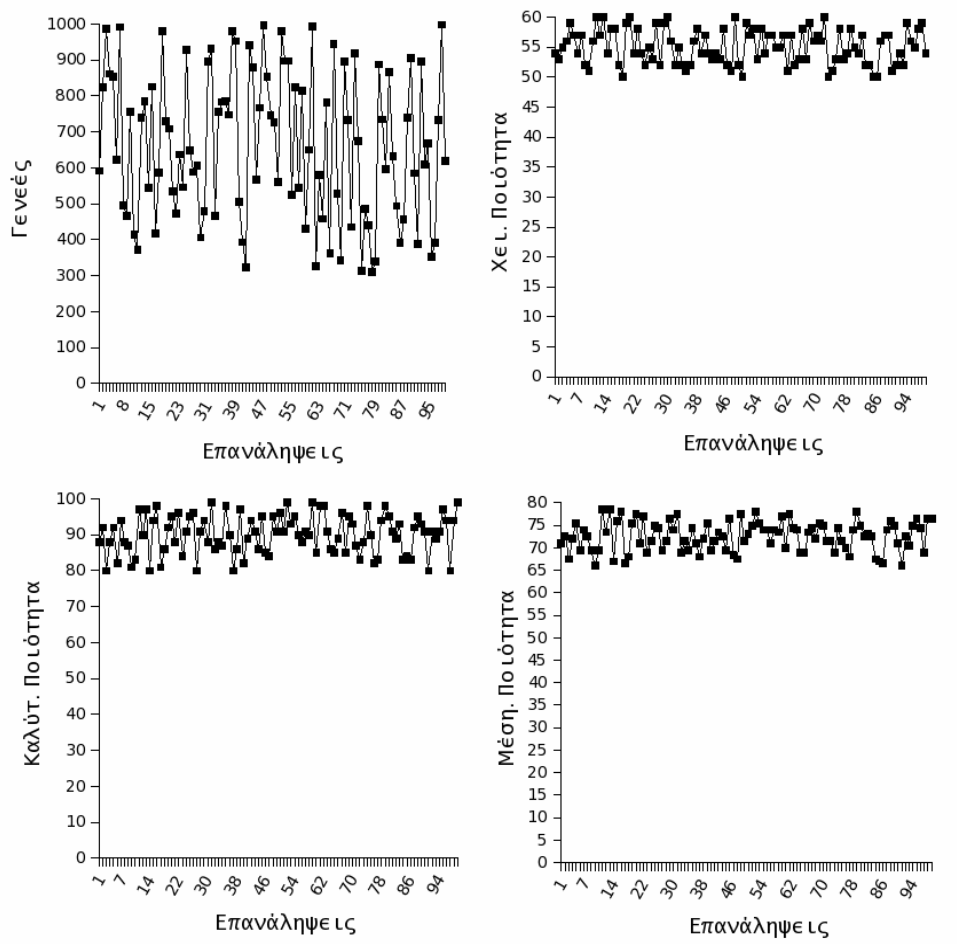
6.5 Συμπεράσματα

Στο πρόβλημα χρονοπρογραμματισμού παραγωγής ρεύματος πρέπει ένα σύνολο εργασιών, συνδεδεμένων με χρονικά όρια παροχής, να υποβληθούν σε επεξεργασία στους σταθμούς



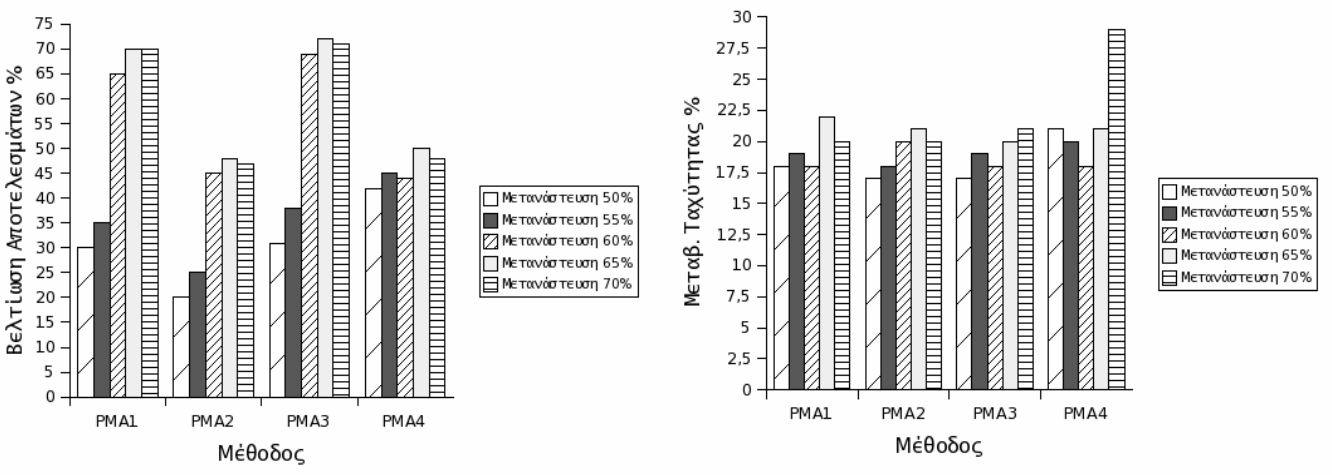
Σχήμα 6.5: Αποτελέσματα της PMA3 για 100 επαναλήψεις

παραγωγής και μάλιστα αυτή η επεξεργασία πρέπει να γίνει με προκαθορισμένη σειρά. Από τα αποτελέσματα, γίνεται φανερή η υπεροχή της χρήσης εξελικτικών τελεστών έναντι της μη. Η υπεροχή αυτή είναι καθολική σε όλο το εύρος των πειραμάτων μας. Η PMA4 αντιμετώπισε με μεγαλη επιτυχία και το πρόβλημα αυτό. Οι μεγάλοι πληθυσμοί δημιουργούν προβλήματα καθώς κινούνται αργά προς τη βέλτιστη λύση και δημιουργούν πολλούς απογόνους σε κάθε γενιά με αποτέλεσμα να χρειάζονται πολλές επαναλήψεις. Συχνά με

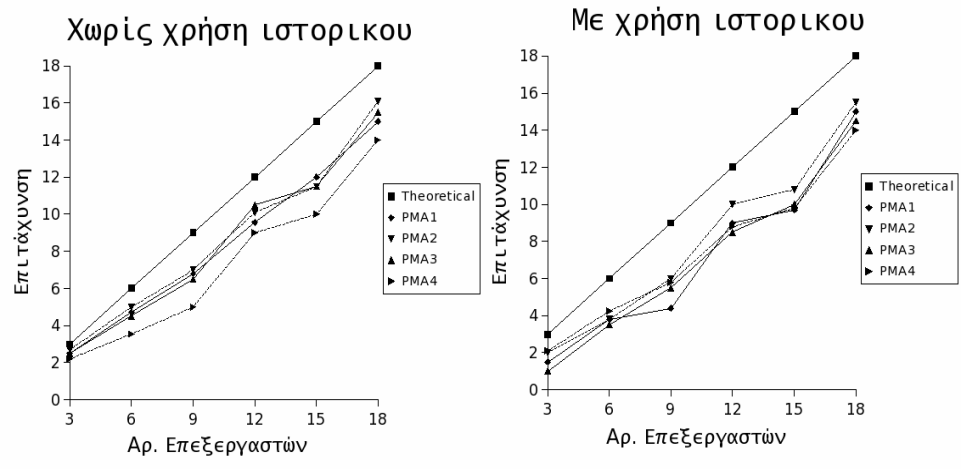


Σχήμα 6.6: Αποτελέσματα της PMA4 για 100 επαναλήψεις

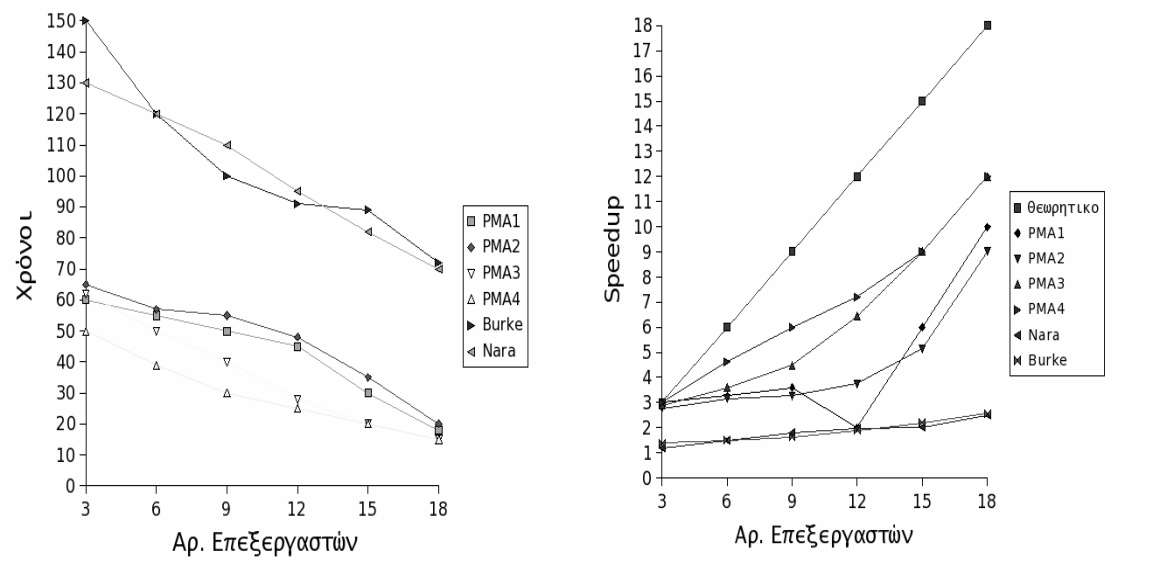
τη χρήση μεγάλου πληθυσμού ο αλγόριθμος παραβλέπει μία λύση με συνάρτηση κόστους άνω του μέσου όρου η οποια βρίσκεται σε μια στενή κορυφή του πεδίου αναζήτησης, καθώς έχει την τάση να οδηγείται σε πιο ευρείες κορυφές. Η απόδοση των μεθοδολογιών μας με μικρό μέγεθος πληθυσμού είναι καλύτερη από την απόδοση των άλλων μεγεθών πληθυσμού, αλλά συνήθως καθηλώνεται σε τοπικά σημεία. Μπορούμε να συμπεράνουμε



Σχήμα 6.7: Επίδραση της μεταβολής του ποσοστού μετανάστευσης στην ταχύτητα και στην ποιότητα των αποτελεσμάτων



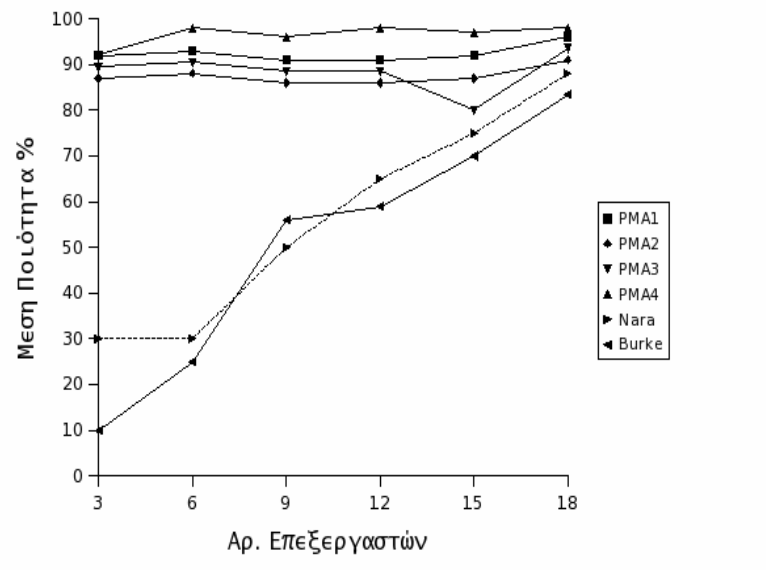
Σχήμα 6.8: Επιτάχυνση των παράλληλων υλοποιήσεων συναρτήση της χρήσης καταγραφης του ιστορικού εξέλιξης (Αριστερά : Χωρίς ιστορικό , Δεξιά : Με ιστορικό)



Σχήμα 6.9: Χρόνοι για διαφορετικό αριθμό επεξεργαστών και οι επιταχύνσεις για 6 μεθόδους

ότι οι μικρές τιμές μεγέθους υποπληθυσμών είναι καλό να χρησιμοποιούνται όταν ο στόχος είναι η ταχεία βελτιστοποίηση και όχι τόσο η εύρεση του ολικού ελάχιστου. Αντιθετα η χρήση μεγάλων τιμών υποπληθυσμών βοηθά στην ευρεση μιας πολύ καλής λύσης αλλά όχι και στην ταχεία βελτιστοποίηση. Βασική υπόθεση υπήρξε στην παρούσα εργασία ότι όλες οι εργασίες είναι της ίδιας σημαντικότητας.

Οι μιμητικοί αλγόριθμοι είναι μέρος της εξελικτικής πληροφορικής η οποία με τη σειρά της είναι ταχέως εξελισσόμενη περιοχή. Μέσω των μηχανισμών τους και περνώντας το πλήθος των λύσεων από γενιά σε γενιά, οι κακές λύσεις τείνουν να πεθάνουν ενώ οι καλές λύσεις συνδυάζονται ώστε να παράγουν ακόμα καλύτερες.



Σχήμα 6.10: Μέση ποιότητα για 6 μεθόδους

Η PMA2 είναι λιγοτερο αποτελεσματική από τις υπόλοιπες τρεις υλοποιήσεις ενώ παρατηρείται και μια απότομη πτώση στους 12 σταθμούς εργασίας στην καμπύλη επιτάχυνσης της υλοποίησης PMA3 κάτι το οποίο οφείλεται στο ότι υπάρχει μια ανισορροπία του φόρτιου εξαιτίας της τεχνικής υλοποίησης της απαγορευμένης λίστας. Ο καθορισμός των γειτονικών λύσεων απαιτεί μια διαδικασία διακριτοποίησης του εφικτού χώρου κάτι το οποίο κάνει τον αργό σταθμό εργασίας να τελειώνει πάντα τελευταίος και να υπάρχει ένας σημαντικός χρόνος αδράνειας για τους ταχύτερους σταθμούς εργασίας σε ένα ετερογενές περιβάλλον. Η δημιουργία του προγράμματος έγινε σε γλώσσα C με χρήση της βιβλιοθήκης διεπαφής περάσματος μηνυμάτων και του PARAMENOAS. Αυτό επιτεύχθηκε με την παρουσίαση ενός μοντελοποιημένου προβλήματος χρονοπρογραμματισμού όπου από μία αρχική λύση μπορέσαμε να καταλήξουμε σε μία πολύ καλύτερη με

μικρότερο χρόνο υλοποίησης των λειτουργιών. Επιπλέον, ο υπολογιστικός χρόνος που χρειάστηκε ήταν μόλις μερικά δευτερόλεπτα και για τις τέσσερις μεθοδους σε αντίθεση με άλλες μεθόδους οι οποίες για το ίδιο μέγεθος προβλήματος (60*60) χρειάζονται πολύ περισσότερο χρόνο .

Είναι μία πρότυπη εφαρμογή που μπορεί να επεκταθεί και να καλύψει τις ανάγκες μεγάλων εταιριών ηλεκτροδότησης, με σχετικά εύκολο τρόπο. Αυτό μπορεί να γίνει με την είσοδο των αρχικών μεταβλητών που απαιτούνται για τη μελέτη του προβλήματος της κάθε εταιρίας.

Επίσης ο κώδικας που έχει γραφτεί περιέχει συναρτήσεις που είναι εύκολο να κατανοηθούν από κάποιον γνώστη του αντικειμένου και να χρησιμοποιηθούν ανεξάρτητα και σε άλλες εφαρμογές. Βέβαια είναι δυνατή και η προσθήκη συναρτήσεων καθώς και στοιχείων που θα καθιστούν τον κώδικα ικανό να ικανοποιήσει μεγάλη ποικιλία προβλημάτων καθώς και αναγκών.

Ακόμα στον τομέα της έρευνας και μελέτης τέτοιων εφαρμογών μπορεί να χρησιμοποιηθεί για την περαιτέρω ανάλυση προβλημάτων χρονοπρογραμματισμού και να αποτελέσει έτσι σημείο αναφοράς για μεγαλύτερη εξέλιξή τους.

Κεφάλαιο 7

Κατασκευή Ωρολογίου Προγράμματος

Οι πρώτες εφαρμογές ΕΑ που ασχολήθηκαν με το θέμα του αυτοματοποιημένου ωρολογίου προγράμματος (automated timetabling) εμφανίστηκαν στις αρχές τις δεκαετίας του 90 (π.χ. η εργασία του Colorni [51]). Έκτοτε το πεδίο έχει αναπτυχθεί με ταχύτατους ρυθμούς, φτάνοντας στο σημείο στο διεθνές συνέδριο αυτοματοποιημένου ωρολογίου προγράμματος Practice and theory of Automated Timetabling (PATAT02) το ένα τρίτο των εργασιών που παρουσιάστηκαν να κάνουν χρήση εξελικτικών τεχνικών ενώ οι υβριδικές τεχνικές συμπεριλαμβανομένων και των ΜΑ αποτελούσαν το 47% των ΕΑ. Από το γεγονός αυτό συμπεραίνουμε ότι οι εξελικτικοί αλγόριθμοι έχουν πια καθιερωθεί ως μια επιτυχημένη μεθοδολογία για τη λύση πλήθους προβλημάτων ωρολογίου προγράμματος παράλληλα με τις συνήθεις τεχνικές της Επιχειρησιακής Έρευνας ενώ οι ΜΑ

αποτελούν μια πολύ ενδιαφέρουσα υβριδική εξελικτική τεχνική πολλά υποσχόμενη στην επίλυση τέτοιων προβλημάτων.

Το κεφάλαιο αυτό είναι διαρθρωμένο ως εξής : στην ενότητα 7.1 κάνουμε μια σύντομη επισκόπηση της έρευνας στην κατεύθυνση των εξελικτικών κύριως υλοποιήσεων αναφορικά με εφαρμογές αυτοματοποιημένου ωρολογίου προγράμματος. Στην ενότητα 7.2 γίνεται μια εκτενής περιγραφή του προβλήματος, στην ενότητα 7.3 περιγράφεται η υλοποίηση που προτάσεται στα πλαίσια της εργασίας αυτής και στην συνέχεια ακολουθούν τα αποτελέσματα και τα συμπεράσματα από τις εκτελέσεις των πειραμάτων μας. Ένα μέρος των αποτελεσμάτων αυτού του κεφαλαίου έχει υποβληθεί για δημοσίευση στο περιοδικό Journal of Operational Research.

7.1 Παρούσα κατάσταση - Είδη ωρολογίων προγραμμάτων

Η πλειονότητα των εργασιών φαίνεται να ασχολείται με προβλήματα ωρολογίου προγράμματος από τον πραγματικό κόσμο και όχι με απλοποιημένες εκδόσεις προβλημάτων ή τεχνητά εύκολα προβλήματα.

Διακρίνεται μια μεγάλη συγκέντρωση προσπαθειών προς τα πανεπιστημιακά ωρολόγια προγράμματα, όχι όμως και απροσδόκητη, αφού οι περισσότερες εργασίες αναπτύσσονται μέσα σε πανεπιστήμια [35]. Παρά το μεγάλο πλήθος υλοποιήσεων πανεπιστημιακών και εξεταστικών ωρολογίων προγραμμάτων, δεν υπάρχει διαθέσιμο εξελικτικό λογισμικό

για σχολικά ωρολόγια προγράμματα, αν και υπάρχουν αρκετές μη εξελικτικές εφαρμογές, όπως το SchoolMagic [166] ένα γιαπωνέζικο σύστημα.

Ένας άλλος τομέας ο οποίος φαίνεται να έχει παραμεληθεί είναι αυτός του ωρολογίου προγράμματος στον εργασιακό χώρο. Εργασίες που ασχολούνται με το ωρολόγιο πρόγραμμα σε εργασιακό χώρο είναι αυτές των Chiarandini, Schaerf και Tiozzo[48]. Αλλες κατηγορίες ωρολογίων προγραμμάτων που αναφέρθηκαν στα συνέδρια αλλά αγνοήθηκαν από την κοινότητα των εξελικτικών αλγορίθμων είναι τα ωρολόγια προγράμματα αθλητικών γεγονότων και νοσοκομείων.

Πολλά από τα προβλήματα που μελετήθηκαν είναι εξαιρετικά ογκώδη, ένδειξη της ικανότητας αντιμετώπισης μεγάλων προβλημάτων από τους εξελικτικούς και ειδικά από τους υβριδικούς εξελικτικούς αλγόριθμους. Για παράδειγμα, ο Zoriktuev [217] από το τμήμα αυτοματισμού του UFA Technical University της Ρωσικής Ακαδημίας αντιμετώπισε ένα πρόβλημα με 20000 φοιτητές, όπου 450 διαλέξεις και 457 μαθήματα πρέπει να ανατεθούν σε ένα ωρολόγιο πρόγραμμα 350 ωρών χρησιμοποιώντας 98 καθηγητές. Το πεδίο των προβλημάτων που εξετάζονται γίνεται πιο ρεαλιστικό. Για παράδειγμα πολλές εργασίες χρησιμοποιούν πρακτικούς περιορισμούς λαμβάνοντας υπόψη στοιχεία όπως ο χρόνος μετακίνησης μεταξύ των αιθουσών, προτιμήσεις καθηγητών και περιορισμούς αιθουσών. Ίσως οι εξελικτικές υλοποιήσεις να οδηγούνται στη χρήση τέτοιων πολλαπλών περιορισμών λόγω της ευκολίας ενσωμάτωσης των παραβιάσεων των περιορισμών αυτών μέσα σε μία και μόνο συνάρτηση αξιολόγησης.

Μια μεγάλου εύρους έρευνα σχετικά με τη φιλολογία σε πρακτικές εφαρμογές αυτοματοποιημένου ωρολογίου προγράμματος δημοσιεύτηκε από τον Cartert 1986 [133]. Η έρευνα

αυτή επεκτάθηκε το 1996 [178], για να συμπεριλάβει πρόσφατες εξελίξεις σχετικά με το εξεταστικό ωρολόγιο πρόγραμμα και ξανά το 1997 [35], για να συμπεριλάβει πρακτικές εφαρμογές σε ωρολόγια προγράμματα μαθημάτων. Και στις δύο τελευταίες ενημερώσεις υπάρχει πλήθος υλοποιήσεων με χρήση εξελικτικών αλγορίθμων, εν αντιθέσει της αρχικής έρευνας του 1986 όπου δεν υπήρχε καμία.

7.1.1 Τεχνικές υλοποίησης

Κάθε εξελικτικός αλγόριθμος απαιτεί κάποια μέθοδο αναπαράστασης του ωρολογίου προγράμματος με τη μορφή ενός χρωμοσώματος. Ένα σημαντικό μέρος των εργασιών χρησιμοποιεί κάποιο είδος άμεσης (direct) αναπαράστασης όπου το χρωμόσωμα αναπαριστάνει άμεσα το ωρολόγιο πρόγραμμα αυτό καθεαυτό, συμπεριλαμβάνοντας όλες τις αναθέσεις αιθουσών, καθηγητών και μαθημάτων για κάθε χρονική περίοδο. Η μέθοδος αυτή καθιστά την αναπαράσταση αρκετά εξειδικευμένη για το εκάστοτε πρόβλημα. Στην έμμεση (indirect) αναπαράσταση, το χρωμόσωμα συνήθως αναπαριστάνει μια διατεταγμένη λίστα από παραδόσεις μαθημάτων, οι οποίες τοποθετούνται σε ένα ωρολόγιο πρόγραμμα βάσει κάποιας προκαθορισμένης διαδικασίας. Η διαδικασία αυτή δύναται να χρησιμοποιεί οποιονδήποτε συνδυασμό ευρετικής και τοπικής αναζήτησης για την τοποθέτηση των παραδόσεων.

Έχει αποδειχθεί ότι χρησιμοποιώντας διαφορετικές μεθόδους αναπαράστασης για το ίδιο πρόβλημα, οδηγούμαστε σε διαφορετικά αποτελέσματα [36]. Η σχέση που έχει η χρησιμοποιούμενη αναπαράσταση με την αποτελεσματικότητα του αλγορίθμου χρήζει περαιτέρω

μελέτης.

Σε κάθε άμεση αναπαράσταση, ο γόνος που προκύπτει από την εφαρμογή ενός γενετικού τελεστή, όπως η μετάλλαξη ή ο ανασυνδυασμός, είναι πιθανόν να είναι ακατάλληλος λόγω του ότι μπορεί να παραβιάζει κάποιους σημαντικούς αυστηρούς περιορισμούς (hard constraints) . Έχουν επικρατήσει δύο κυρίως τρόποι αντιμετώπισης τέτοιων περιπτώσεων. Βάσει του πρώτου, ο γόνος διορθώνεται πριν τοποθετηθεί στο νέο πληθυσμό. Η εναλλακτική λύση είναι να επιτρέπονται τέτοιοι γόνοι, αλλά συγχρόνως να τους επιβάλλεται μια πολύ βαριά ποινή μέσω μιας σταθμισμένης συνάρτησης επιβολής ποινής (weighted penalty function).

Ανεξαρτήτως της αναπαράστασης, ο αλγόριθμος απαιτεί κάποια μέθοδο διαχείρισης των περιορισμών του ωρολογίου προγράμματος. Σε πραγματικά προβλήματα, οι περιορισμοί ποικίλουν σε σημαντικότητα και πλήθος. Οι περισσότερες εργασίες, χρησιμοποιούν μια συνάρτηση αξιολόγησης βασισμένη σε ποινές, προκειμένου να μετρήσουν την ποιότητα ενός ωρολογίου προγράμματος. Συνήθεις τρόποι υλοποίησης μιας τέτοιας συνάρτησης είναι:

- Χρήση μιας σταθμισμένης συνάρτησης επιβολής ποινής, όπου το βάρος αντικατοπτρίζει τη σπουδαιότητα του κάθε περιορισμού. Αυτή μπορεί να είναι ένας γραμμικός συνδυασμός των βαρών των παραβιάσεων, αλλά μπορεί να εμπεριέχει και κάποιους εκθετικούς όρους, αν και οι Burke και Newall [37] προειδοποιούν ότι οι τιμές που χρησιμοποιούνται σε εκθετικές συναρτήσεις πρέπει να επιλεγούν με μεγάλη προσοχή.

- Χρήση μιας συνάρτησης δύο βημάτων, όπου οι λιγότερο σημαντικοί χαλαροί περιορισμοί

(soft constraints) υπολογίζονται μόνο στην περίπτωση που ικανοποιούνται οι αυστηρότεροι.

Πολλοί ερευνητές τονίζουν τη δυσκολία που υπάρχει στην επιλογή των κατάλληλων βαρών, καθώς και το μεγάλο βαθμό στον οποίο αυτά επηρεάζουν την ποιότητα της λύσης. Ο Paechter με τους συνεργάτες του [168] ξεπέρασε αυτή τη δυσκολία με το να επιτρέπει στα βάρη να α λλάζουν δυναμικά από το χρήστη κατά την εκτέλεση του αλγορίθμου. Σχετικά μ΄ αυτήν τη λύση έχει σχολιασθεί το γεγονός ότι αρκετές φορές είναι δύσκολο για το χρήστη να αποφασίσει ποια στοιχεία συνθέτουν ένα καλό ωρολόγιο πρόγραμμα και ποιοι περιορισμοί είναι σημαντικότεροι από τους άλλους. Συχνά οι απαιτήσεις ενός ωρολογίου προγράμματος εκφράζονται με προτάσεις όπως «Πρέπει να υπάρχουν λιγότερες παραβιάσεις του περιορισμού Α από τον περιορισμό Β», αντί για «Ο περιορισμός Α είναι δύο φορές σημαντικότερος από τον Β», πράγμα που καθιστά δύσκολη την ανάθεση κατάλληλων βαρών. Σ΄ αυτή τη περίπτωση θα βοηθούσε η ύπαρξη μιας αναφοράς των παραβιάσεων που γίνονται σε συνδυασμό με τον καθορισμό των βαρών από τον χρήστη, αν και θα πρέπει ο αλγόριθμος να είναι αρκετά γρήγορος, ώστε μια τέτοια λύση να είναι πρακτικά εφικτή.

Πολλές από τις παραπάνω εργασίες χρησιμοποιούν υβριδικές γενετικές τεχνικές, συμπεριλαμβάνοντας κάποια στοιχεία τοπικής αναζήτησης. Αυτό μπορεί να πάρει τη μορφή ενός έξυπνου τελεστή μετάλλαξης, ή μιμητικής αναζήτησης. Πολλοί εξελικτικοί αλγόριθμοι αντιμετωπίζουν προβλήματα σύντομης σύγκλισης του πληθυσμού, η οποία μπορεί να οδηγήσει σε μη βέλτιστες λύσεις (σύγκλιση σε τοπικό βέλτιστο). Μια ενδιαφέρουσα

μέθοδος που προτάθηκε από τον Burke και τους συνεργάτες τους,[34] χρησιμοποιεί προσωμοιούμενη ανόπτηση για να ρυθμίσει τα βάρη των περιορισμών που παραβιάζονται κάθε φορά που έχουμε σύγκλιση του πληθυσμού, έως ότου βρεθεί μια ικανοποιητική λύση.

Ένα άλλο πρόβλημα, είναι αυτό της επιλογής ανάμεσα σε ομοιογενή ή ετερογενή συστήματα πειραματικών συστοιχιών πάνω στις οποίες θα υλοποηθούν διάφοροι αλγόριθμοι για την επίλυση του προβλήματος. Πρόκειται για μια εμπειρική διαδικασία δοκιμής και λάθους [137]. Ποικίλες είναι οι εφαρμογές που αναπτύχθηκαν για τη δημιουργία αυτοματοποιημένων ωρολογίων προγραμμάτων, και οι οποίες βασίζονται σε εξελικτικούς αλγορίθμους. Ενδεικτικά αναφέρονται οι πιο γνωστές.

Μια πολύ καλή εφαρμογή για πανεπιστημιακά ωρολόγια προγράμματα είναι η (Automated Timetabler), η οποία αναπτύχθηκε στην Ιαπωνία[112]. Ο μηχανισμός που χρησιμοποιεί η εφαρμογή ενσωματώνει Hill Climbing, έναν εξελικτικό αλγόριθμο και έναν αλγόριθμο αναζήτησης. Είναι ικανή να επιλύσει όλους τους αυστηρούς περιορισμούς και να βελτιστοποιήσει τους χαλαρούς χρησιμοποιώντας την τεχνική των δύο βημάτων. Παράγει ωρολόγιο πρόγραμμα μαθημάτων, αιθουσών και καθηγητών. Έχει δοκιμασθεί με επιτυχία σε δεκα πανεπιστήμια της Ιαπωνίας και μπορεί να χειρισθεί ογκώδη προβλήματα. Στο μεγαλύτερο πανεπιστήμιο που δοκιμάστηκε, το πρόβλημα είχε να κάνει με εβδομήντα τμήματα, επτακόσιους καθηγητές και τέσσερις χιλιάδες παραδόσεις. Περιείχε πέντε αυστηρούς περιορισμούς και δέκα χαλαρούς. Χρησιμοποιώντας έναν επεξεργαστή Pentium 450MHZ, η λύση μπορεί να βρεθεί μέσα σε διακοσιαείκοσι λεπτά, πρόκειται δηλαδή για μια εξαιρετικά γρήγορη εφαρμογή.

Πολλά άλλα είναι τα πανεπιστήμια που χρησιμοποιούν παρόμοιες εφαρμογές αυτοματοποιημένων ωρολογίων προγραμμάτων, για να καλύψουν τις ανάγκες τους. Για παράδειγμα, το Napier University του Εδιμβούργου έχει αναπτύξει την εφαρμογή Neeps and Tatties. Πρόκειται για μια τελείως διαφορετική προσέγγιση του θέματος απ΄ αυτή του AOT. Χρησιμοποιείται με επιτυχία από το πανεπιστήμιο για τη δημιουργία των ωρολογίων προγραμμάτων του. Στα πλεονεκτήματα του προγράμματος συγκαταλέγεται η δυνατότητα τροφοδότησης του με προηγούμενες λύσεις, επιτρέποντας έτσι τη βαθμιαία εξέλιξη των λύσεων, καθώς και η δυνατότητα επεξεργασίας λύσεων προηγουμένων ετών για την επιτάχυνση του αλγορίθμου. Καινοτομία της εφαρμογής είναι η δυνατότητα αλλαγής των βαρών από το χρήστη κατά την εκτέλεση του προγράμματος.

Άλλες ενδιαφέρουσες υλοποίησεις όχι εξελικτικές αλλά παράλληλες είναι και αυτές των Aydin et all [13] και των Hermosilla et all [104]. Οι πρώτοι έχουν παραλληλοποιήσει την μέθοδο της προσομοιούμενης ανόπτησης και οι δεύτεροι της αποτρεπτικής αναζήτησης. Τα αποτελέσματα που έδωσε η μεθοδολογία τους για το πρόβλημα μας τα συγκρίνουμε στο τέλος του κεφαλαίου με αυτά των δικών μας υλοποιήσεων.

7.2 Περιγραφή του προβλήματος

Το πρόβλημα που κλήθηκε να αντιμετωπίσει ο παράλληλος μιμητικός αλγόριθμος της παρούσης εργασίας, είναι η δημιουργία του ωρολογίου προγράμματος μαθημάτων. Πρόκειται για την ανάθεση ενός δεδομένου συνόλου μαθημάτων, καθηγητών και αιθουσών σε συγκεκριμένες χρονικές περιόδους της εβδομάδας. Η ανάθεση πρέπει να γίνει με τέτοιο

τρόπο, ώστε να πληρούνται ορισμένες προκαθορισμένες συνθήκες (περιορισμοί) προκειμένου αυτό να έχει πρακτικό ενδιαφέρον και χρησιμότητα. Από τη φύση του είναι ένα αρκετά δύσκολο πρόβλημα με μεγάλο και πολύπλοκο χώρο αναζήτησης και επιπλέον με αρκετές ιδιαιτερότητες σε σχέση με άλλα πανεπιστημιακά ωρολόγια προγράμματα.

7.2.1 Παράμετροι του προβλήματος

Μερικά από τα χαρακτηριστικά-ιδιαιτερότητες του συγκεκριμένου προβλήματος είναι:

- Ο αυξημένος αριθμός των εργαστηριακών τμημάτων.
- Η συχνά αναπόφευκτη επικάλυψη μαθημάτων του ίδιου εξαμήνου.
- Υπάρχει περιορισμένος αριθμός εργαστηριακών αιθουσών.
- Πολλές φορές το ίδιο μάθημα διδάσκεται περισσότερες από μία φορές την εβδομάδα (αυτές οι πολλαπλές παραδόσεις από εδώ και στο εξής θα καλούνται ταίρια).

7.2.2 Περιορισμοί

Οι περιορισμοί που πρέπει να πληρούνται από το ωρολόγιο πρόγραμμα μπορούν να χωριστούν σε δύο κατηγορίες. Η πρώτη περιλαμβάνει τους λεγόμενους αυστηρούς περιορισμούς οι οποίοι και πρέπει να πληρούνται εις το ακέραιο (π.χ. ένας καθηγητής δεν μπορεί την ίδια χρονική στιγμή να διδάσκει περισσότερα του ενός μαθήματα). Στη δεύτερη έχουμε τους χαλαρούς περιορισμούς οι οποίοι δεν έχουν τόσο σχέση με την ακεραιότητα του ωρολογίου προγράμματος, αλλά εξυπηρετούν πρακτικές απαιτήσεις (π.χ. η επιθυμία ενός

καθηγητή να εργάζεται μόνο κάποιες συγκεκριμένες ώρες της εβδομάδας). Το πλήθος των χαλαρών περιορισμών είναι ανεξάντλητο και άμεσα εξαρτώμενο από το διαχειριστή του ωρολογίου προγράμματος. Οι περιορισμοί που χρησιμοποιήθηκαν στην παρούσα εργασία είναι οι εξής:

1. Σε κάθε εργαστηριακή παράδοση οι δύο διδάσκοντες πρέπει να είναι διαφορετικοί.
2. Οι παραδόσεις που αποτελούν 'ταίρια' πρέπει να γίνονται από τους ίδιους καθηγητές.
3. Οι παραδόσεις 'ταίρια' πρέπει να γίνονται σε διαφορετικές ημέρες.
4. Ο ημερήσιος αριθμός ωρών διδασκαλίας ενός καθηγητή δεν πρέπει να βγαίνει εκτός των προκαθορισμένων ορίων του.
5. Ο εβδομαδιαίος αριθμός ωρών διδασκαλίας ενός καθηγητή δεν πρέπει να βγαίνει εκτός των προκαθορισμένων ορίων του.
6. Κάθε αίθουσα μπορεί να φιλοξενεί μία μόνο παράδοση ανά χρονική περίοδο.
7. Δεν πρέπει να υπάρχουν επικαλύψεις μεταξύ μαθημάτων.
8. Ένας καθηγητής μπορεί να διδάσκει ένα μόνο μάθημα ανά χρονική περίοδο.
9. Δεν πρέπει να υπάρχουν κενές ώρες στο πρόγραμμα ενός καθηγητή.
10. Δεν πρέπει να υπάρχουν κενές ώρες στο πρόγραμμα.
11. Πρέπει να ληφθούν υπόψη οι προτιμήσεις ενός καθηγητή για το ποιες ώρες θέλει ή δεν θέλει να εργάζεται.

Στο σημείο αυτό θα πρέπει να τονίσουμε ότι ορισμένοι περιορισμοί εκ των πραγμάτων δεν είναι δυνατό να πληρούνται εις το ακέραιο. Για παράδειγμα, το πλήθος των παραδόσεων

είναι τόσο μεγάλο που είναι αδύνατο να μην υπάρχουν επικαλύψεις.

7.3 Υλοποίηση

7.3.1 Γενικά

Η υλοποίηση του παράλληλου μιμητικού αλγορίθμου για τη δημιουργία του ωρολογίου προγράμματος, βασίστηκε στις μεθοδολογίες PMA1 - PMA4 (βλ. κεφαλαιο 5). Τα δεδομένα μας είναι τα ίδια με αυτά που χρησιμοποίησαν οι Aydin et al [104], Di Gaspero και Schaerf [189] και Hermosilla et al [13] και είναι διαθέσιμα στην διεύθυνση http:// www.diegm.uniud.it/ schaerf/ projects/ coursett/ Κύριο μέλημα της εφαρμογής ήταν η βελτίωση της απόδοσης των αλγορίθμων (προσθήκη νέων τεχνικών, εμπλοκή του ανθρώπινου παράγοντα στη διαδικασία της εξέλιξης).

Η εφαρμογή χωρίζεται σε τρία βασικά μέρη:

- Ο παράλληλος μιμητικός αλγόριθμος περιλαμβάνει όλες τις απαραίτητες διαδικασίες για την πραγματοποίηση της εξέλιξης σε μια συστοιχία υπολογιστων και επιπλέον καταγράφει και ορισμένα στατιστικά στοιχεία.

- Η Βάση δεδομένων. Περιέχει όλη την απαραίτητη πληροφορία για τη δημιουργία των ατόμων του πληθυσμού (καθηγητές, μαθήματα, αίθουσες και σχέσεις μεταξύ αυτών)

Καθόρισε k αρχικές νησίδες επεξεργαστών $P^0, ..., P^{k-1}$ και διένειμε ένα αρχιπέλαγος σε k αρχικές νησίδες $pop^0, ..., pop^{k-1}$
Γενεά = 0
Για κάθε νησίδα επεξεργαστών P^i διαίρεσε την νησίδα pop^i σε g υποπληθυσμούς $subpop^j, ..., subpop^{g-1}$
Γενεά = Γενεά + 1
Εφάρμοσε μια από τις μεθόδους (βλέπε 5.3.2, 5.3.3, 5.3.4) σε κάθε pop^i.
Αν έχει σχηματιστεί ο απαραίτητος αριθμός καλών λύσεων της νησίδας pop^i αυτές μεταναστεύουν στο $pop^{(i+1)modk}$
Αν τουλάχιστον μία λύση παρελήφθη από το $pop^{(i-1)modk}$
τότε η λυση που παραλήφθηκε τοποθετείται στο pop^i
Ελέγχεται αν κάποια νησίδα ικανοποιεί τη συνθήκη τερματισμού και αν όχι τότε επιστρέφουμε στο βήμα 3.

Σχήμα 7.1: Μοντέλο Νησίδων Μιμητικών Αλγορίθμων σε Νησίδες επεξεργαστών

Ο αλγόριθμος που παρατίθεται περιγράφει τον καθορισμό των νησίδων στην υλοποίηση μας και τον καθορισμό της μεθόδου εξέλιξης των νησίδων. Στο παράρτημα Α.2 παρατίθεται ο κώδικας MPI για το ωρολόγιο πρόγραμμα. Στις επομένες παραγράφους θα μελετήσουμε τον τρόπο με τον οποίο γίνεται η αναπαράσταση του εβδομαδιαιου ωρολογίου προγράμματος, την συνάρτηση αξιολόγησης, την βάση δεδομένων που χρησιμοποιουμε και τον τρόπο που γίνεται η παρακολούθηση της εξέλιξης.

7.3.2 Αναπαράσταση

Κάθε άτομο του πληθυσμού αναπαριστάνει και ένα ολοκληρωμένο εβδομαδιαίο ωρολόγιο πρόγραμμα. Η αναπαράσταση είναι άμεση και έχει τη μορφή ενός δισδιάστατου πίνακα 7.1, όπου η κάθε στήλη αντιστοιχεί σε μία παράδοση (και όχι μάθημα γιατί, όπως έχει

	1	2	3	...	Max
Αίθουσα	4	2			
Καθηγητής 1	1	5			
Καθηγητής 2	4	NULL			
Ημέρα	2	2	3		
Ωρα Έναρξης	4	2			

Πίνακας 7.1: Σχηματική αναπαράσταση ωρολογίου προγράμματος

προαναφερθεί, ένα μάθημα μπορεί να έχει πολλές παραδόσεις), ενώ η κάθε μία από τις γραμμές του κωδικοποιεί ένα χαρακτηριστικό της παράδοσης. Αυτά κατά σειρά είναι: η αίθουσα όπου θα γίνει η παράδοση, ο καθηγητής 1, ο καθηγητής 2 (εάν πρόκειται για εργαστηριακό μάθημα), η ημέρα της εβδομάδας και η ώρα έναρξης. Οι τιμές του κάθε κελιού είναι ακέραιες.

Για παράδειγμα, σύμφωνα με το σχήμα 7.1, η παράδοση με τον κωδικό 1, θα γίνει στην αίθουσα με κωδικό 4, από τους καθηγητές με κωδικό 1 και 4, την Δευτέρα στις 11 το πρωί. Στη δεύτερη παράδοση παρατηρούμε ότι ο καθηγητής 2 δεν έχει τιμή (σύμβολο NULL).

Σε κάθε κελί αντιστοιχεί ένα σύνολο επιτρεπτών τιμών. Οι τιμές του συνόλου είτε είναι προκαθορισμένες (π.χ. επιτρεπτές τιμές του χαρακτηριστικού ημέρα είναι από 2 έως 5), είτε απορρέουν από τη βάση δεδομένων, άμεσα (π.χ. ο χρήστης ορίζει το ποιοι καθηγητές μπορούν να διδάξουν ένα μάθημα) ή έμμεσα (ο χρήστης ορίζει τη διάρκεια μιας παράδοσης συνεπώς το επιτρεπτό όριο για την ώρα έναρξης αυτής είναι από 1 έως 11 μείον τη διάρκεια).

Αν θελήσουμε να περιγράψουμε ένα τέτοιο άτομο με ορολογία της γενετικής μπορούμε να πούμε ότι: Ολόκληρος ο πίνακας είναι ένα χρωμόσωμα και η κάθε στήλη του ένα γονίδιο. Κάθε γονίδιο έχει πέντε τιμές χαρακτηριστικών και ο φαινότυπος του χρωμοσώματος είναι το ωρολόγιο πρόγραμμα.

7.3.3 Συνάρτηση αξιολόγησης

Η συνάρτηση αξιολόγησης είναι υπεύθυνη για τον υπολογισμό της ποιότητας ενός ατόμου δηλαδή ενός ωρολογίου προγράμματος. Η ποιότητα ενός ατόμου είναι αντιστρόφως ανάλογη του αριθμού των περιορισμών που παραβιάζονται. Σε κάθε παραβίαση αντιστοιχεί ένα βάρος ανάλογα με το πόσο σημαντικός ή όχι είναι ο περιορισμός που παραβιάζεται. Η ποινή που επιβάλλεται σε ένα άτομο για ένα συγκεκριμένο περιορισμό είναι ίση με το γινόμενο του βάρους του περιορισμού αυτού επί το σύνολο των παραβιάσεων.

Έστω C το σύνολο των περιορισμών, P_i η ποινή που σχετίζεται με τον περιορισμό $i, u_i(t) = 1$ εαν το ωρολόγιο πρόγραμμα t παραβιάζει τον περιορισμό t, διαφορετικά 0. Τότε η συνάρτηση αξιολόγησης δίνεται από τη σχέση:

$$f(t) = \frac{1}{1 + \sum i \in C P_i u_i(t)}$$

Υποθέτοντας ότι όλες οι ποινές είναι θετικές, η συνάρτηση ισούται με 1 όταν και μόνο όταν δεν υπάρχει καμία παραβίαση, διαφορετικά είναι μικρότερη του ένα.

Ένα σύνηθες πρόβλημα στους εξελικτικούς αλγορίθμους είναι η γρήγορη σύγκλιση του

πληθυσμού. Δηλαδή ο αλγόριθμος έχει συναντήσει ένα τοπικό βέλτιστο και δεν μπορεί να ξεφύγει απ΄ αυτό. Για την υπερπήδηση τέτοιων προβλημάτων, ο αλγόριθμος μετά την πάροδο ενός συγκεκριμένου αριθμού γενιών (γενιές κατά τις οποίες η ποιότητα της καλύτερης λύσης παραμένει στάσιμη και δεν βελτιώνεται περαιτέρω), επαναρχικοποιεί ένα μέρους του πληθυσμού ευελπιστώντας ότι θα ξεπεράσει το τοπικό βέλτιστο. Πρόκειται για μια ενέργεια απελπισίας κατά την οποία ένας αριθμός ατόμων του πληθυσμού αρχικοποιείται με τυχαίες τιμές, χάνοντας οποιαδήποτε γνώση είχε αποκτήσει μέχρι τώρα. Η τυχαία αυτή μεταβολή εμπεριέχει μια μικρή πιθανότητα να προκύψουν άτομα με καλύτερη ποιότητα από πριν και να ξεφύγουμε έτσι από το τοπικό βέλτιστο.

Τα άτομα που θα τροποποιηθούν μπορούν να επιλεχθούν με δύο τρόπους: Είτε μπορεί να είναι ένα τυχαίο δείγμα του πληθυσμού, είτε μια πιο συντηρητική προσέγγιση όπου τροποποιούνται όλα τα άτομα εκτός από τα ν καλύτερα. Σε κάθε περίπτωση το άτομο με την καλύτερη ποιότητα δεν συμμετέχει στην επαναρχικοποίηση και παραμένει στον πληθυσμό, εξασφαλίζοντας ότι δεν πρόκειται να οδηγηθούμε σε χειρότερες λύσεις σε περίπτωση αποτυχίας της επαναρχικοποίησης. Κάθε παράμετρος της μεθόδου (το πότε θα συμβεί, το πλήθος των ατόμων που θα τροποποιηθεί και το ποια δομή γειτονιάς πληθυσμού θα χρησιμοποιηθεί κατά την επιλογή) ορίζεται δυναμικά από τον χρήστη.

Λόγω του τυχαίου τρόπου δράσης η μέθοδος της επαναρχικοποίησης δεν εγγυάται πάντα την επίλυση του προβλήματος των τοπικών βέλτιστων, αν και από πρόχειρες δοκιμές που έγιναν φαίνεται η εφαρμογή της να είναι αποτελεσματική.

Σχήμα 7.2: Μη κανονικοποιημένο διάγραμμα

7.3.4 Βάση Δεδομένων

Η βάση δεδομένων μπορεί να περιγραφεί με το ER διάγραμμα του σχήματος 7.2 Ο όρος βάση δεδομένων χρησιμοποιείται κατά παράβαση αφού η εφαρμογή δε χρησιμοποιεί κάποιο DBMS αλλά απλώς έχει γίνει μια πολύ μικρή προσομοίωση αυτού. Η βάση περιέχει όλη την απαραίτητη πληροφορία για τη δημιουργία του ωρολογίου προγράμματος, καθώς και αυτή που είναι απαραίτητη για τον έλεγχο κάποιων περιορισμών (π.χ. προτιμήσεις καθηγητών). Πριν την έναρξη της εξέλιξης η βάση υφίσταται μια επεξεργασία προκειμένου να δημιουργηθεί η μήτρα με τις παραδόσεις και με τα σύνολα των αποδεκτών τιμών του κάθε κελιού της.

7.3.5 Παρακολούθηση της εξέλιξης

Η σπουδαιότητα ύπαρξης στατιστικών στοιχείων για την πορεία που ακολουθεί ο αλγόριθμος προς την εύρεση της ποθητής λύσης είναι αυτονόητη. Βοηθάει τόσο στην εξαγωγή χρήσιμων συμπερασμάτων, όσο και στον έλεγχο των μεθόδων που χρησιμοποιούνται. Παρόλο που το πλήθος και το είδος των στατιστικών στοιχείων ποικίλει ανάλογα με τις ανάγκες του κάθε χρήστη, έγινε προσπάθεια μιας όσο το δυνατόν πληρέστερης καταγραφής και ένταξης τους στην εφαρμογή. Το πρόγραμμα παρέχει στο χρήστη ένα πλήθος στατιστικών στοιχείων για την κατάσταση της τρέχουσας γενιάς, αλλά και για την πορεία της εξέλιξης γενικότερα. Η καταγραφή των στοιχείων αυτών γίνεται από την ίδια τη διαδικασία εξέλιξης, γεγονός που επιτρέπει την αξιοποίηση τους ανάλογα με τις ανάγκες του κάθε χρήστη.

Για κάθε γενιά παρακολουθούνται τα εξής στοιχεία:

- Η καλύτερη και η χειρότερη λύση (ως οντότητες).
- Η μέση τιμή της ποιότητας του πληθυσμού.
- Το ποσοστό του πληθυσμού που συγκλίνει προς την τρέχουσα καλύτερη λύση.
- Το πλήθος των ανασυνδυασμών και των μεταλλάξεων που έχουν γίνει.

Για τη συνολική εξέλιξη καταγράφονται τα ακόλουθα:

- Η χρονική διάρκεια της εξέλιξης.
- Η καλύτερη λύση που έχει βρεθεί από την αρχή της εξέλιξης (είναι χρήσιμη στην περίπτωση που ο αλγόριθμος δεν είναι ελιτιστικός).

- Η γενιά στην οποία βρέθηκε η λύση αυτή.
- Το πλήθος των στάσιμων γενιών (γενιές κατά τις οποίες η καλύτερη λύση παραμένει στάσιμη).

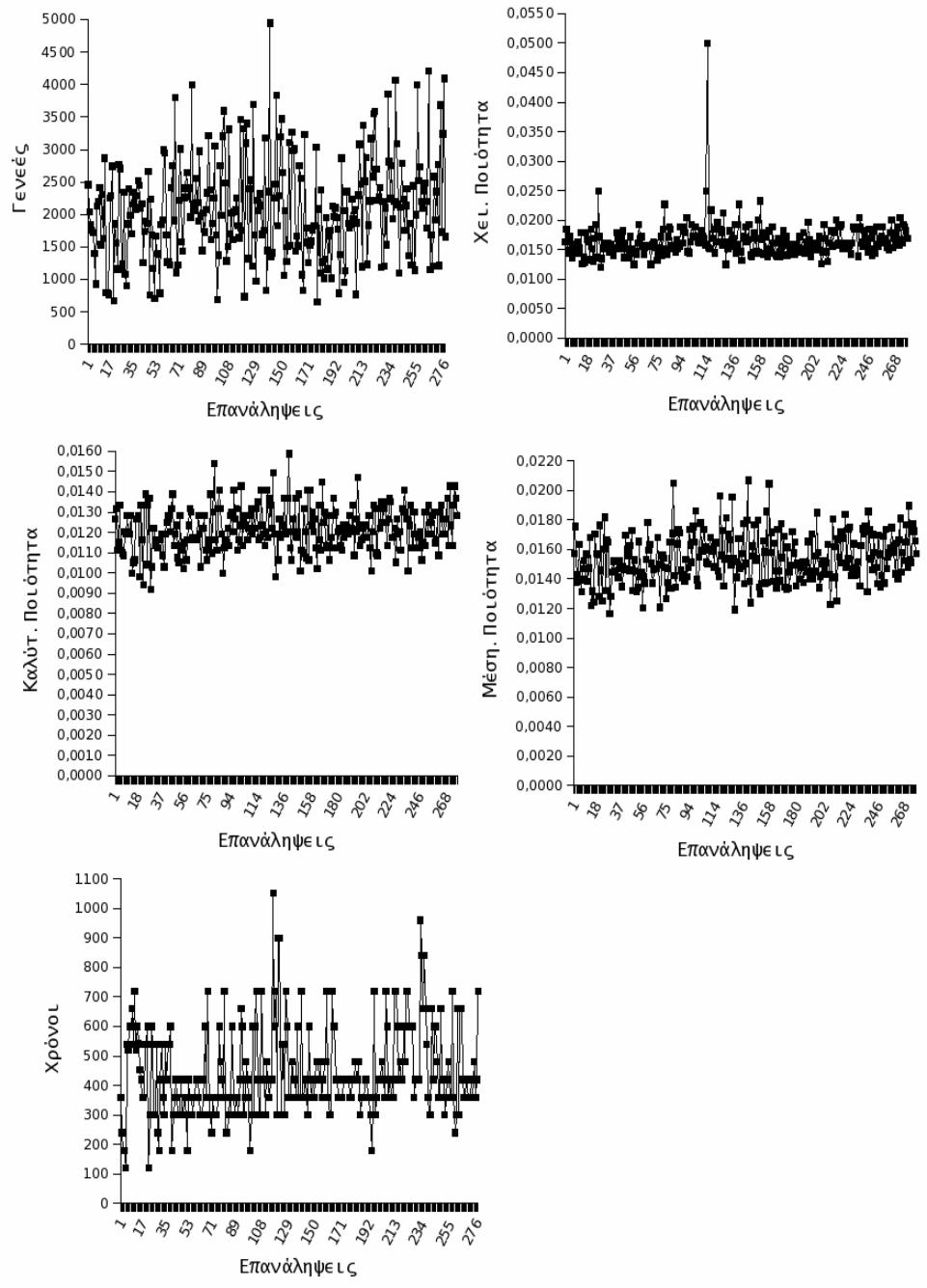
Ανεξάρτητα είναι τα στοιχεία που θα μπορούσαν να προστεθούν στην εφαρμογή, για τη βελτίωση της. Στοιχεία που έχουν να κάνουν τόσο με τον αλγόριθμο, όσο και με το περιβάλλον αυτής. Θα μπορούσε για παράδειγμα να ληφθεί υπόψη ο αριθμός των σπουδαστών που παρακολουθεί ένα μάθημα, ώστε να αποφεύγεται η επικάλυψη μαθημάτων με μεγάλη ζήτηση ή να λαμβάνεται υπόψη ο χρόνος μετακίνησης μεταξύ των αιθουσών. Η παρούσα εργασία παραβλέπει διάφορα γεγονότα, όπως το ότι μπορεί να υπάρχουν ώρες της εβδομάδας που να μη γίνονται μαθήματα.

7.4 Αποτελέσματα

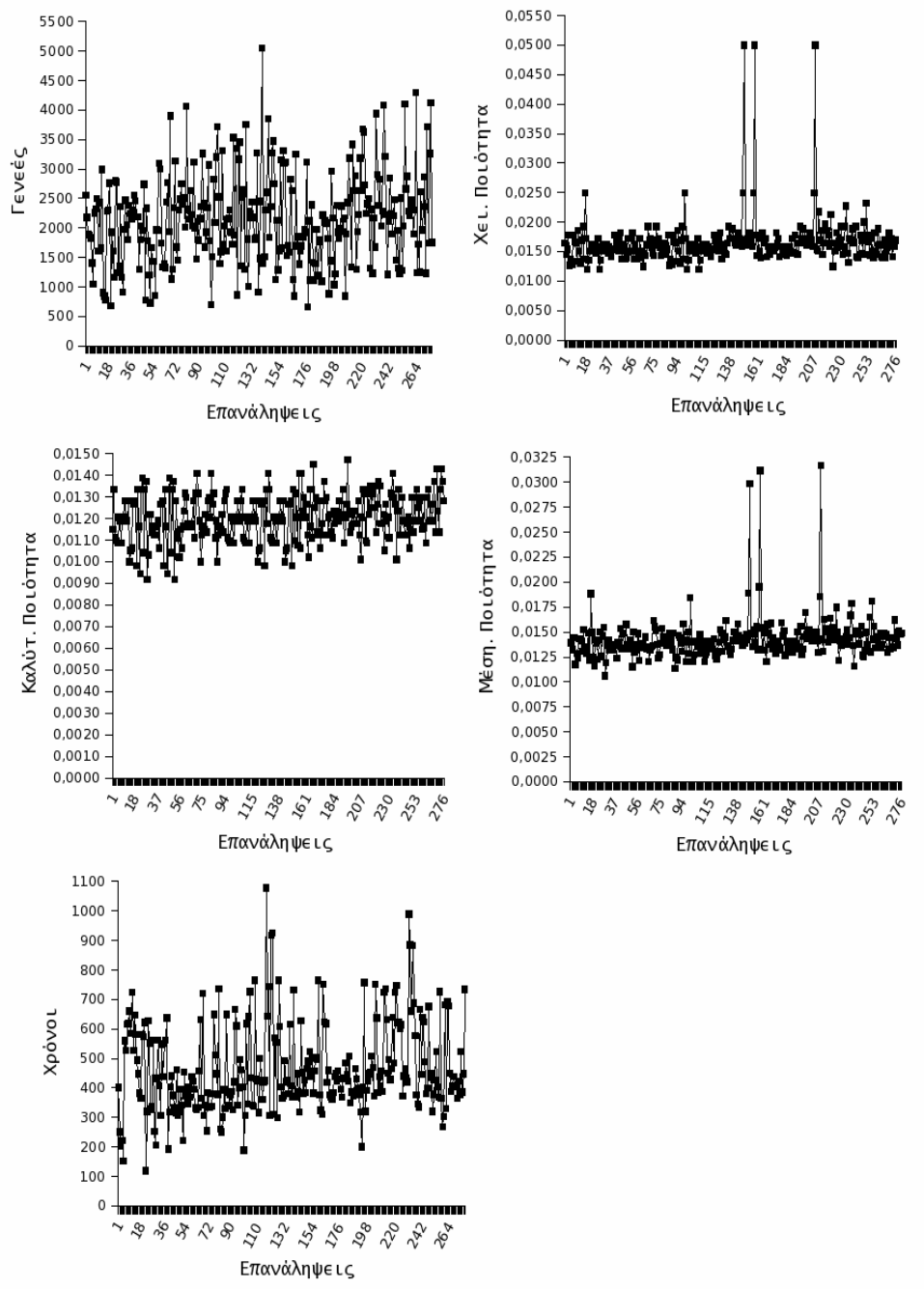
Όλα τα πειράματα έγιναν σε κοινή βάση δεδομένων όπως αυτά διαμορφώθηκαν από τις απαιτήσεις ενός εξαμήνου. Τα δεδομένα μας είναι τα ίδια με αυτά που χρησιμοποίησαν οι Aydin et al [104] και Hermosilla et al [13].Στην ενότητα αυτή παρατίθενται τα αποτελέσματα των πειραμάτων με την μορφή τεσσάρων γραφικών παραστάσεων για κάθε υλοποίηση. Στα σχήματα 7.3, 7.4, 7.5 και 7.6 δίνονται η καλύτερη, η χειρότερη και η μέση ποιότητα που βρέθηκε μέσα από ένα σύνολο διακοσίωνογδόντα πειραμάτων. Έχουμε χρησιμοποιήσει για δομή γειτονιάς του πληθυσμού δομή δακτυλίου. Επίσης σε διαγράμματα των σχημάτων 7.3, 7.4, 7.5 και 7.6 παρουσιάζονται και οι χρόνοι από το κάθε σύνολο

πειραμάτων καθώς και το πλήθος των γενιών που εξελίχθησαν συνολικά σε κάθε πείραμα. Τα αποτελέσματα αυτών των σχημάτων αφορούν εκτελέσεις που εγιναν και στους 18 επεξεργαστές της συστοιχίας. Στα σχήματα 7.13 και 7.14 έχουμε συγκριτικά αποτελέσματα για το ίδιο πρόβλημα αλλών δύο υλοποιήσεων (παράλληλης προσομοιούμενης ανόπτησης και παράλληλης αποτρεπτικής αναζήτησης) με τις δικές μας. Οι υλοποίησεις αυτές εγιναν από τους Aydin et al [104] και Hermosilla et al [13] Στα αποτελέσματα (βλ. σχήμα 7.8) παρατηρούμε μια απότομη πτώση των καμπυλών επιτάχυνσης που οφείλεται στην ετερογένεια του περιβάλλοντος μας. Στο σχήμα 7.8 παρατηρούμε ότι η PMA2 είναι λιγοτερο αποτελεσματική από τις υπόλοιπες τρεις υλοποιήσεις ενώ παρατηρείται και μια απότομη πτώση στους 12 σταθμούς εργασίας στην καμπύλη επιτάχυνσης της υλοποίησης PMA3 κάτι το οποίο οφείλεται στο ότι υπάρχει μια ανισορροπία του φόρτιου εξαιτίας της τεχνικής υλοποίησης της απαγορευμένης λίστας. Ο καθορισμός των γειτονικών λυσεων απαιτεί μια διαδικασία διακριτοποίησης του εφικτού χώρου κάτι το οποίο κάνει τον αργό σταθμό εργασίας να τελειώνει πάντα τελευταίος και να υπάρχει ένας σημαντικός χρόνος αδράνειας για τους ταχύτερους σταθμούς εργασίας σε ένα ετερογενές περιβάλλον.

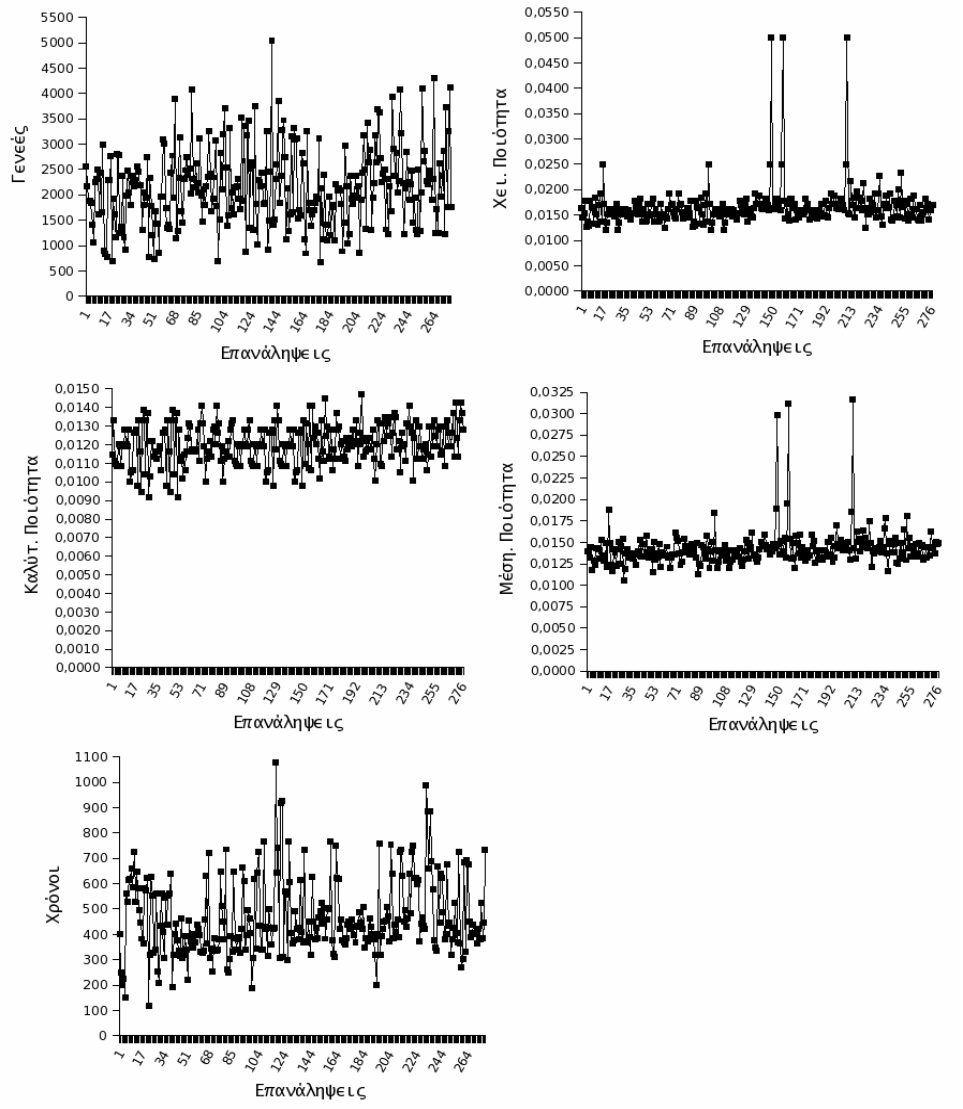
Στα διαγράμματα του σχήματος 7.8 παρουσιάζουμε τις επιταχύνσεις σε σχέση με τη καταγραφή του ιστορικού εξέλιξης χρησιμοποιώντας της μεθόδους PMA1, PMA2, PMA3, PMA4. Παρατηρούμε ότι υπάρχει μια σημαντική επίδραση στην απόδοση των υλοποίησεων μας. Συγκεκριμένα βλέπουμε ότι με την καταγραφή του ιστορικού εξέλιξης ο χρόνος επικοινωνίας αυξάνεται. Η επιβάρυνση αυτή όμως δεν φαίνεται να επηρεάζει στην συνολική απόδοση των παράλληλων υλοποίησεων. Συνεπώς ο λόγος του χρόνου υπολογισμού προς τον χρόνο επικοινωνίας είναι αρκετά υψηλός. Παρατηρώντας τα αποτελέσματα



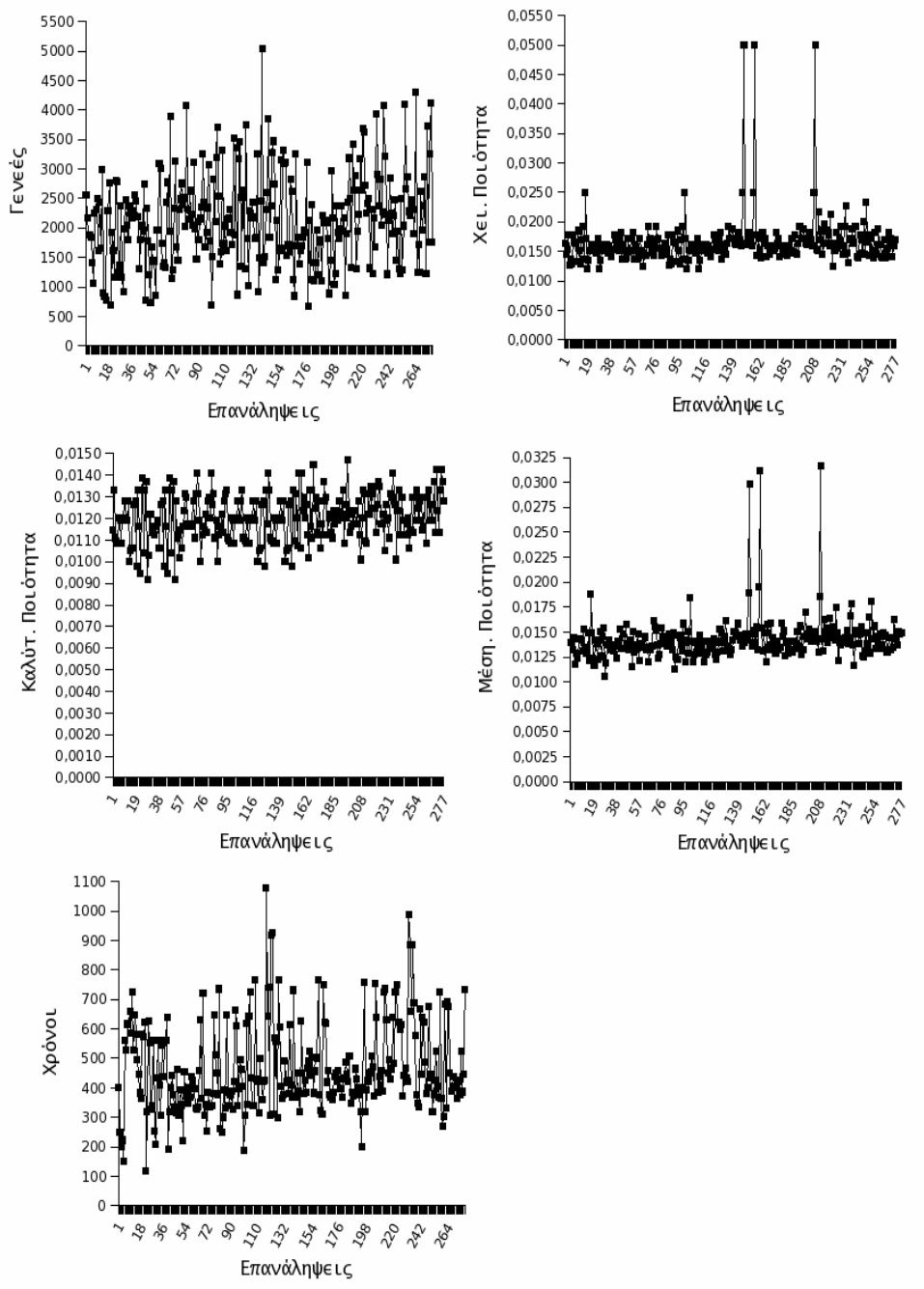
Σχήμα 7.3: Αποτελέσματα της PMA1 για 280 επαναλήψεις



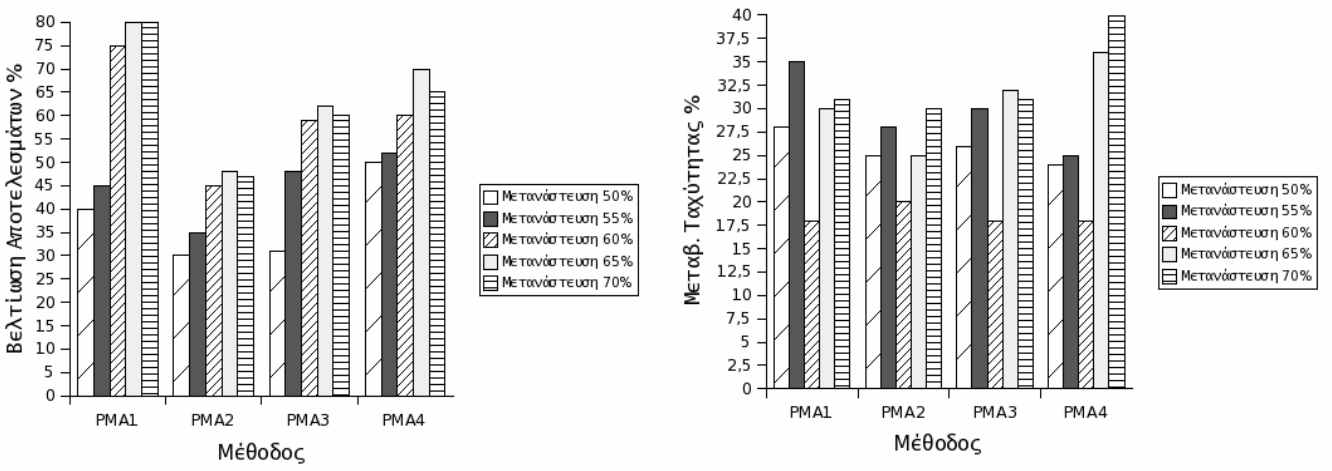
Σχήμα 7.4: Αποτελέσματα της PMA2 για 280 επαναλήψεις



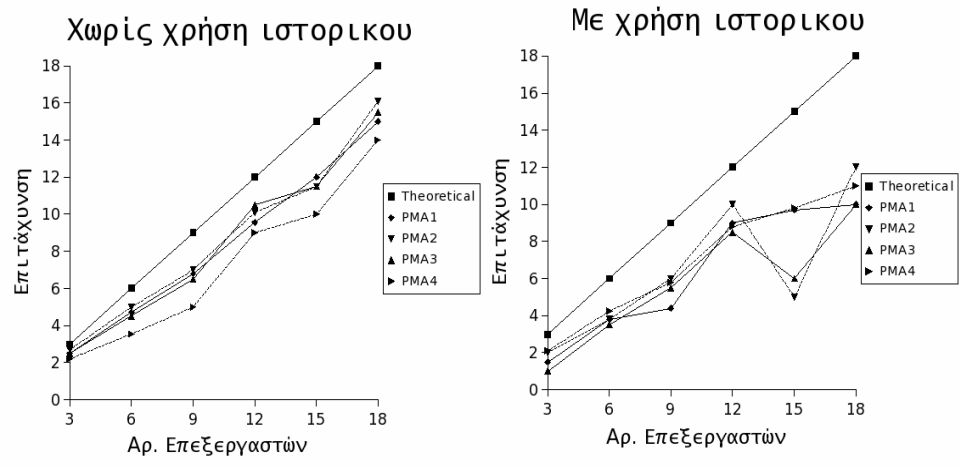
Σχήμα 7.5: Αποτελέσματα της PMA3 για 280 επαναλήψεις



Σχήμα 7.6: Αποτελέσματα της PMA4 για 280 επαναλήψεις



Σχήμα 7.7: Επίδραση της μεταβολής του ποσοστού μετανάστευσης στην ταχύτητα και στην ποιότητα των αποτελεσμάτων



Σχήμα 7.8: Επιτάχυνση των παράλληλων υλοποιήσεων συναρτήση της χρήσης καταγραφης του ιστορικού εξέλιξης (Αριστερά : Χωρίς ιστορικό , Δεξιά : Με ιστορικό)

PMA1	139
PMA2	180
PMA3	137
PMA4	136
Aydin et al	142
Hermosilla et al	138
PSA	167
PTS	165
PEA	152

Πίνακας 7.2: Οι καλύτερες τιμές για 9 μεθόδους

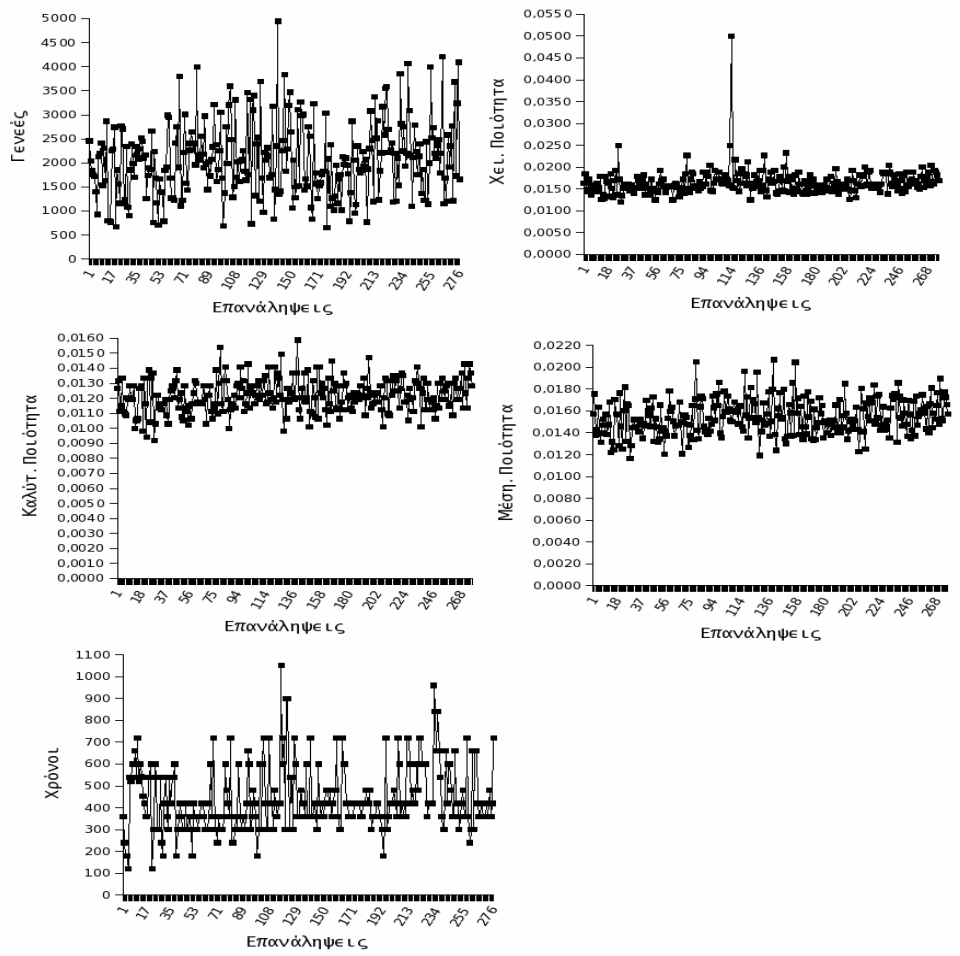
βλέπουμε ότι υπάρχει μια αντίστροφη σχέση ανάμεσα στον παράλληλο χρόνο εκτέλεσης και στον αριθμό των σταθμών εργασίας. Ο λόγος του χρόνου υπολογισμού προς το χρόνο επικοινωνίας εξαρτάται ως ένα βαθμό απο την ομοιογένεια των αλγορίθμων που εκτελούνται στις διάφορες νησίδες επεξεργαστών. Χρησιμοποιώντας ετερογενείς αλγορίθμους στην PMA4 παρατηρούμε ότι ο παράλληλος χρόνος εκτέλεσης των υλοποιήσεων μας βελτιώνεται σε σχέση με τον παράλληλο χρόνο εκτέλεσης με χρήση ομοιογενών αλγορίθμων. Ο χρόνος όμως επικοινωνίας είναι με μικρές αποκλίσεις περίπου ο ίδιος εφόσον διατηρούμε το μέγεθος πληθυσμού σταθερό. Στον πίνακα 7.2 έχουμε τις καλύτερες τιμές για 9 μεθόδους που εξετάστηκαν.

Στη διαρκεια των πειραμάτων μας διαπιστώσαμε ότι η μη συχνή μεταναστευση βοηθά στην αποφυγή της πρόωρης σύγκλισης του συνόλου των πληθυσμών. Επιπλέον η μετανάστευση σε συνδυασμό με την χρήση μιας μεθοδου τοπικής αναζήτησης βοηθά στο να μην χάνεται η ποικιλία του γενετικού υλικού και να μην γίνεται πρόωρη σύγκλιση σε

κάποιο τοπικό βέλτιστο. Η χρήση μιας μεθόδου τοπικής αναζήτησης δημιουργεί νέες ελπιδοφόρες περιοχές αναζήτησης λύσεων (βλ. σχήμα 7.7α). Αυτό είναι αρκετά σημαντικό γενικά, καθότι αυτές οι λύσεις αρχικά είναι τοπικές, που είναι και ο συνήθης κανόνας με τις ποικιλίες στη φύση, έτσι ώστε όμοια τροποποιημένα άτομα συχνά αναπαράγονται μεταξύ τους. Εάν ο νέος πληθυσμός είναι επιτυχής στον αγώνα για επιβίωση , θα διαδοθεί αργά, ανταγωνιζόμενος τους υπόλοιπους και κατακτώντας άτομα στα όρια ενός διαρκώς αυξανόμενου κύκλου. Στο σχήμα 7.8 βλέπουμε την επίδραση που έχει στην αποδοτικότητα και στην αποτελεσματικότητα της μεθοδου PMA4 ή καταγραφή του ιστορικού εξέλιξης. Τα καλύτερα αποτελέσματα απόδοσης προκύπτουν για τις μεθόδους PMA1, PMA4 ενώ λιγότερο καλά είναι τα αποτελέσματα για την PMA2 και την PMA3 .
Οι επιτάχυνσεις που πραγματοποιούνται οφείλονται όχι μόνο στον παραλληλισμό αλλά και στον χρόνο που επιτυγχάνεται με την αύξηση των αριθμών των νησίδων. Στην εργασία μας η εννοια της νησίδας διαφοροποιείται από την έννοια της παραλληλοποιηση ενός MA.

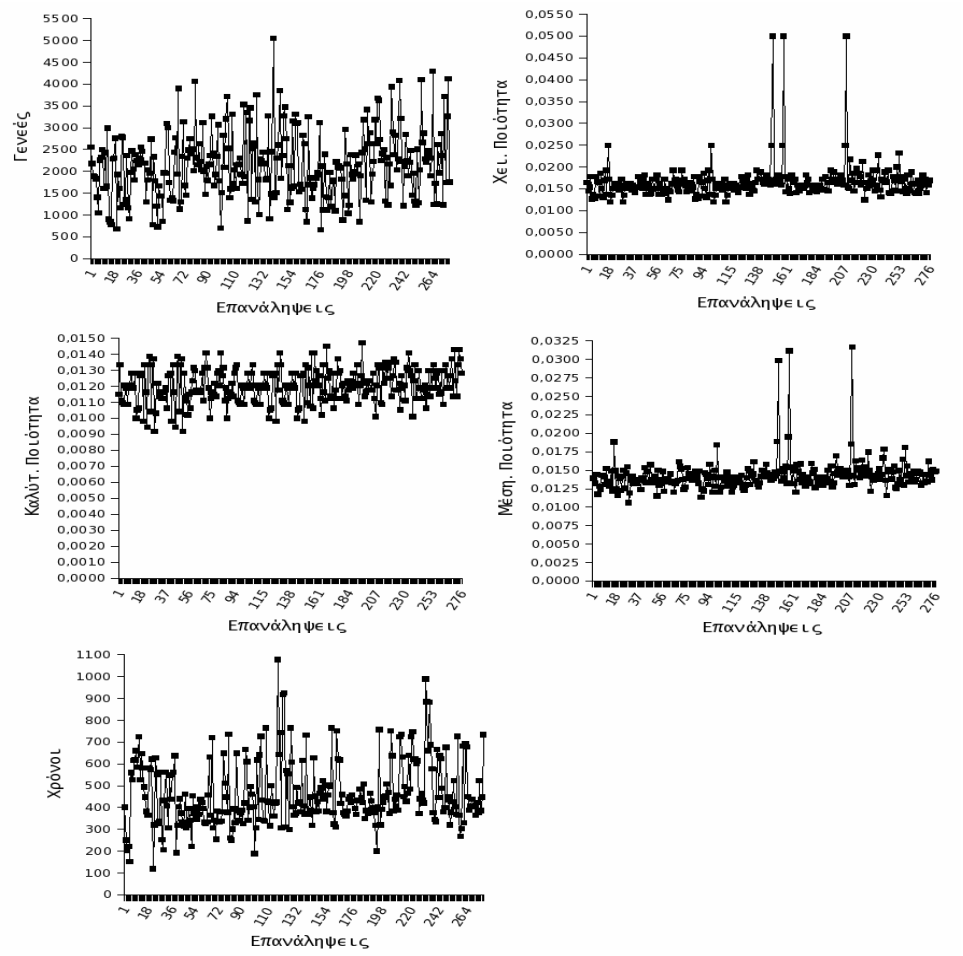
7.5 Συμπεράσματα

Από τα πειράματα που διεξήχθησαν παρατηρούμε ότι οι λύσεις που μας έδωσε ο αλγόριθμος είναι σχετικά καλές. Αυτό βέβαια σε ένα μεγάλο ποσοστό οφείλεται στο γεγονός ότι ορισμένοι περιορισμοί εκ των πραγμάτων είναι αδύνατο να πληρούνται εις το ακέραιο, ενώ παράλληλα οι συσχετίσεις που αναπτύσσονται μεταξύ των περιορισμών έχουν ως αποτέλεσμα ορισμένοι από αυτούς να λειτουργούν εις βάρος κάποιων άλλων.



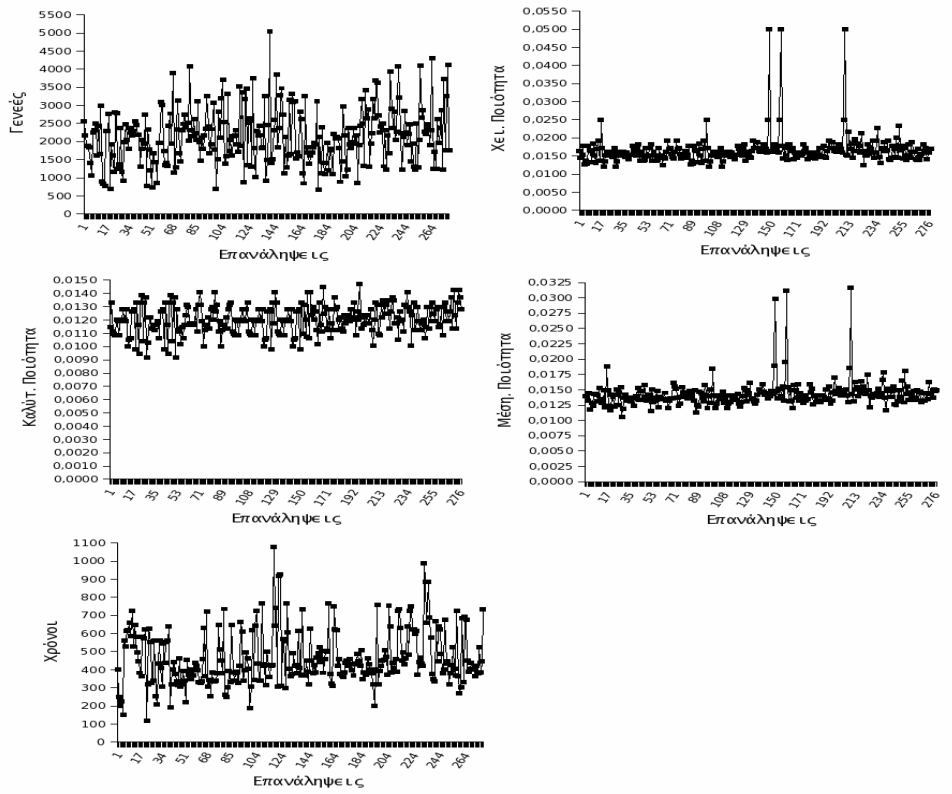
Σχήμα 7.9: Αποτελέσματα της PMA1 για 100 επαναλήψεις

Παρόλα αυτά, το κύριο πρόβλημα που αντιμετωπίζει ο αλγόριθμος είναι η σύντομη σύγκλιση του πληθυσμού γύρω από ένα τοπικό βέλτιστο. Το πρόβλημα αυτό είναι σύνηθες και συναντάται στις περισσότερες υλοποιήσεις εξελικτικών τεχνικών. Θα ήταν πολύ ενδιαφέρον να εφαρμοσει κάποιος ένα έξυπνο τελεστή μετάλλαξης, όπου η μετάλλαξη του ατόμου δεν θα γινόταν με απολύτως τυχαίο τρόπο, αλλά ο τελεστής θα προνοούσε ώστε οι τροποποιήσεις που θα έκανε πάνω στα γονίδια του ατόμου να μην οδηγούσαν



Σχήμα 7.10: Αποτελέσματα της PMA2 για 100 επαναλήψεις

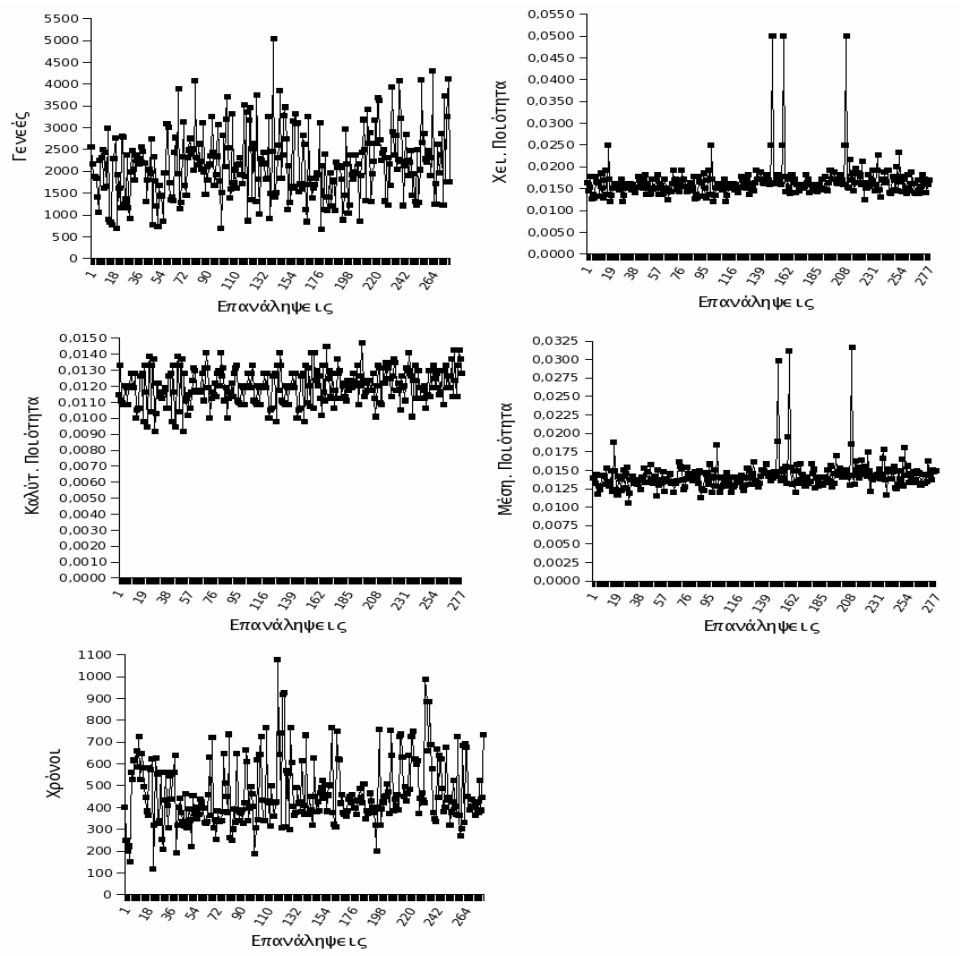
σε παραβιάσεις των αυστηρών περιορισμών. Από τα αποτελέσματα (σχήμα 7.8), γίνεται φανερή η υπεροχή της χρήσης εξελικτικών τελεστών έναντι της μη. Η υπεροχή αυτή είναι καθολική σε όλο το εύρος των πειραμάτων μας. Χαρακτηριστικό είναι το γεγονός ότι η καλύτερη λύση (σχήμα 7.14 που βρέθηκε χωρίς τη χρήση εξελικτικών τελεστών είναι μικρότερη από τη χειρότερη λύση που μας έδωσε η PMA2 που από τις υπόλοιπες



Σχήμα 7.11: Αποτελέσματα της PMA3 για 100 επαναλήψεις

4 υλοποιήσεις εδωσε και τα χειρότερα αποτελέσματα. Η χρησιμοποίηση γενετικών τελεστών όχι μόνο έδωσε καλύτερες λύσεις, αλλά βοήθησε και στη γρηγορότερη εύρεση αυτών, ανεβάζοντας ταυτόχρονα τη συνολική ποιότητα του πληθυσμού. Η υπεροχή της προσαρμογής οφείλεται σε δύο κυρίως λόγους:

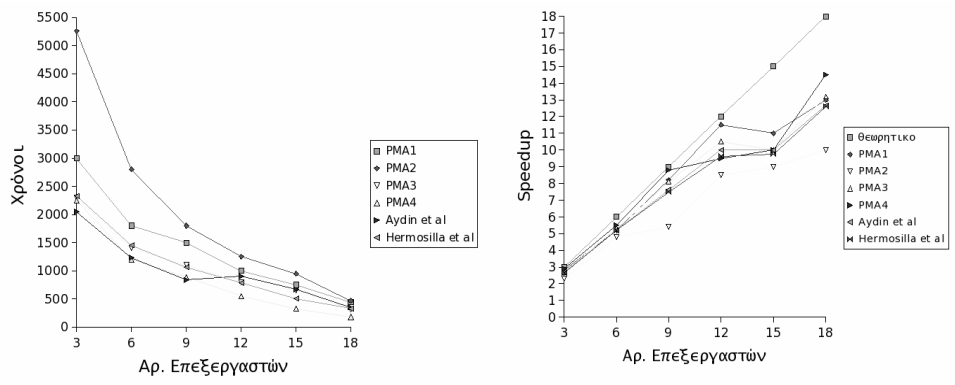
– Πρώτον, στο ότι η εύρεση των κατάλληλων πιθανοτήτων εφαρμογής των τελεστών



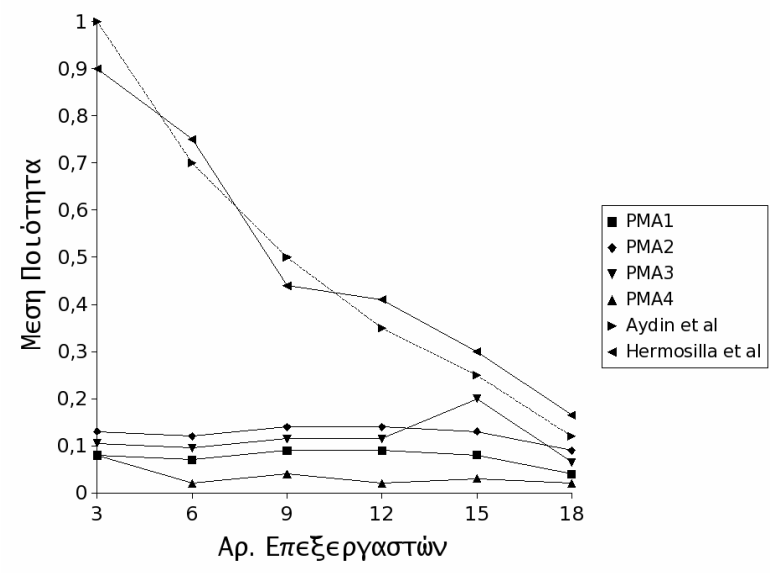
Σχήμα 7.12: Αποτελέσματα της PMA4 για 100 επαναλήψεις

είναι μια δύσκολη και περισσότερο εμπειρική διαδικασία δοκιμής και λάθους, αφού οι πιθανότητες διαφέρουν κατά πολύ από πρόβλημα σε πρόβλημα.

– Δεύτερον, στο ότι κατά τη διάρκεια της εξέλιξης οι ανάγκες του αλγορίθμου για αναζήτηση νέων λύσεων και εκμετάλλευση των υπαρχουσών είναι μεταβαλλόμενες,



Σχήμα 7.13: Χρόνοι για διαφορετικό αριθμό επεξεργαστών και οι επιταχύνσεις για 6 μεθόδους



Σχήμα 7.14: Μέσες λύσεις για 6 μεθόδους

με αποτέλεσμα οι πιθανότητες των δύο τελεστών να μην μπορούν να είναι σταθερές. Για παράδειγμα, κατά την εκκίνηση της εξέλιξης και δεδομένου ότι ο αρχικός

πληθυσμός αποτελείται από τυχαία άτομα, η ανάγκη εξερεύνησης του χώρου αναζήτησης είναι εντονότερη, ενώ με την πάροδο των γενιών αυξάνει η ανάγκη για εκμετάλλευση της αποκτούμενης γνώσης, γεγονός το οποίο επαληθεύεται και μέσα από τα πειράματα.

Εμφανείς είναι οι δυνατότητες στην επίλυση του αυτοματοποιημένου ωρολογίου προγράμματος μαθημάτων σαν βοηθητικών εργαλείων, των συγκεκριμένων υλοποιήσεων. Αυτό που επιτυγχάνεται με την χρήση των μεθόδων τοπικής αναζήτησης είναι η υπερπήδηση τοπικών βέλτιστων, ενώ ιδιαίτερα σημαντικό είναι η μελέτη του τρόπου με τον οποίο αλληλεπιδρούν οι περιορισμοί μεταξύ τους.

Κεφάλαιο 8

Συμπεράσματα - Προτάσεις

Σε αυτό το κεφάλαιο συνοψίζουμε τα αποτελέσματα της διατριβής μας και επίσης παρουσιάζουμε ορισμένες προτάσεις για μελλοντική έρευνα.

8.1 Αποτελέσματα

Στα πλαίσια της παρούσας διδακτορικής διατριβής επιχειρήθηκε μια ολοκληρωμένη μελέτη υβριδικών εξελικτικών τεχνικών και συγκεκριμένα των μιμητικών αλγορίθμων και η ανάπτυξη τεσσάρων νέων υλοποιήσεων σε συστοιχία από σταθμούς εργασίας. Πραγματοποιήθηκε διερεύνηση της επίδοσης των εξελικτικών μεθόδων, βάσει θεωρητικών προβλημάτων καθώς και πραγματικών προβλημάτων.

Τα κύρια συμπεράσματα της εργασίας συνοψίζονται στα εξής:

Μέχρι το πρόσφατο παρελθόν, οι δρόμοι των διαφόρων μεθοδολογιών προσέγγισης του

προβλήματος ήταν αποκλίνοντες. Ωστόσο, η σύγχρονη τάση συνίσταται στην ανάπτυξη υβριδικών εξελικτικών σχημάτων , τα οποία χρησιμοποιούν ιδέες και στρατηγικές προερχόμενες απο διαφορετικές μεθοδολογικές προσεγγίσεις, συμπεριλαμβάνομένων και των κλασσικών ευρετικών μεθόδων.

Δεδομένου ότι εξ ορισμού καμμία μέθοδος βελτιστοποίησης δεν εγγυάται παρά στατιστική και μόνο σύγκλιση στο ολικό ακρότατο, ζητούμενο είναι η εύρεση της μεθόδου εκείνης η οποία προσαρμόζεται στα χαρακτηριστικά του εκάστοτε προβληματος και μπορεί να έχει μια ικανοποιητική λύση, με το μικρότερο υπολογιστικό φόρτο. Οι Μιμητικοί Αλγόριθμοι έχουν γενικά πιο πολύπλοκη δομή από ένα ευρετικό αλγόριθμο καθώς είναι σύνθεση ευρετικών αλγορίθμων που περιέχουν και στοιχεία μίμησης βιολογικών και κοινωνικών διαδικασιών. Ανάλογα με την υλοποίηση εργάζονται με κωδικοποιημένο σύνολο λύσεων ή με τις λύσεις αυτές καθεαυτές. Στην πρώτη περίπτωση συγγενεύει πιο πολύ με τους γενετικούς αλγορίθμους στην δεύτερη με τους εξελικτικούς. Εκτελεί αναζητήσεις χρησιμοποιώντας ένα πληθυσμό λύσεων και όχι μια μοναδική λύση. Ο χρόνος εκτέλεσης του είναι εν γένει μεγαλύτερος από αυτόν κάποιου αντίστοιχου ευρετικού αλλά σε αντιστάθμισμα αναμένουμε λύσεις καλύτερης ποιότητας από αυτές που παράγει ο αντίστοιχος ευρετικός ή εξελικτικός. Κοινό χαρακτηριστικό όλων των μεθόδων που εξετάστηκαν ήταν η ευαισθησία τους ως προς ορισμένες αλγοριθμικές παραμέτρους εισόδου, όπως για παράδειγμα το μέγεθος του πληθυσμού. Κατά κανόνα, οι παράμετροι αυτές ορίζονται εμπειρικά, ωστόσο θα είχε ενδιαφέρον η ανάπτυξη τεχνικών αυτόματης ρύθμισης τους, αλλά το θέμα αυτό απαιτεί εκτενέστερη διερεύνηση. Στις επόμενες παραγράφους συνοψίζουμε τα συμπεράσματα μας για κάθε μια από τις περιπτώσεις που εξετάσαμε και

στην συνέχεια για κάθε μια από τις παραμέτρους που ελέγξαμε.

Εξελικτικοί Αλγόριθμοι

Η επίδοση του απλού εξελικτικού αλγορίθμου, με χρήση δυαδικής κωδικοποίησης των μεταβλητών ελέγχου, δεν ήταν ικανοποιητική, όχι μόνο λόγω της σχετικά χαμηλής του αποτελεσματικότητας αλλά κυρίως εξαιτίας του υπερβολικά μεγάλου πλήθους δοκιμών που απαιτείται για τη σύγκλιση στη βέλτιστη λύση. Η συμπεριφορά του απλού εξελικτικού αλγορίθμου δεν μπορεί να χαρακτηριστεί ικανοποιητική, ούτε ως προς την αποτελεσματικότητα ούτε (κυρίως) ως προς την αποδοτικότητα.

Η επίδραση των παραμέτρων εισόδου, δηλαδή του μεγέθους του πληθυσμού και των συχνοτήτων διασταύρωσης και μετάλλαξης, ήταν αρκετά σημαντική, και σε ορισμένες περιπτώσεις καθοριστική. Γενικά, αυξάνοντας το μέγεθος του πληθυσμού βελτιώθηκε η αξιοπιστία του αλγορίθμου, χωρίς η βελτίωση αυτή να είναι τέτοια που να δικαιολογεί τη μεγάλη αύξηση του πλήθους των απαιτούμενων δοκιμών.

Αυξάνοντας την τυχαιότητα του αλγορίθμου αυξήθηκε η πιθανότητα εντοπισμού της ολικά βέλτιστης λύσης μόνο στην περίπτωση που η αντικειμενική συνάρτηση ήταν έντονα μη γραμμική, διαφορετικά μειώθηκε η αποτελεσματικότητα του αλγορίθμου. Η αύξηση τόσο της συχνότητας διασταύρωσης όσο και της συχνότητας μετάλλαξης είχε ως συνέπεια μεγαλύτερο υπολογιστικό φόρτο.

Η συμπεριφορά των απλών παράλληλων εξελικτικών αλγορίθμων δεν μπορεί να χαρακτηριστεί πολύ ικανοποιητική, ούτε ως προς την αποτελεσματικότητα ούτε (κυρίως) ως προς την αποδοτικότητα. Σε σχέση όμως με τις σειριακές εκτελέσεις ειδαμε ότι δεν εχουμε καθολική αποτυχία εντοπισμού του ολικού ακρότατου στην 10-διάστατη Rozenbrock

όπως και άλλων προβλημάτων στα οποία με τη σειριακή εκτέλεση ειχαμε αποτύχει.

Γενετικοί Αλγόριθμοι

Η συμπεριφορά του γενετικού αλγορίθμου δεν μπορεί να χαρακτηριστεί ικανοποιητική. Γενικά, αυξάνοντας το μέγεθος του πληθυσμού βελτιώθηκε η αξιοπιστία του αλγορίθμου, χωρίς η βελτίωση αυτή να είναι τέτοια που να δικαιολογεί τη μεγάλη αύξηση του πλήθους των απαιτούμενων δοκιμών. Η αύξηση της τιμής της συχνότητας διασταύρωσης από 60% σε 100% όπως και η αύξηση της συχνότητας μετάλλαξης από 0% σε 20% διαφοροποίησε προς το καλύτερο τα αποτελέσματα.

Η επίδραση των παραμέτρων εισόδου, δηλαδή του μεγέθους του πληθυσμού και των συχνοτήτων διασταύρωσης και μετάλλαξης, ήταν αρκετά σημαντική, και σε ορισμένες περιπτώσεις καθοριστική. Τα ποσοστά ανασυνδυασμου τάξης $70\% \sim 75\ \%$ βοηθούν στην αποφυγή της πρόωρης σύγκλισης του πληθυσμού, δηλαδή του μοντέλου των GA. Δίνεται έτσι η δυνατότητα για περαιτέρω διερεύνηση διαφορετικών περιοχών του χώρου λύσεων και δημιουργίας γενετικής πληροφορίας η οποία καθώς εισάγεται στον πληθυσμό βοηθά στον να προχωρήσει η αναζήτηση του ολικού βέλτιστου. Το τελικό συμπέρασμα που προκύπτει για αυτή τη παράμετρο είναι ότι πολύ μικρά ποσοστά ανασυνδυασμού των γονέων με μικρά μεγέθη πληθυσμού υποβαθμίζουν και άλλο την απόδοση των GA.

Η συμπεριφορά των παράλληλων γενετικών αλγορίθμων είναι λιγότερο ικανοποιητική από αυτήν των παράλληλων εξελικτικών αλγορίθμων. Χαρακτηριστικό είναι το γεγονός της καθολικής αποτυχίας εντοπισμού αρκετών προβλημάτων και στην παράλληλη εκδοχή.

Μιμητικός Αλγόριθμος με αποτρεπτική αναζήτηση

Ο μιμητικός αλγόριθμος με αποτρεπτική αναζήτηση αντιμετώπισε με σχετική επιτυχία ορισμένα από τα προβλήματα, ενώ απέτυχε εντελώς σε άλλα. Παρατηρήθηκε βέβαια αύξηση της αποτελεσματικότητας με αύξηση του πλήθους εκκινήσεων, ωστόσο τα περιθώρια βελτίωσης ήταν πεπερασμένα και εξαρτώμενα από τις ιδιαιτερότητες του εκάστοτε προβλήματος. Η αποτελεσματικότητα της μεθόδου παρουσίασε βελτίωση με αύξηση του πλήθους των εκκινήσεων, χωρίς ωστόσο η βελτίωση αυτή να είναι ομοιόμορφη. Ο αλγόριθμος αυτός πραγματοποιεί αρχικά μια αδρή και στη συνέχεια μια πιο λεπτομερή διερεύνηση του εφικτού χώρου, Προφανώς, απαιτήθηκε η θεώρηση μεγαλύτερου πληθυσμού όσο αυξανόταν ο βαθμός δυσκολίας του προβλήματος.

Μιμητικός αλγόριθμος με προσωμοιούμενη ανόπτηση

Η μεθοδος του μιμητικού αλγορίθμου με προσωμοιούμενη ανόπτηση αποδείχθηκε ιδιαίτερα αποτελεσματική για τις περισσότερες συναρτήσεις ενώ ο μιμητικός αλγόριθμος με καθοδηγούμενη τοπική αναζήτηση αποδείχθηκε εξίσου αποτελεσματικός, μειονεκτώντας λίγο ως προς την ταχύτητα σύγκλισης.Η μεθοδος αντιμετώπισε με πολυ καλό ποσοστό σχεδόν τις περισσότερες από τις συναρτήσεις ελέγχου και μάλιστα με πολύ μικρότερο αριθμό δοκιμών σε σχέση με τον απλό εξελικτικό αλγόριθμο και τον μιμητικό αλγόριθμο με αποτρεπτική αναζήτηση.

Η θεώρηση μεγαλύτερου πληθυσμού όσο αυξανόταν ο βαθμός δυσκολίας του προβλήματος. Η αύξηση της τιμής της συχνότητας διασταύρωσης από 60% σε 100% επαιξε σημαντικό ρόλο στα αποτελέσματα, ενώ αντίθετα η αύξηση της συχνότητας μετάλλαξης από 1% σε 20% τα διαφοροποίησε όχι σημαντικά, πάντα όμως προς τη θετική κατεύθυνση. Δηλαδή, αυξάνοντας την τυχαιότητα του αλγορίθμου αυξήθηκε η πιθανότητα εντοπισμού

της ολικά βέλτιστης λύσης μόνο στην περίπτωση που η αντικειμενική συνάρτηση ήταν έντονα μη γραμμική, διαφορετικά μειώθηκε η αποτελεσματικότητα του αλγορίθμου.

Μιμητικός αλγόριθμος με κατευθυνόμενη τοπική αναζήτηση

Ο αλγόριθμος αυτός παρουσίασε πολύ καλή επίδοση, κυρίως ως προς την αποτελεσματικότητα αλλά και ως προς την αποδοτικότητα. Και σε αυτήν την μέθοδο είναι προφανές ότι με χρήση μικρού μεγέθους πληθυσμού απαιτείται μικρότερος αριθμός δοκιμών για τον εντοπισμό της βέλτιστης λύσης, και συνεπώς βελτιώνεται σημαντικά η αποδοτικότητα του αλγορίθμου. Η μέθοδος αντιμετώπισε με καλό ποσοστό σχεδόν τις περισσότερες από τις συναρτήσεις ελέγχου. Η χρήση της κατευθυνόμενης τοπικής αναζήτησης στο σταδιο της αρχικοποίησης του πληθυσμού σε συνδυασμό με τις ρυθμίσεις των υπολοίπων παραμέτρων απεβηκε απόλυτη επαρκής για την αντίμετωπιση προβλημάτων που εξετάστηκαν στην παρούσα διατριβή. Η χρήση της κατευθυνόμενης τοπικής αναζήτησης αυξησε την αποτελεσματικότητα της μεθόδου καθώς αντιμετωπιστηκαν στην πλειοψηφία των περιπτώσεων οι αδυναμίες διαφυγής από τις κορυφογραμμές.

Παράλληλοι Μιμητικοί Αλγόριθμοι

Στην κατεύθυνση της αύξηση κυρίως της αποδοτικότητας και της βελτίωσης της αποτελεσματικότητας των ευρετικών και εξελεκτικών μεθόδων μεσω της παραλληλοποίησης βρίσκονται αρκετοί από τους ερευνητές του χώρου. Σε όλες τις περιπτώσεις που εξετάστηκαν στα πλαίσια της διατριβής ο συνολικός πληθυσμός λύσεων επιμερίζεται σε ομάδες, σε κάθε μια από τις οποίες η εξέλιξη γίνεται ανεξάρτητα, ενώ κατά διαστήματα πραγματοποιείται ανταλλαγή πληροφοριών πράγμα το οποίο καθιστά εφικτή την παράλληλη υλοποίηση τους.

Οι τρείς από τις μεθόδους μας PMA1-PMA3 παρουσιάζουν ομοιογένεια ως προς τους αλγορίθμους, ενώ στην PMA4 φαίνεται από τα αποτελέσματα ότι η υλοποίηση ετερογενών εξελικτικών και ευρετικών μεθόδων σε μια γενική και κατάνεμημένη αρχιτεκτονική αυτή της πειραματικής συστοιχίας ετερογενών σταθμών εργασίας βρίσκεται στην κατεύθυνση της βελτίωσης όχι μόνο της αποδοτικότητας αλλά και της αποτελεσματικότητας. Είναι μια νέα προσέγγιση του προβλήματος ολικής βελτιστοποίησης, έχοντας ως βάση μια αποτελεσματική σύνθεση ιδεών παρμένων από επιμέρους μεθοδολογίες. Η εφαρμογή μηχανισμών τοπικής αναζητησης εχει ευεργετικά αποτελέσματα στην ποιότητα των αποτελεσμάτων αλλά απαιτεί πολύ υπολογιστικό χρόνο κάτι που μετα την παραλληλοποίηση αντισταθμίζεται. Στο επίπεδο της παραλληλοποίησης σε συστοιχία εφαρμόζουμε το προγραμματιστικό μοντέλο συντονιστης - εργαζόμενος που ο συντονιστής χωρίζει τον πληθυσμό σε νησίδες και αυτές διανέμονται με το μοντέλο της δυναμικής διανομής.

Υλοποίηση PMA1

Η μέθοδος PMA1 αντιμετώπισε με σχετική επιτυχία αρκετά από τα προβλήματα. Απαιτήθηκε η θεώρηση μεγαλύτερου πληθυσμού όσο αυξανόταν ο βαθμός δυσκολίας του προβλήματος. Έτσι, ενώ ήταν επαρκής μία και μόνο νησίδα για την εύρεση του ελαχίστου της διδιάστατης Rozenbrock και της Extended Rozenbrock, απαιτήθηκαν τέσσερις νησίδες για την επίλυση των ίδιων προβλήματων στις 10 διαστάσεις, ενώ οι 4 νησίδες δεν ήταν επαρκείς για την επίλυση των προβλήματων Penalty II, Rastrigin, Powel badly scaled με ικανοποιητική αξιοπιστία. Η ταχύτητα της μεθόδου παρουσίασε σημαντικές διαφορές, ανάλογα με το βαθμό δυσκολίας του εκάστοτε προβλήματος και ήταν προφανώς ανάλογη του αριθμού των επεξεργαστών.

Υλοποίηση PMA2

Στα αποτελέσματα παρατηρούμε ότι η PMA2 είναι λιγοτερο αποτελεσματική από τις υπόλοιπες τρεις υλοποιήσεις. Ο καθορισμός των γειτονικών λυσεων απαιτεί μια διαδικασία διακριτοποίησης του εφικτού χώρου κάτι το οποίο κάνει τον αργό σταθμό εργασίας να τελειώνει πάντα τελευταίος και να υπάρχει ένας σημαντικός χρόνος αδράνειας για τους ταχύτερους σταθμούς εργασίας. Ο χρόνος όμως επικοινωνίας είναι με μικρές αποκλίσεις περίπου ο ίδιος εφόσον διατηρούμε το μέγεθος πληθυσμού σταθερό. Οι χειρότερες επιδόσεις προκύπτουν για πολυ μικρές και μεγάλες νησίδες κάτι το οποίο οφείλεται στο ότι ο χρόνος που απαιτείται από την διαδικασία της ανταλλαγής ατόμων εξαρτάται από τον αριθμό των νησίδων και οι υπολογισμοί χρειάζονται λιγότερο χρόνο.

Υλοποίηση PMA3

Από τα αποτελέσματα γίνεται φανερό ότι η μέθοδος παρουσίασε πολύ καλή επίδοση, κυρίως ως προς την αποτελεσματικότητα αλλά και ως προς την αποδοτικότητα. Συγκεκριμένα, εντόπισε λύσεις της τάξης του 95% για 5 από τις συναρτήσεις ελέγχου. Η μη συχνή μετανάστευση βοηθά στην αποφυγή της πρόωρης σύγκλισης του συνόλου των πληθυσμών. Η χρήση της κατευθυνόμενης τοπικής αναζήτησης δημιουργεί νέες ελπιδοφόρες περιοχές αναζήτησης λύσεων (βλ. σχήματα 5.11). Αυτό είναι αρκετά σημαντικό γενικά, καθότι αυτές οι λύσεις αρχικά είναι τοπικές. Όταν ο νέος πληθυσμός είναι επιτυχής στον αγώνα για επιβίωση , διαδίδεται αργά. Και σε αυτήν την μέθοδο είναι προφανές ότι με χρήση μικρού μεγέθους πληθυσμού απαιτείται μικρότερος αριθμός δοκιμών για τον εντοπισμό της βέλτιστης λύσης, και συνεπώς βελτιώνεται σημαντικά η αποδοτικότητα του αλγορίθμου. Είναι προφανές ότι με χρήση μικρου μεγέθους πληθυσμού απαιτείται μικρότερος αριθμός

δοκιμών για τον εντοπισμό της βέλτιστης λυσης, και συνεπώς βελτιώνεται σημαντικά η αποδοτικότητα του αλγορίθμου. Η μεθοδος αυτή βρήκε σχετικά καλές λύσεις στα προβλήματα μας.

Υλοποίηση PMA4

Από τα αποτελέσματα φαίνεται ότι η PMA4 αντιμετώπισε με μεγαλη επιτυχία από 80% ∼ 100% τα περισσότερα από τα προβλήματα. Φαίνεται ότι η μεθοδος είναι πράγματι μια σχετικά καλυτερη μέθοδος από αυτή του απλού εξελικτικού αλγορίθμου, αφού αντιμετώπισε με καλύτερο ποσοστό σχεδόν τις περισσότερες από τις συναρτήσεις ελέγχου (με εξαίρεση την 10-διαστατη Extended Rozenbrock και την Powel badly scaled). Απαιτήθηκε η θεώρηση μεγαλύτερου πληθυσμού όσο αυξανόταν ο βαθμός δυσκολίας του προβλήματος.

Πληθυσμός

Αρχικά η απόδοση των μοντέλων με μέγεθος πληθυσμού 100 είναι καλυτερη από την απόδοση άλλων μεγεθών πληθυσμού, αλλά συνήθως καθηλώνεται ο αλγόριθμος σε τοπικά ελάχιστα. Η χρήση μεγάλων τιμών για το μέγεθος πληθυσμού δεν συνεπάγεται σε όλες τις περιπτώσεις καλύτερα αποτελέσματα και εκτός τούτου έχει σαν αποτέλεσμα την αύξηση του υπολογιστικού χρόνου. Μια μεγάλη τιμή για το μέγεθος πληθυσμού μπορεί να χρησιμοποιηθεί για να βελτιώσει την απόδοση, δηλαδή όταν υπάρχει διαθέσιμος χρόνος, ενώ μια μικρή τιμή απαιτείται όταν ο στόχος είναι η καλή απόδοση, δηλαδή όταν τα αποτελέσματα απαιτούνται σε σύντομο χρόνο.

Οι μεγάλοι πληθυσμοί δημιουργούν προβλήματα καθώς κινούνται αργά πρός τη βέλτιστη λύση και δημιουργούν πολλούς απογόνους σε κάθε γενιά με αποτέλεσμα να χρειάζονται

πολλές επαναλήψεις. Πολλές φορές οδηγούν σε πιο ακριβή λύση αλλά δεν δίνουν πάντα τη καλύτερη απάντηση σε ένα πρόβλημα. Ο λόγος για αυτό είναι ότι συχνά με τη χρήση μεγάλου πληθυσμού ο αλγόριθμος παραβλέπει μια λύση με συνάρτηση κόστους άνω του μέσου όρου η οποία βρίσκεται σε μια στενή κορυφή. Από την άλλη πλευρά οι πολύ μικροί πληθυσμοί οδηγούν σε πρώιμη σύγκλιση. Η μείωση του πληθυσμού οδηγεί σε μείωση της ποικιλότητας των ατόμων με αποτέλεσμα ο αλγόριθμος να μην έχει τη δυνατότητα να ψάξει σε μεγάλο τμήμα του πεδίου αναζήτησης και να παγιδευτεί σε τοπικά μέγιστα. Το βέλτιστο μέγεθος του πληθυσμού ουσιαστικά εξαρτάται από το πρόβλημα που πρέπει να λύσουμε και το είδος της μεθοδολογίας.

Μέθοδος Αντικάτασταση

Το μοντέλο ολικής αντικατάστασης και το μοντέλο σταθερής αντικατάστασης. Πιο συγκεκριμένα στη μέθοδο γενεαλογικής αντικατάστασης σε κάθε γενιά ο πληθυσμός αντικαθίσταται ολόκληρος ενώ στη δεύτερη περίπτωση μόνο ένα μέρος αυτού. Στην πλειονότητα των περιπτώσεων η μέθοδος γενεαλογικής αντικάτασταση απαιτουσε περισσότερο υπολογιστικό χρόνο από ότι η σταθερής αντικατάστασης

Ανασυνδυασμός και Μετάλλαξη

Η μέθοδος που ακολουθείται για το σταδιο του ανασυνδυασμου και της μετάλλαξης επιδρά σημαντικά στο φορτίο επικοινωνίας ενός παράλληλου μιμητικού αλγόριθμου που πρέπει να ανταλλάξει τα στοιχεία αυτά. Ο μηχανισμός του ανασυνδυασμού δεν εφαρμόζεται συνήθως σε όλα τα ζευγάρια των ατόμων που έχουν επιλεγεί για αναπαραγωγή. Εάν δεν εφαρμοστεί η μέθοδος αυτή, οι απόγονοι γεννιουνται με απλή αντιγραφή των

γονιών. Τα καλύτερα αποτελέσματα τα έδωσε ο ευρετικός αλγόριθμος. Το κυριώτερο προτέρημα του τελεστή αυτού συνεπώς είναι οτι οδηγεί στη καλύτερη κατεύθυνση αναζήτησης. Επίσης συμβάλλει στην ακρίβεια της λύσης που βρίσκει ο αλγόριθμος. Η τιμή της συχνότητας μετάλλαξης είναι αντιστρόφως ανάλογη του μεγέθους πληθυσμού. Ο μηχανισμός της μετάλλαξης βοηθά τον μιμητικό αλγόριθμο να διαφέυγει από τα τοπικά ακρότατα, παρέχοντας μια επιπλέον συνιστώσα τυχαιότητας στη διαδικασία της εξέλιξης.

Ιστορικό Εξέλιξης

Με την κατάγραφή του ιστορικού εξέλιξης ο χρόνος επικοινωνίας αυξάνεται. Οι μικρές τιμές του πληθυσμού αυξάνουν την επιβάρυνση της διαεπεξεργαστικής επικοινωνίας, ενώ οι μεγάλες τιμές παρέχουν χαμηλή ισορροπία φορτίου. Όταν το ιστορικό εξέλιξης χρησιμοποιείται ως πηγή πληροφοριών, τότε το ιστορικό αυτό θα πρέπει να διατηρείται σταθερό και κάθε μονάδα επεξεργασίας πρέπει να μπορεί να έχει πρόσβαση σε αυτό, όταν είναι απαραίτητο. Το υπολογιστικό κόστος της πρόσβασης αυτής είναι υψηλό, όσον αφορά την επικοινωνία, όποια τεχνική και αν χρησιμοποιηθεί για να πραγματοποιηθεί.

Μετανάστευση

Όταν οι υποπληθυσμοί επικοινωνούν αραιά έχουν την δυνατότητα να ερευνήσουν διαφορετικές περιοχές του χώρου λύσεων και να διαθέτουν διαφορετικές περιοχές του χώρου λύσεων και να διαθέτουν διαφορετικές πληροφορίες. Όταν αυτές οι πληροφορίες εισάγονται σε άλλους υποπληθυσμούς, οι οποίοι έχουν συγκλίνει πρόωρα σε κάποιο τοπικό βέλτιστο, τους βοηθούν να ξεφύγουν και να συνεχίσουν την αναζήτηση του ολικού βέλτιστου. Η χρήση της τοπικής αναζήτησης βοηθά επιπλέον στον να μην χρησιμοποιούμε

μεγάλο ποσοστό μετανάστευσης αλλα και στον να μην έχουμε απώλεια της ποικιλίας του γενετικού υλικού.

Νησίδες και Αριθμός Επεξεργαστών

Οι χειρότερες επιδόσεις προκύπτουν για πολυ μικρές και μεγάλες νησίδες κάτι το οποίο μπορεί να εξηγηθεί με δύο τρόπους:

α) μόνο κάποια μέρη του αλγορίθμου έχουν πολυπλοκότητα $\Theta(n^2)$ ενώ το υπόλοιπο έχει χαμηλότερη πολυπλοκότητα και κατά συνέπεια μόνο ορισμένα τμήματα του αλγορίθμου επηρεάζουν τις τιμές του χρόνου.

β) ο χρόνος που απαιτείται από την διαδικασία της ανταλλαγής ατόμων εξαρτάται από τον αριθμό των νησίδων και οι υπολογισμοί χρειάζονται λιγότερο χρόνο.

Από τις κατηγορίες εφαρμογών που διερευνήθηκαν, δύο είναι αυτές που παρουσίασαν το μεγαλύτερο βαθμό δυσκολίας. Στην πρώτη εντάσσονται προβλήματα με επίπεδη επιφάνεια απόκρισης. Σε αυτά υπάρχει μεγάλη δυσκολία στον εντοπισμό της διεύθυνσης βελτίωσης της αντικειμενικής συνάρτησης, η οποία επιδεινώνεται με την αύξηση του πλήθους των μεταβλητών ελεγχου. Στη δεύτερη κατηγορία ανήκουν προβλήματα που παρουσιάζουν μεγάλη δυσκολία στις τιμές των παραμέτρων και πολλά τοπικά ακρότατα.

Εφαρμογές

Τα πραγματικά προβλήματα που επιλυσαμε μπορούν να χαρακτηριστούν αντιπροσωπευτικά προβλήματα βελτιστοποίησης με περιορισμούς. Τα προβλήματα αυτά αφορούσαν τον χρονοπρογραμματισμό παραγωγής από μονάδες ηλεκτρικού ρεύματος και το εξαμηνιαίο πανεπιστημιακό ωρολόγιο πρόγραμμα. Τα προβλήματα αυτά μπορούν να θεωρηθουν ως

ένα βαθμό πρωτοτυπα.

Το πρόγραμμα για τον χρονοπρογραμματισμό παραγωγής από μονάδες ηλεκτρικού ρεύματος είναι μία πρότυπη εφαρμογή που μπορεί να επεκταθεί και να καλύψει τις ανάγκες μεγάλων εταιριών ηλεκτροδότησης, με σχετικά εύκολο τρόπο. Αυτό μπορεί να γίνει με την είσοδο των αρχικών μεταβλητών που απαιτούνται για τη μελέτη του προβλήματος της κάθε εταιρίας.

Εμφανείς είναι οι δυνατότητες των μεθόδων μας και στην επίλυση του αυτοματοποιημένου ωρολογίου προγράμματος μαθημάτων σαν βοηθητικών εργαλείων, των συγκεκριμένων υλοποιήσεων. Αυτό που επιτυγχάνεται με την χρήση των μεθόδων τοπικής αναζήτησης είναι η υπερπήδηση τοπικών βέλτιστων, ενώ ιδιαίτερα σημαντικό είναι η μελέτη του τρόπου με τον οποίο αλληλεπιδρούν οι περιορισμοί μεταξύ τους.

8.2 Προτάσεις για μελλοντική έρευνα

Στην ενότητα αυτή θα κάνουμε μερικές προτάσεις για μελλοντική εργασία γύρω από την βελτίωση των μεθόδων μας. Το βασικότερο μειονέκτημα των μεθόδων μας είναι το μεγάλο πλήθος των παραμέτρων που πρέπει να ρυθμιστούν έτσι ώστε να είναι βέλτιστη η απόδοση του. Η βέλτιστη ρύθμιση των παραμέτρων των αλγορίθμων είναι και αυτή ένα δύσκολο πρόβλημα βελτιστοποίησης και επομένως ο χρήστης θα πρέπει να συμβιβαστεί με μια περιοχή τιμών που θα δίνει καλά αποτελέσματα.

Ενδιαφέρον θα ήταν να επεκτείνει κανείς την παραπάνω υλοποίηση σε υπολογισμούς πλέγματος που περιέχουν επιμέρους συστοιχίες σταθμών εργασίας με ετερογένεια όχι μόνο

ως προς τις ταχύτητες των σταθμών εργασίας και των αλγορίθμων που τρέχουν αλλά και ως προς το δίκτυο επικοινωίας, το λειτουργικό σύστημα, τις ταχύτητες μνήμης, τις ταχύτητες πολυεπίπεδων δικτύων κλπ.

Μια αλλη επίσης εξελισσόμενη τα τελευταία δύο χρόνια μεθοδολία είναι και αυτή της βελτιστοποίησης με αποικίες ψηφιακών μυρμηγκιών. Θα ήταν ενδιαφέρον αν κάποιος επιχειρούσε να συνδυάσει ευρετικές τεχνικές όπως της προσομοιούμενης ανόπτησης και της αποτρεπτικής αναζήτησης στο στάδιο της επαναρχικοποίησης του πληθυσμού κατά τη δημιουργία ανεξάρτητων αποικιών μυρμηγκιών (distributed colonies) η οποίες θα διανέμονται σε μια συστοιχία σταθμών εργασίας.

Είδαμε ότι οι υλοποίησεις έχουν το μειονέκτημα της υψηλής επιβάρυνσης επικοινωνίας. Το πρόβλημα αυτό μπορεί να εξαλειφθεί από μια μέθοδο προεπεξεργασίας διανομής πληθυσμών, όπου οι υποπληθυσμοί τοποθετούνται σε ένα από τους σταθμούς εργασίας έτσι ώστε η διαφορά ανάμεσα στο μέγεθος του υποπληθυσμού στο μικρότερο και μεγαλύτερο σταθμό εργασίας να ελαχιστοποιείται.

Η διερεύνηση διαδικασιών αυτόματης ρύθμισης ορισμένων παραμέτρων εισόδου της μεθοδολογίας PMA2 που περιγράφηκαν όπως οι συντελεστές του χρονοδιαγράμματος ανόπτησης θα μπορούσε να συμβάλλει τόσο στην μείωση της παρέμβασης του χρήστη στις εσωτερικές διεργασίες του αλγορίθμου PMA2 όσο και στην εξαγωγή πιο αξιόπιστων αποτελεσμάτων με υπολογιστικό φορτίο.

Παράρτημα Α΄

Λογισμικό PARAMENOAS

Α΄.1 Οδηγός Εγκατάστασης του συνοδευτικού λογισμικού

Α΄.1.1 Οδηγίες για την εγκατάσταση

Η βιβλιοθήκη PARAMENOAs μπορεί να εγκατασταθεί σε ένα μεγάλο αριθμό συστημάτων και αρχιτεκτονικών. Για την καλύτερη κατανόηση των επόμενων παραγράφων κάνουμε τις εξής παραδοχες. Με το PARAMENOASDIRECTORY αναφερόμαστε στον κατάλογο που περιέχει όλο το λογισμικό, με το MPIINCDIR αναφερόμαστε στον κατάλογο που περιέχει το αρχείο κεφαλίδας mpi.h, ενώ ο κατάλογος MPILIBDIR περιέχει την βιβλιοθήκη libmpi.a.

Η βιβλιοθήκη PARAMENOAS βασίζεται στον κώδικα της βιβλιοθήκης parSA[122], της τροποποιημένης έκδοσης της βιβλιοθηκης PGAPack [132] καθώς και άλλων προγραμμάτων [27]. Έχει σχεδιαστεί για παράλληλες αρχιτεκτονικές και απαιτείται η εγκατάσταση της βιβλιοθήκης

διεπαφής περάσματος μηνυμάτων MPI. Η τελευταία έκδοση είναι αυτή που δίδεται στη διεύθυνση http://paramenoas.sourceforge.net/.Στα σχεδια μας για το προσεχές μέλλον είναι να προσθέσουμε και κώδικα για αποικίες ψηφιακών μυρμηγκιών ant colony optimization.

Α′.1.2 Εγκατάσταση της Βιβλιοθήκης

Σε πρώτη φάση θα πρέπει να

- Ελέγχετε αν
- είναι εγκατεστημένο το πακέτο gcc2.96-cpp καθώς και το gcc-g77
- αν έχουν οριστεί οι κατάλογοι MPIINCDIR,paramenoas/example, paramenoas/include στα αρχεία κεφαλίδας .

Δημιουργείτε ένα καταλογο στην περιοχή σας με το όνομα paramenoas. Στην συνέχεια αποσυμπιέζετε το αρχείο που υπάρχει στο C μέσα στον κατάλογο που δημιουργήσατε με τις εντολές:

- gzip -d paramenoas.tar.gz
- tar -xvf paramenoas.tar

Στην συνέχεια δημιοργούνται οι υποκατάλογοι lib και include που περιέχουν τα αρχεία βιβλιοθηκών και κεφαλίδων αντίστοιχα. Τα βασικά αρχεία των βιβλιοθηκών είναι της μορφής libparMEME$(OPERATINGSYSTEM)$(OS_VERSION).a. Στον κατάλογο src έχουμε τα αρχεία με τον πηγαίο κώδικα του PARAMENOAS.Ο υποκατάλογος example περιέχει παραδείγματα εφαρμογών. Τέλος υπάρχει και άλλος ένας κατάλογος ο cfg με αρχεία ρυθμίσεων .

Στην συνέχεια χρησιμοποιείται η εντολή configure για να δημιουργήσουμε τα αρχεία makefiles ως εξής:

configure -arch ARCH_TYPE

όπου το όρισμα ARCH_TYPE παίρνει τις παρακάτω τιμές

- linux για απλούς υπολογιστές που τρέχουν λειτουργικό σύστημα Linux.
- irix για Silicon Graphics σταθμούς εργασίας.

Η πλήρης σύνταξη της εντολής configure είναι :

```
configure -arch ARCH_TYPE [-cc CC]  [-cflags CFLAGS] [-f77 F2C]  [-debug]
[-mpiinc MPI_INCLUDE_DIRECTORY] [-mpilib MPI_LIBRARY]  [-help]
```

όπου

- CC ειναι ο C μεταγλωττιστής (Εχει προεπιλεχθεί ο g++).
- CFLAGS διάφορα ορίσματα που απαιτούνται απο τον μεταγλωττιστή που επιλέξαμε.
- F2C ο μετατροπέας προγραμμάτων Fortran77 σε γλώσσα C
- -debug στην περίπτωση που θέλουμε κατά την μεταγλώτισση των προγραμματων μας με τη χρήση του flag -g να γίνεται και η διαδικασία της αποσφαλματωσης.
- MPI_INCLUDE_DIRECTORY ο κατάλογος που περιέχει τα αρχεία κεφαλίδας του MPI.
- MPI_LIBRARY ο κατάλογος που περιέχει τα αρχεία των βιβλιοθηκών του MPI.

Αν δεν χρησιμοποιήσουμε τα flags MPI_INCLUDE_DIRECTORY και MPI_LIBRARY τότε εγκαθίσταται το λογισμικό μας μονο στην σειριακή του εκδοχή.

Στην συνέχεια πληκρολογούμε την εντολή make install και ορίζουμε στο path μας τον κατάλογο /paramenoas/man και με την εντολή

setenv MANPATH "$MANPATH" ":/home/username/paramenoas/man".

Στη συνέχεια ελέγχουμε αν εχει γίνει η διασύνδεση των βιβλιοθηκών του MPI libmpi.a,libsocket.a, libnsl.a και της βιβλιοθήκης libparMemetic.a Συνοψίζοντας δίνουμε τα παρακάτω δύο παραδείγματα εγκατάστασεων

Σειριακή Εγκατάσταση σε συστήματα Linux

```
tar xvf /home/username/paramenoas.tar
configure -arch linux -debug
make install
/home/username/paramenoas/examples/c/maxbit
```

Παράλληλη Εγκατάσταση σε συστήματα Linux

```
configure -arch linux -mpiinc /usr/local/mpi/include \
          -mpilib /usr/local/mpi/lib/libmpi.a
make install
mpirun -np 18 /home/username/paramenoas/examples/c/maxbit
```

Α΄.2 Ένα απλό πρόγραμμα στον PARAMENOAS

Α΄.2.1 Σειριακή Εκδοχή

```
  #include "paramenoas.h"
  double evaluate (PMAContext *dma, int p, int pop);
  int main(int argc, char **argv)
  {
   PMAContext *dma;
dma = PMACreate(&argc,argv, PMA_DATATYPE_BINARY,100, PMA_MAXIMIZE);
PMASetUserFunction(dma, PMA_USERFUNCTION_LOCALSEARCH, GuidedLocalSearch);
PMASetup(dma);
PMAEvaluate(dma, PMA_OLDPOP,evaluate, NULL);
PMAFitness(dma,PMA_OLDPOP);
while(!PMADone(dma,NULL)){
PMASelect (dma, PMA_OLDPOP);
PMARunMutationAndCrossover(dma, PMA_OLDPOP,PMA_NEWPOP);
PMAEvaluate(dma, PMA_NEWPOP, evaluate, NULL);
PMAFitness(dma, PMA_NEWPOP);
PMAUpdateGeneration(dma, NULL);
PMAPrintReport(dma, stdout, PMA_OLDPOP);
}
PMADestroy(dma);
return(0);
}
```

Α΄.2.2 Παράλληλη Εκδοχή

```
  #include "paramenoas.h"
  double evaluate (PMAContext *dma, int p, int pop);
  int main(int argc, char **argv)
  {

   PMAContext *dma;
int rank, numislands;
MPI_Init(&argc,&argv);
MPI_Comm_rank(MPI_COMM_WORLD, &rank);
dma = PMACreate(&argc,argv, PMA_DATATYPE_BINARY,100, PMA_MINIMIZE);
PMASetUserFunction(dma, PMA_USERFUNCTION_LOCALSEARCH, GuidedLocalSearch);
if (rank == 0){
```

```
printf("Number of islands?");
scanf("%d", &numislands);
}
MPI_Bcast(&numislands, 1, MPI_INT,0, MPI_COMM_WORLD);
PMASetup(dma);

PMAEvaluate(dma, PMA_OLDPOP,evaluate, NULL);
if ( rank ==0 )

PMAFitness(dma,PMA_OLDPOP);

while(!PMADone(dma,NULL)){
if (rank ==0 ) {
PMASelect (dma, PMA_OLDPOP);
PMARunMutationAndCrossover(dma, PMA_OLDPOP,PMA_NEWPOP);
}
PMAEvaluate(dma, PMA_NEWPOP, evaluate, NULL);
if (rank == 0)
PMAFitness(dma, PMA_NEWPOP);


PMAUpdateGeneration(dma, NULL);

if (rank ==0)
PMAPrintReport(dma, stdout, PMA_OLDPOP);
}
PMADestroy(dma);
MPI_Finalize();
return(0);
}
```

Α΄.3 Κυρίως πρόγραμμα για τα μαθηματικά προβλήματα της διατριβής PARAMENOAS

```
 #include <paramenoas.h>
#include <objfuncion.h>
#include <mpi.h>
#include <mpe.h>
```

```
#ifndef M_PI
#define M_PI 3.14159265354
#endif


void   printResultInterpretation(PMAContext *, int);

int    NumCoords[3]  = { 10, 20, 10 };
double Lower[3]      = { -512.0, -5.12, -512.0 };
double Upper[3]      = { 511.0, 5.11, 511.0 };

/*****************************************************************
 *                   user main program                           *
 *****************************************************************/
void main( int argc, char **argv )
{
    int testnum;         /* the DeJong test to run */
PMAContext *dma;
    int maxiter;         /* the maximum number of iterations */
    int rank;
    double l[20], u[20]; /* for initializing lu ranges */
    int i;
    MPI_Init(&argc, &argv);

testnum = 1;
maxiter = 100;
    for (i=0; i<20; i++) {
l[i] = Lower[testnum];
u[i] = Upper[testnum];
     }



    dma = PMACreate(&argc, argv, PMA_DATATYPE_REAL,
    NumCoords[testnum], PMA_MINIMIZE);

    PMASetRandomSeed(dma, 1);

    PMASetRealInitRange(dma, l, u);
    PMASetMaxGAIterValue(dma, maxiter);

  PMASetPrintOptions(dma, PMA_REPORT_OFFLINE);
```

```
    PMASetPrintOptions(dma, PMA_REPORT_STRING);
    PMASetPrintOptions(dma, PMA_REPORT_WORST);
    PMASetPrintOptions(dma, PMA_REPORT_AVERAGE);
    PMASetPrintOptions(dma, PMA_REPORT_ONLINE);

   PMASetCommunicator(dma, MPI_COMM_WORLD);
    PMASetUp(dma);

MPE_Init_log();
  MPI_Comm_rank(MPI_COMM_WORLD, &rank);
    if (rank == 0) {
MPE_Describe_state(1, 2, "Broadcast", "red:vlines3");
        MPE_Describe_state(3, 4, "Receive", "blue:gray3");
MPE_Describe_state(5, 6, "Send",    "green:light_gray");
MPE_Describe_state(7, 8, "Compute",      "yellow:gray");

    }
MPE_Start_log();
      cout << "Starting example program" << endl;
        MA_Initializer start;
        MA_Problem p;

\\ the starter is used to read the Configurationfile SA.cfg and to set up
\\ the choosen solver
        MA_Solver *sp = start.ReadConfigFile(argc,argv,p);
        if (  sp != NULL )
        {

sp->Runalgorithm();     \\ Run the algorithm selected via the configuration file
                  if (testnum == 1)    PMARun(dma, f1,   MPI_COMM_WORLD);
    if (testnum == 2)    PMARun(dma, f2,   MPI_COMM_WORLD);
    if (testnum == 3)    PMARun(dma, f3,   MPI_COMM_WORLD);
    if (testnum == 4)    PMARun(dma, f4,   MPI_COMM_WORLD);
    if (testnum == 5)    PMARun(dma, f5,   MPI_COMM_WORLD);
    if (testnum == 6)    PMARun(dma, f6,   MPI_COMM_WORLD);
    if (testnum == 7)    PMARun(dma, f7,   MPI_COMM_WORLD);
    if (testnum == 8)    PMARun(dma, f8,   MPI_COMM_WORLD);
    if (testnum == 9)    PMARun(dma, f9,   MPI_COMM_WORLD);
    if (testnum == 10)    PMARun(dma, f10,   MPI_COMM_WORLD);
    if (testnum == 11)    PMARun(dma, f11,   MPI_COMM_WORLD);
    if (testnum == 12)    PMARun(dma, f12,   MPI_COMM_WORLD);
    if (testnum == 13)    PMARun(dma, f13,   MPI_COMM_WORLD);
    if (testnum == 14)    PMARun(dma, f14,   MPI_COMM_WORLD);
```

```
delete sp;
        }



    PMADestroy(dma);



    MPI_Finalize();
}}
```

MPI_Datatype	PMABuildDatatype(PMAContext *dma, int p, int pop)
void	PMAChange(PMAContext *dma, int p, int pop)
int	PMACheckStoppingConditions(PMAContext *dma)
int	PMACheckSum(PMAContext *dma, int p, int pop)
void	PMAClearDebugLevel(PMAContext *dma, int level)
void	PMAClearDebugLevelByName(PMAContext *dma, char *funcname)
void	PMACopyIndividual(PMAContext *dma, int pi, int popl, int p2, int pop2)
PMAContext *	PMACreate(int *argc, char **argv, int datatype, int len, int maxormin)
void	PMACrossover(PMAContext *dma, int pi, int p2, int popl, int cl, int c2, int pop2)
void	PMADebugPrint(PMAContext *dma, int level, char *funcname, char *msg, int datatype, void *data)
void	PMADestroy(PMAContext *dma)
int	PMADone(PMAContext *dma, MPLComm comm)
int	PMADuplicate(PMAContext *dma, int p, int popl, int pop2, int n)
void	PMAEncodeIntegerAsBinary(PMAContext *dma, int p, int pop, int start, int end, int val)
void	PMAEncodeIntegerAsGrayCode(PMAContext *dma, int p, int pop, int start, int end, int val)
void	PMAEncodeRealAsBinary (PMAContext *dma, int p, int pop, int start, int end, double low, double high, double val)
void	PMAEncodeRealAsGrayCode (PMAContext *dma, int p, int int start, int end, double low, double high, double val)
void	PMAError(PMAContext *dma, char *msg, int level, int datatype, void *data)
void	PMAEvaluate(PMAContext *dma, int pop, double(*f) (PMAContext *, int, int), MPLComm comm)
void	PMAFitness(PMAContext *dma, int popindex)
int	PMAGetBestIndex(PMAContext *dma, int pop)
int	PMAGetBinaryAllele(PMAContext *dma, int p, int pop, int i)
double	PMAGetBinarylnitProb (PMAContext *dma)
char	PMAGetCharacterAllele(PMAContext *dma, int p, int pop, int i)
MPI_Comm	PMAGetCommunicator(PMAContext *dma)
double	PMAGetCrossoverProb (PMAContext *dma)
int	PMAGetCrossoverType(PMAContext *dma)
int	PMAGetDataType(PMAContext *dma)
double	PMAGetEvaluation(PMAContext *dma, int p, int pop)
int	PMAGetEvaluationUpToDateFlag(PMAContext *dma, int p, int pop)
double	PMAGetFitness(PMAContext *dma, int p, int pop)
double	PMAGetFitnessCmaxValue(PMAContext *dma)
int	PMAGetFitnessMinType(PMAContext *dma)
int	PMAGetFitnessType(PMAContext *dma)
int	PMAGetGAIterValue(PMAContext *dma)
int	PMAGetIntegerAllele(PMAContext *dma, int p, int pop, int i)
int	PMAGetIntegerFromBinary(PMAContext *dma, int p, int pop, int start, int end)
int	PMAGetIntegerFromGrayCode(PMAContext *dma, int p, int pop, int start, int end)
int	PMAGetIntegerInitType(PMAContext *dma)
double	PMAGetMaxFitnessRank(PMAContext *dma)
int	PMAGetMaxGAIterValue(PMAContext *dma)
int	PMAGetMaxIntegerInitValue(PMAContext *dma, int i)
double	PMAGetMaxMachineDoubleValue(PMAContext *dma)
int	PMAGetMaxMachineIntValue(PMAContext *dma)
double	PMAGetMaxRealInitValue(PMAContext *dma, int i)
int	PMAGetMinIntegerInitValue(PMAContext *dma, int i)
double	PMAGetMinMachineDoubleValue(PMAContext *dma)
int	PMAGetMinMachineIntValue(PMAContext *dma)
double	PMAGetMinRealInitValue(PMAContext *dma, int i)
int	PMAGetMutationAndCrossoverFlag(PMAContext *dma)
int	PMAGetMutationBoundedFlag(PMAContext *dma)
int	PMAGetMutationIntegerValue(PMAContext *dma)
int	PMAGetMutationOrCrossoverFlag(PMAContext *dma)
double	PMAGetMutationProb (PMAContext *dma)
double	PMAGetMutationRealValue(PMAContext *dma)
int	PMAGetMutationType(PMAContext *dma)

Πίνακας Α΄.1: Τύποι και συναρτήσεις του PARAMENOAS

MPI_Datatype	PMABuildDatatype(PMAContext *dma, int p, int pop)
void	PMAChange(PMAContext *dma, int p, int pop)
int	PMACheckStoppingConditions(PMAContext *dma)
int	PMACheckSum(PMAContext *dma, int p, int pop)
void	PMAClearDebugLevel(PMAContext *dma, int level)
void	PMAClearDebugLevelByName(PMAContext *dma, char *funcname)
void	PMACopyIndividual(PMAContext *dma, int pi, int popl, int p2, int pop2)
PMAContext *	PMACreate(int *argc, char **argv, int datatype, int len, int maxormin)
void	PMACrossover(PMAContext *dma, int pi, int p2, int popl, int cl, int c2, int pop2)
void	PMADebugPrint(PMAContext *dma, int level, char *funcname, char *msg, int datatype, void *data)
void	PMADestroy(PMAContext *dma)
int	PMADone(PMAContext *dma, MPLComm comm)
int	PMADuplicate(PMAContext *dma, int p, int popl, int pop2, int n)
void	PMAEncodeIntegerAsBinary(PMAContext *dma, int p, int pop, int start, int end, int val)
void	PMAEncodeIntegerAsGrayCode(PMAContext *dma, int p, int pop, int start, int end, int val)
void	PMAEncodeRealAsBinary (PMAContext *dma, int p, int pop, int start, int end, double low, double high, double val)
void	PMAEncodeRealAsGrayCode (PMAContext *dma, int p, int int start, int end, double low, double high, double val)
void	PMAError(PMAContext *dma, char *msg, int level, int datatype, void *data)
void	PMAEvaluate(PMAContext *dma, int pop, double(*f) (PMAContext *, int, int), MPLComm comm)
void	PMAFitness(PMAContext *dma, int popindex)
int	PMAGetBestIndex(PMAContext *dma, int pop)
int	PMAGetBinaryAllele(PMAContext *dma, int p, int pop, int i)
double	PMAGetBinarylnitProb (PMAContext *dma)
char	PMAGetCharacterAllele(PMAContext *dma, int p, int pop, int i)
MPI_Comm	PMAGetCommunicator(PMAContext *dma)
double	PMAGetCrossoverProb (PMAContext *dma)
int	PMAGetCrossoverType(PMAContext *dma)
int	PMAGetDataType(PMAContext *dma)
double	PMAGetEvaluation(PMAContext *dma, int p, int pop)
int	PMAGetEvaluationUpToDateFlag(PMAContext *dma, int p, int pop)
double	PMAGetFitness(PMAContext *dma, int p, int pop)
double	PMAGetFitnessCmaxValue(PMAContext *dma)
int	PMAGetFitnessMinType(PMAContext *dma)
int	PMAGetFitnessType(PMAContext *dma)
int	PMAGetGAIterValue(PMAContext *dma)
int	PMAGetIntegerAllele(PMAContext *dma, int p, int pop, int i)
int	PMAGetIntegerFromBinary(PMAContext *dma, int p, int pop, int start, int end)
int	PMAGetIntegerFromGrayCode(PMAContext *dma, int p, int pop, int start, int end)
int	PMAGetIntegerInitType(PMAContext *dma)
double	PMAGetMaxFitnessRank(PMAContext *dma)
int	PMAGetMaxGAIterValue(PMAContext *dma)
int	PMAGetMaxIntegerInitValue(PMAContext *dma, int i)
double	PMAGetMaxMachineDoubleValue(PMAContext *dma)
int	PMAGetMaxMachineIntValue(PMAContext *dma)
double	PMAGetMaxRealInitValue(PMAContext *dma, int i)
int	PMAGetMinIntegerInitValue(PMAContext *dma, int i)
double	PMAGetMinMachineDoubleValue(PMAContext *dma)
int	PMAGetMinMachineIntValue(PMAContext *dma)
double	PMAGetMinRealInitValue(PMAContext *dma, int i)
int	PMAGetMutationAndCrossoverFlag(PMAContext *dma)
int	PMAGetMutationBoundedFlag(PMAContext *dma)
int	PMAGetMutationIntegerValue(PMAContext *dma)
int	PMAGetMutationOrCrossoverFlag(PMAContext *dma)
double	PMAGetMutationProb (PMAContext *dma)
double	PMAGetMutationRealValue(PMAContext *dma)
int	PMAGetMutationType(PMAContext *dma)

Πίνακας Α΄.2: Τύποι και συναρτήσεις του PARAMENOAS

int	PMAGetNoDuplicatesFlag(PMAContext *dma)
int	PMAGetNumProcs(PMAContext *dma, MPLComm comm)
int	PMAGetNumReplaceValue(PMAContext *dma)
int	PMAGetOptDirFlag(PMAContext *dma)
double	PMAGetPTournamentProb(PG AContext *dma)
int	PMAGetPopReplaceType(PMAContext *dma)
int	PMAGetPopSize(PMAContext *dma)
int	PMAGetPrintFrequencyValue(PMAContext *dma)
int	PMAGetRandomInitFlag(PMAContext *dma)
int	PMAGetRandomSeed(PMAContext *dma)
int	PMAGetRank(PMAContext *dma, MPLComm comm)
double	PMAGetRealAllele(PMAContext *dma, int p, int pop, int i)
double	PMAGetRealFromBinary(PG AContext *dma, int p, int pop, int start, int end, double lower, double upper)
double	PMAGetRealFromGrayCode(PG AContext *dma, int p, int pop, int start, int end, double lower, double upper)
int	PMAGetRealInitType(PMAContext *dma)
double	PMAGetRestart AlleleChangeProb (PG AContext *dma)
int	PMAGetRestartFlag(PMAContext *dma)
int	PMAGetRestartFrequencyValue(PMAContext *dma)
int	PMAGetSelectType(PMAContext *dma)
int	PMAGetSortedPopIndex(PMAContext *dma, int n)
int	PMAGetStoppingRuleType(PMAContext *dma)
int	PMAGetStringLength(PMAContext *dma)
double	PMAGetUniformCrossoverProb (PG AContext *dma)
int	PMAGetWorstIndex(PMAContext *dma, int pop)
double	PMAHammingDistance(PMAContext *dma, int popindex)
double	PMAMean(PMAContext *dma, double *a, int n)
int	PMAMutate(PMAContext *dma, int p, int pop)
void	PMAPrintContextVariable(PMAContext *dma, FILE *fp)
void	PMAPrintIndividual(PMAContext *dma, FILE *fp, int p, int pop)
void	PMAPrintPopulation(PG AContext *dma, FILE *fp, int pop)
void	PMAPrintReport(PMAContext *dma, FILE *fp, int pop)
void	PMAPrintString(PMAContext *dma, FILE *file, int p, int pop)
void	PMAPrintVersionNumber(PMAContext *dma)
double	PMARandom01(PMAContext *dma, int newseed)
int	PMARandomFlip(PMAContext *dma, double p)
double	PMARandomGaussian(PMAContext *dma, double mean, double sigma)
int	PMARandomInterval(PMAContext *dma, int start, int end)
double	PMARandomUniform(PMAContext *dma, double start, double end)
int	PMARank(PMAContext *dma, int p, int *order, int n)
void	PMAReceiveIndividual(PMAContext *dma, int p, int pop, int source, int tag, MPI_Comm comm, MPI_Status *status)
void	PMARestart(PMAContext *dma, int source-pop, int dest$_p op$)
Int	PMARound(PMAContext *dma, double x)
void	PMARun(PMAContext *dma, double(*evaluate)(PMAContext *c, int p, int pop))
void	PMARunGM(PMAContext *dma, double(*f)(PMAContext *, int, int), MPLComm comrn)
void	PMARunMutationAndCrossover(PMAContext *dma, int oldpop, int newpop)
void	PMARunMutationOrCrossover(PMAContext *dma, int oldpop, int newpop)
void	PMASelect(PMAContext *dma, int popix)

Πίνακας Α′.3: Τύποι και συναρτήσεις του PARAMENOAS

int	PMAGetNoDuplicatesFlag(PMAContext *dma)
int	PMAGetNumProcs(PMAContext *dma, MPLComm comm)
int	PMAGetNumReplaceValue(PMAContext *dma)
int	PMAGetOptDirFlag(PMAContext *dma)
double	PMAGetPTournamentProb(PG AContext *dma)
int	PMAGetPopReplaceType(PMAContext *dma)
int	PMAGetPopSize(PMAContext *dma)
int	PMAGetPrintFrequencyValue(PMAContext *dma)
int	PMAGetRandomInitFlag(PMAContext *dma)
int	PMAGetRandomSeed(PMAContext *dma)
int	PMAGetRank(PMAContext *dma, MPLComm comm)
double	PMAGetRealAllele(PMAContext *dma, int p, int pop, int i)
double	PMAGetRealFromBinary(PG AContext *dma, int p, int pop, int start, int end, double lower, double upper)
double	PMAGetRealFromGrayCode(PG AContext *dma, int p, int pop, int start, int end, double lower, double upper)
int	PMAGetRealInitType(PMAContext *dma)
double	PMAGetRestart AlleleChangeProb (PG AContext *dma)
int	PMAGetRestartFlag(PMAContext *dma)
int	PMAGetRestartFrequencyValue(PMAContext *dma)
int	PMAGetSelectType(PMAContext *dma)
int	PMAGetSortedPopIndex(PMAContext *dma, int n)
int	PMAGetStoppingRuleType(PMAContext *dma)
int	PMAGetStringLength(PMAContext *dma)
double	PMAGetUniformCrossoverProb (PG AContext *dma)
int	PMAGetWorstIndex(PMAContext *dma, int pop)
double	PMAHammingDistance(PMAContext *dma, int popindex)
double	PMAMean(PMAContext *dma, double *a, int n)
int	PMAMutate(PMAContext *dma, int p, int pop)
void	PMAPrintContextVariable(PMAContext *dma, FILE *fp)
void	PMAPrintIndividual(PMAContext *dma, FILE *fp, int p, int pop)
void	PMAPrintPopulation(PG AContext *dma, FILE *fp, int pop)
void	PMAPrintReport(PMAContext *dma, FILE *fp, int pop)
void	PMAPrintString(PMAContext *dma, FILE *file, int p, int pop)
void	PMAPrintVersionNumber(PMAContext *dma)
double	PMARandom01(PMAContext *dma, int newseed)
int	PMARandomFlip(PMAContext *dma, double p)
double	PMARandomGaussian(PMAContext *dma, double mean, double sigma)
int	PMARandomInterval(PMAContext *dma, int start, int end)
double	PMARandomUniform(PMAContext *dma, double start, double end)
int	PMARank(PMAContext *dma, int p, int *order, int n)
void	PMAReceiveIndividual(PMAContext *dma, int p, int pop, int source, int tag, MPI_Comm comm, MPI_Status *status)
void	PMARestart(PMAContext *dma, int source-pop, int dest$_p op$)
Int	PMARound(PMAContext *dma, double x)
void	PMARun(PMAContext *dma, double(*evaluate)(PMAContext *c, int p, int pop))
void	PMARunGM(PMAContext *dma, double(*f)(PMAContext *, int, int), MPLComm comrn)
void	PMARunMutationAndCrossover(PMAContext *dma, int oldpop, int newpop)
void	PMARunMutationOrCrossover(PMAContext *dma, int oldpop, int newpop)
void	PMASelect(PMAContext *dma, int popix)

Πίνακας Α΄.4: Τύποι και συναρτήσεις του PARAMENOAS

int	PMASelectNextIndex(PMAContext *dma)
void	PMASendIndividual(PMAContext *dma, int p, int pop, int dest, int tag, MPLComm comm)
void	PMASendReceiveIndividual(PMAContext *dma, int send_p, int send_pop, int dest, int send_tag, int recv_p, int recv_pop, int source, int recv_tag, MPLComm comm, MPI_Status *status)
void	PMASetBinaryAllele(PMAContext *dma, int p, int pop, int i, int val)
void	PMASetBinaryInitProb(PMAContext *dma, double probability)
void	PMASetCharacterAllele(PMAContext *dma, int p, int pop, int i, char value)
void	PMASetCharacterInitType(PMAContext *dma, int value)
void	PMASetCommunicator(PMAContext *dma, MPLComm comm)
void	PMASetCrossoverProb(PMAContext *dma, double crossover_prob)
void	PMASetCrossoverType(PMAContext *dma, int crossover_type)
void	PMASetDebugLevel(PMAContext *dma, int level)
void	PMASetDebugLevelByName(PMAContext *dma, char *funcname)
void	PMASetEvaluation(PMAContext *dma, int p, int pop, double val)
void	PMASetEvaluationUpToDateFlag(PMAContext *dma, int p, int pop, int status)
void	PMASetFitnessCmaxValue(PMAContext *dma, double val)
void	PMASetFitnessMinType(PMAContext *dma, int fitness$_type$)
void	PMASetFitnessType(PMAContext *dma, int fitness$_type$)
void	PMASetIntegerAllele(PMAContext *dma, int p, int pop, int i, int value)
void	PMASetIntegerInitPermute(PMAContext *dma, int min, int max)
void	PMASetIntegerInitRange(PMAContext *dma, int *min, int *max)
void	PMASetMaxFitnessRank(PMAContext *dma, double fitness$_rank_max$)
void	PMASetMaxGAIterValue(PMAContext *dma, int maxiter)
void	PMASetMaxNoChangeValue(PMAContext *dma, int max$_no_change$)
void	PMASetMaxSimilarityValue(PMAContext *dma, int max$_similarity$)
void	PMASetMutationAndCrossoverFlag(PMAContext *dma, int flag)
void	PMASetMutationBoundedFlag(PMAContext *dma, int val)
void	PMASetMutationIntegerValue(PMAContext *dma, int val)
void	PMASetMutationOrCrossoverFlag(PMAContext *dma, int flag)
void	PMASetMutationProb(PMAContext *dma, double mutation$_prob$)
void	PMASetMutationRealValue(PMAContext *dma, double val)
void	PMASetMutationType(PMAContext *dma, int mutation-type)
void	PMASetNoDuplicatesFlag(PMAContext *dma, int no$_dup$)
void	PMASetNumReplaceValue(PMAContext *dma, int pop-replace)
void	PMASetPTournamentProb(PMAContext *dma, double ptournament$_prob$)
void	PMASetPopReplaceType(PMAContext *dma, int pop-replace)
void	PMASetPopSize(PMAContext *dma, int popsize)
void	PMASetPrintFrequencyValue(PMAContext *dma, int print$_freq$)
void	PMASetPrintOptions(PMAContext *dma, int option)
void	PMASetRandomInitFlag(PMAContext *dma, int RandomBoolean)
void	PMASetRandomSeed(PMAContext *dma, int seed)
void	PMASetRealAllele(PMAContext *dma, int p, int pop, int i, double value)
void	PMASetRealInitPercent(PMAContext *dma, double *median, double *percent)
void	PMASetRealInitRange(PMAContext *dma, double *min, double *max)
void	PMASetRestartAlleleChangeProb(PMAContext *dma, double prob)
void	PMASetRestartFlag(PMAContext *dma, int val)
void	PMASetRestartFrequencyValue(PMAContext *dma, int numiter)
void	PMASetSelectType(PMAContext *dma, int select-type)
void	PMASetStoppingRuleType(PMAContext *dma, int stoprule)
void	PMASetUniformCrossoverProb(PMAContext *dma, double uniform$_cross_prob$)
void	PMASetUp(PMAContext *dma)
void	PMASetUserFunction(PMAContext *dma, int constant, void *f)
void	PMASortPop(PMAContext *dma, int pop)
double	PMAStddev(PMAContext *dma, double *a, int n, double mean)
void	PMAUpdateGeneration(PMAContext *dma, MPLComm comm)
void	PMAUsage(PMAContext *dma)

Πίνακας Α΄.5: Τύποι και συναρτήσεις του PARAMENOAS

int	PMASelectNextIndex(PMAContext *dma)
void	PMASendIndividual(PMAContext *dma, int p, int pop, int dest, int tag, MPLComm comm)
void	PMASendReceiveIndividual(PMAContext *dma, int send_p, int send_pop, int dest, int send_tag, int recv_p, int recv_pop, int source, int recv_tag, MPLComm comm, MPI_Status *status)
void	PMASetBinaryAllele(PMAContext *dma, int p, int pop, int i, int val)
void	PMASetBinaryInitProb(PMAContext *dma, double probability)
void	PMASetCharacterAllele(PMAContext *dma, int p, int pop, int i, char value)
void	PMASetCharacterInitType(PMAContext *dma, int value)
void	PMASetCommunicator(PMAContext *dma, MPLComm comm)
void	PMASetCrossoverProb(PMAContext *dma, double crossover_prob)
void	PMASetCrossoverType(PMAContext *dma, int crossover_type)
void	PMASetDebugLevel(PMAContext *dma, int level)
void	PMASetDebugLevelByName(PMAContext *dma, char *funcname)
void	PMASetEvaluation(PMAContext *dma, int p, int pop, double val)
void	PMASetEvaluationUpToDateFlag(PMAContext *dma, int p, int pop, int status)
void	PMASetFitnessCmaxValue(PMAContext *dma, double val)
void	PMASetFitnessMinType(PMAContext *dma, int fitness$_type$)
void	PMASetFitnessType(PMAContext *dma, int fitness$_type$)
void	PMASetIntegerAllele(PMAContext *dma, int p, int pop, int i, int value)
void	PMASetIntegerInitPermute(PMAContext *dma, int min, int max)
void	PMASetIntegerInitRange(PMAContext *dma, int *min, int *max)
void	PMASetMaxFitnessRank(PMAContext *dma, double fitness$_rank_max$)
void	PMASetMaxGAIterValue(PMAContext *dma, int maxiter)
void	PMASetMaxNoChangeValue(PMAContext *dma, int max$_nochange$)
void	PMASetMaxSimilarityValue(PMAContext *dma, int max$_similarity$)
void	PMASetMutationAndCrossoverFlag(PMAContext *dma, int flag)
void	PMASetMutationBoundedFlag(PMAContext *dma, int val)
void	PMASetMutationIntegerValue(PMAContext *dma, int val)
void	PMASetMutationOrCrossoverFlag(PMAContext *dma, int flag)
void	PMASetMutationProb(PMAContext *dma, double mutation$_prob$)
void	PMASetMutationRealValue(PMAContext *dma, double val)
void	PMASetMutationType(PMAContext *dma, int mutation-type)
void	PMASetNoDuplicatesFlag(PMAContext *dma, int no$_dup$)
void	PMASetNumReplaceValue(PMAContext *dma, int pop-replace)
void	PMASetPTournamentProb(PMAContext *dma, double ptournament$_prob$)
void	PMASetPopReplaceType(PMAContext *dma, int pop-replace)
void	PMASetPopSize(PMAContext *dma, int popsize)
void	PMASetPrintFrequencyValue(PMAContext *dma, int print$_freq$)
void	PMASetPrintOptions(PMAContext *dma, int option)
void	PMASetRandomInitFlag(PMAContext *dma, int RandomBoolean)
void	PMASetRandomSeed(PMAContext *dma, int seed)
void	PMASetRealAllele(PMAContext *dma, int p, int pop, int i, double value)
void	PMASetRealInitPercent(PMAContext *dma, double *median, double *percent)
void	PMASetRealInitRange(PMAContext *dma, double *min, double *max)
void	PMASetRestartAlleleChangeProb(PMAContext *dma, double prob)
void	PMASetRestartFlag(PMAContext *dma, int val)
void	PMASetRestartFrequencyValue(PMAContext *dma, int numiter)
void	PMASetSelectType(PMAContext *dma, int select-type)
void	PMASetStoppingRuleType(PMAContext *dma, int stoprule)
void	PMASetUniformCrossoverProb(PMAContext *dma, double uniform$_cross_prob$)
void	PMASetUp(PMAContext *dma)
void	PMASetUserFunction(PMAContext *dma, int constant, void *f)
void	PMASortPop(PMAContext *dma, int pop)
double	PMAStddev(PMAContext *dma, double *a, int n, double mean)
void	PMAUpdateGeneration(PMAContext *dma, MPLComm comm)
void	PMAUsage(PMAContext *dma)

Πίνακας Α΄.6: Τύποι και συναρτήσεις του PARAMENOAS

Population size	100	PMASetPopSize
Copied for population replacement	PMA_POPREPL_BEST	PMASetPopReplacementType
Stopping rule	PMA_STOP_MAXITER	PMASetStoppingRuleType
Maximum iterations	1000	PMASetMaxGAIterValue
Maximum no change iters	100	PMASetMaxNoChangeValue
Max. population homogeneity before stopping	95	PMASetMaxSimilarityValue
Number of new strings to generate	10	PMASetNumReplaceValue
Apply mutation and crossover	PMA_FALSE	PMASetMutationAndCrossoverFlag
Apply mutation or crossover	PMA.TRUE	PMASetMutationOrCrossoverFlag
Crossover type	PMA_CROSSOVER_TiOPT	PMASetCrossoverType
Probability of crossover	0.85	PMASetCrossoverProb
Uniform crossover bias	0.6	PMASetUniformCrossoverProb
Mutation type (Real strings)	PMA_MUTATION_GAUSSIAN	PMASetMutationType
Mutation type (Integer strings)	PMA_MUTATION_PERMUTE	PMASetMutationType
Mutation type (Character strings)	Same as initialization	PMASetCharacterlnitType
Mutation probability	1/L	PMASetMutationProb
Real mutation constant	0.1	PMASetMutationRealValue
Integer mutation constant	1	PMASetMutationlntegerValue
Mutation range bounded	PMA.TRUE	PMASetMutationBoundedFlag
Select type	PMA_SELECT_TOURNAMENT	PMASetSelectType
Probabilistic binary tournament parameter	0.6	PMASetPTournamentProb
Use restart operator	PMA_FALSE	PMASetRestartFlag
Restart frequency	50	PMASetRestartFrequencyValue
Restart allele mutation rate	0.5	PMASetRestartAlleleChangeProb
Allow duplicate strings	PMA_FALSE	PMASetNoDuplicatesFlag
Fitness type	PMA_FITNESS_RAi	PMASetFitnessType
Fitness type for minimization	PMA_FITNESSMIN_CMAX	PMASetFitnessMinType
Multiplier for minimization problems	1.01	PMASetCMaxValue
Parameter MAX in fitness by ranking	1.2	PMASetMaxFitnessRank
Frequency of statistics printing	10	PMASetPrintFrequencyValue
Print strings	PMA_FALSE	PMASetPrintOptions
Print offline statistics	PMA_FALSE	PMASetPrintOptions
Print online statistics	PMA_FALSE	PMASetPrintOptions
Print best string	PMA_FALSE	PMASetPrintOptions
Print worst string	PMA_FALSE	PMASetPrintOptions
Print Hamming distance	PMA_FALSE	PMASetPrintOptions
Randomly initialize population	PMA_TRUE	PMASetRandomlnitFlag
Probability of initializing a bit to one	0.5	PMASetBinarylnitProb
How to initialize real strings	Range	PMASetreallnitRange
Real initialization range	[0,1]	PMASetReallnitRange
How to initialize integer strings	Permutation	PMASetIntegerlnitPermute
Integer initialization range	[0,L_1]	PMASetIntegerlnitPermute
How to initialize character strings	PMA_CINIT_LOiER	PMASetCharacterlnitFlag
Seed random number with clock	PMA_TRUE	PMASetRandomSeed
Default MPI communicator	MPI_COMM_WORLD	PMASetCommunicator
L is the string length		

Πίνακας Α′.7: Προκαθορισμένες τιμές για τον PARAMENOAS

Population size	100	PMASetPopSize
Copied for population replacement	PMA_POPREPL_BEST	PMASetPopReplacementType
Stopping rule	PMA_STOP_MAXITER	PMASetStoppingRuleType
Maximum iterations	1000	PMASetMaxGAIterValue
Maximum no change iters	100	PMASetMaxNoChangeValue
Max. population homogeneity before stopping	95	PMASetMaxSimilarityValue
Number of new strings to generate	10	PMASetNumReplaceValue
Apply mutation and crossover	PMA_FALSE	PMASetMutationAndCrossoverFlag
Apply mutation or crossover	PMA.TRUE	PMASetMutationOrCrossoverFlag
Crossover type	PMA_CROSSOVER_TiOPT	PMASetCrossoverType
Probability of crossover	0.85	PMASetCrossoverProb
Uniform crossover bias	0.6	PMASetUniformCrossoverProb
Mutation type (Real strings)	PMA_MUTATION_GAUSSIAN	PMASetMutationType
Mutation type (Integer strings)	PMA_MUTATION_PERMUTE	PMASetMutationType
Mutation type (Character strings)	Same as initialization	PMASetCharacterlnitType
Mutation probability	1/L	PMASetMutationProb
Real mutation constant	0.1	PMASetMutationRealValue
Integer mutation constant	1	PMASetMutationlntegerValue
Mutation range bounded	PMA.TRUE	PMASetMutationBoundedFlag
Select type	PMA_SELECT_TOURNAMENT	PMASetSelectType
Probabilistic binary tournament parameter	0.6	PMASetPTournamentProb
Use restart operator	PMA_FALSE	PMASetRestartFlag
Restart frequency	50	PMASetRestartFrequencyValue
Restart allele mutation rate	0.5	PMASetRestartAlleleChangeProb
Allow duplicate strings	PMA_FALSE	PMASetNoDuplicatesFlag
Fitness type	PMA_FITNESS_RAi	PMASetFitnessType
Fitness type for minimization	PMA_FITNESSMIN_CMAX	PMASetFitnessMinType
Multiplier for minimization problems	1.01	PMASetCMaxValue
Parameter MAX in fitness by ranking	1.2	PMASetMaxFitnessRank
Frequency of statistics printing	10	PMASetPrintFrequencyValue
Print strings	PMA_FALSE	PMASetPrintOptions
Print offline statistics	PMA_FALSE	PMASetPrintOptions
Print online statistics	PMA_FALSE	PMASetPrintOptions
Print best string	PMA_FALSE	PMASetPrintOptions
Print worst string	PMA_FALSE	PMASetPrintOptions
Print Hamming distance	PMA_FALSE	PMASetPrintOptions
Randomly initialize population	PMA_TRUE	PMASetRandomlnitFlag
Probability of initializing a bit to one	0.5	PMASetBinarylnitProb
How to initialize real strings	Range	PMASetreallnitRange
Real initialization range	[0,1]	PMASetReallnitRange
How to initialize integer strings	Permutation	PMASetlntegerlnitPermute
Integer initialization range	[0,L_1]	PMASetlntegerlnitPermute
How to initialize character strings	PMA_CINIT_LOiER	PMASetCharacterlnitFlag
Seed random number with clock	PMA_TRUE	PMASetRandomSeed
Default MPI communicator	MPI_COMM_WORLD	PMASetCommunicator
L is the string length		

Πίνακας Α΄.8: Προκαθορισμένες τιμές για τον PARAMENOAS

Παράρτημα Β′

Κώδικας εφαρμογών και Δεδομένα

Β′.1 Χρονοπρογραμματισμός παραγωγής σταθμών ηλ. ενέργειας

```
#define MAX_NUM_DEHSTATIONS     5  /* Maximum number of DEH Electical_Stations. */
#define MAX_NUM_JOB_TYPES     3  /* Maximum number of jobsynthrisis  types. */

/* Declare non-simlib global variables. */

#include <stdio.h>
#include <mpi.h>

#include <paramenoas.h>
#include <math.h>
#include <stdlib.h>
#include <sys/time.h>
#include <assert.h>
#include <memory.h>
#include <stdio.h>

extern void ftiaxe_graphos(int, int);

#define NXPROB      20
#define NYPROB      20
```

```
#define STEPS       50                /* number of time steps */
#define MAXWORKER   8                 /* maximum number of worker tasks */
#define MINWORKER   3                 /* minimum number of worker tasks */
#define BEGIN       1                 /* message type */
#define Graphos1     2                 /* message type */
#define Graphos2     3                 /* message type */
#define NONE        0                 /* indicates no graphos */
#define DONE        4                 /* message type */
#define MASTER      0                 /* taskid of first process */

struct Parms {
  float cx;
  float cy;
} alloo = {0.1, 0.1};

int main(argc,argv)
int argc;
char *argv[];
{
void proteraiothta_synthrisis(), prtdat(), update();
float  u[2][NXPROB][NYPROB];        /* array for grid */
int taskid,                     /* this task's unique id */
numworkers,                 /* number of worker processes */
numtasks,                   /* number of tasks */
averow,rows,offset,extra,   /* for sending rows of data */
dest, source,               /* to - from for message send-receive */
graphos1,graphos2,        /* graphos tasks */
msgtype,                    /* for message types */
rc,start,end,               /* misc */
i,kaysimox,dapanesgiay,iz,it;             /* loop variables */
MPI_Status status;




/**************************************************************************
 *  subroutine update
 **************************************************************************/
void update(int start, int end, int ny, float *u1, float *u2)
{
   int kaysimox, dapanesgiay;
   for (kaysimox = start; kaysimox <= end; kaysimox++)
      for (dapanesgiay = 1; dapanesgiay <= ny-2; dapanesgiay++)
```

```
         *(u2+kaysimox*ny+dapanesgiay) =
 *(u1+kaysimox*ny+dapanesgiay)  +
                          alloo.cx * (*(u1+(kaysimox+1)*ny+dapanesgiay) +
                          *(u1+(kaysimox-1)*ny+dapanesgiay) -
                          2.0 * *(u1+kaysimox*ny+dapanesgiay)) +
                          alloo.cy * (*(u1+kaysimox*ny+dapanesgiay+1) +
                         *(u1+kaysimox*ny+dapanesgiay-1) -
                          2.0 * *(u1+kaysimox*ny+dapanesgiay));
}


void proteraiothta_synthrisis(int nx, int ny, float *u) {
int kaysimox, dapanesgiay;

for (kaysimox = 0; kaysimox <= nx-1; kaysimox++)
  for (dapanesgiay = 0; dapanesgiay <= ny-1; dapanesgiay++)
     *(u+kaysimox*ny+dapanesgiay) = (float)(kaysimox * (nx - kaysimox
     - 1) * dapanesgiay * (ny - dapanesgiay - 1));
}



void server(void);
void client(void);
int rank, size;
int main(int argc, char **argv)
{
int i;
int err;
C aC;
int lenc[2];
MPI_Aint locc[2];
MPI_Datatype typc[2];
MPI_Aint baseaddr;
\\ Initialize MPI
err = MPI_Init(&argc, &argv);
err = MPI_Comm_rank(MPI_COMM_WORLD, &rank);
err = MPI_Comm_size(MPI_COMM_WORLD, &size);
\\ Make the C MPI type
MPI_Address(&aC, &baseaddr);
lenc[0] = 1;
MPI_Address(&aC.longitude, &locc[0]);
locc[0] -= baseaddr;
```

```
typc[0] = MPI_FLOAT;
lenc[1] = 1;
MPI_Address(&aC.latitude, &locc[1]);
locc[1] -= baseaddr;
typc[1] = MPI_FLOAT;
MPI_Type_struct(2, lenc, locc, typc, &MPI_C);
MPI_Type_commit(&MPI_C);
/* First, find out my taskid and how many tasks are running */
   rc = MPI_Init(&argc,&argv);
   rc= MPI_Comm_size(MPI_COMM_WORLD,&numtasks);
   rc= MPI_Comm_rank(MPI_COMM_WORLD,&taskid);
   if (rc != 0)
      printf ("error initializing MPI and obtaining task ID information\n");
   else
      printf ("mpi_heat2D MPI task ID = %d\n", taskid);
   numworkers = numtasks-1;

   if (taskid == MASTER)
   {
      /************************ master code *******************************/
      /* Check if numworkers is within range - quit if not */
      if ((numworkers > MAXWORKER) || (numworkers <
       MINWORKER))
      {
         printf("MP_PROCS needs to be between %d and %d for this
 exercise\n",
                  MINWORKER+1,MAXWORKER+1);
         MPI_Finalize();
      }

      /* Initialize grid */
      printf("Grid size: X= %d  Y= %d  Time steps=
      %d\n",NXPROB,NYPROB,STEPS);
      printf("Initializing grid and writing initial.dat file...\n");
      proteraiothta_synthrisis(NXPROB, NYPROB, u);
      prtdat(NXPROB, NYPROB, u, "initial.dat");

      averow = NXPROB/numworkers;
      extra = NXPROB%numworkers;
      offset = 0;
      for (i=1; i<=numworkers; i++)
      {
         rows = (i <= extra) ? averow+1 : averow;
```

```
        if (i == 1)
           graphos1 = NONE;
        else
           graphos1 = i - 1;
        if (i == numworkers)
           graphos2 = NONE;
        else
           graphos2 = i + 1;
        /*  Now send startup information to each worker  */
        dest = i;
        MPI_Send(&offset, 1, MPI_INT, dest, BEGIN,
MPI_COMM_WORLD);
        MPI_Send(&rows, 1, MPI_INT, dest, BEGIN,
MPI_COMM_WORLD);
        MPI_Send(&graphos1, 1, MPI_INT, dest, BEGIN,
MPI_COMM_WORLD);
        MPI_Send(&graphos2, 1, MPI_INT, dest, BEGIN,
MPI_COMM_WORLD);
        MPI_Send(&u[0][offset][0], rows*NYPROB, MPI_FLOAT, dest,
BEGIN, MPI_COMM_WORLD);
        printf("Sent to= %d offset= %d rows= %d graphos1= %d
graphos2= %d\n",
               dest,offset,rows,graphos1,graphos2);
        offset = offset + rows;
     }
     /* Now wait for results from all worker tasks */
     for (i=1; i<=numworkers; i++)
     {
        source = i;
        msgtype = DONE;
        MPI_Recv(&offset, 1, MPI_INT, source, msgtype,
MPI_COMM_WORLD, &status);
        MPI_Recv(&rows, 1, MPI_INT, source, msgtype,
MPI_COMM_WORLD, &status);
        MPI_Recv(&u[0][offset][0], rows*NYPROB, MPI_FLOAT, source,
                  msgtype, MPI_COMM_WORLD, &status);
     }

     /* Write final output and call X graph */
     printf("Writing final.dat file and generating graph...\n");
     prtdat(NXPROB, NYPROB, &u[0][0][0], "final.dat");
     ftiaxe_graphos(NXPROB,NYPROB);
  }   /* End of master code */
```

```
if (taskid != MASTER)
{
   for (iz=0; iz<2; iz++)
      for (kaysimox=0; kaysimox<NXPROB; kaysimox++)
         for (dapanesgiay=0; dapanesgiay<NYPROB; dapanesgiay++)
            u[iz][kaysimox][dapanesgiay] = 0.0;


   source = MASTER;
   msgtype = BEGIN;
   MPI_Recv(&offset, 1, MPI_INT, source, msgtype,
   MPI_COMM_WORLD, &status);
   MPI_Recv(&rows, 1, MPI_INT, source, msgtype,
   MPI_COMM_WORLD, &status);
   MPI_Recv(&graphos1, 1, MPI_INT, source, msgtype,
   MPI_COMM_WORLD, &status);
   MPI_Recv(&graphos2, 1, MPI_INT, source, msgtype,
   MPI_COMM_WORLD, &status);
   MPI_Recv(&u[0][offset][0], rows*NYPROB, MPI_FLOAT, source,
   msgtype, MPI_COMM_WORLD, &status);

   if (offset==0)
      start=1;
   else
      start=offset;
   if ((offset+rows)==NXPROB)
      end=start+rows-2;
   else
      end = start+rows-1;


   iz = 0;
   for (it = 1; it <= STEPS; it++)
   {
      if (graphos1 != NONE)
      {
         MPI_Send(&u[iz][offset][0], NYPROB, MPI_FLOAT, graphos1,
                  Graphos2, MPI_COMM_WORLD);
         source = graphos1;
         msgtype = Graphos1;
         MPI_Recv(&u[iz][offset-1][0], NYPROB, MPI_FLOAT, source,
                  msgtype, MPI_COMM_WORLD, &status);
```

```
        }
        if (graphos2 != NONE)
        {
 or (i = 0; i < POPULATION_SIZE; i++)
dehjobsynthrisiss[i] = transfer(dehElectical_Stations[i]);
select_crossover_choices(crossover_choices, dehjobsynthrisiss);
\\ make newdehElectical_Stations
\\elitism
memcpy(newdehElectical_Stations[POPULATION_SIZE - 1],
dehElectical_Stations[POPULATION_SIZE - 1], sizeof(class));
for (i = 0; i < POPULATION_SIZE - 2; i++) {
docrossover(dehElectical_Stations[crossover_choices[2 * i]],
dehElectical_Stations[crossover_choices[2 * i + 1]],
newdehElectical_Stations[i], newdehElectical_Stations[i + 1]);
if ( LS == 1)
Guided_Local_Search(*dehElectical_Stations, POPULATION_cur,

  sizeof(dehElectical_Stations[0]), comparedehElectical_Stations);
if (LS == 2)
Simulated_Annealing(*dehElectical_Stations, POPULATION_cur,

 sizeof(dehElectical_Stations[0]), comparedehElectical_Stations);
if (LS == 3)
Tabu_search(*dehElectical_Stations, POPULATION_cur,

  sizeof(dehElectical_Stations[0]), comparedehElectical_Stations);
}
    jobsynthrisis_type = transfer[3];
    task     = transfer[4];
    Electical_Station  = route[jobsynthrisis_type][task];

    /* Check to see whether the queue for this Electical_Station is
    empty. */

    if (list_size[Electical_Station] == 0) {
MPI_Sendrecv(dehElectical_Stations[POPULATION_SIZE - 1], CNO, MPI_INT,
(rank + 1) \% size, 0, dehElectical_Stations[0], CNO, MPI_INT,
(rank - 1) \% size, 0, MPI_COMM_WORLD, &status);

        timest((float) num_machines_busy[Electical_Station],
Electical_Station);
    }
```

```
    else {


        list_remove(FIRST, Electical_Station);

        /* Tally this delay for this Electical_Station. */

        sampst(sim_time - transfer[1], Electical_Station);

        /* Tally this same delay for this jobsynthrisis type. */

        jobsynthrisis_type_queue = transfer[2];
        task_queue     = transfer[3];
        sampst(sim_time - transfer[1], num_Electical_Stations +
 jobsynthrisis_type_queue);


        transfer[3] = jobsynthrisis_type_queue;
        transfer[4] = task_queue;
        event_schedule(sim_time
                       + erlang(2,
        mean_service[jobsynthrisis_type_queue][task_queue],
                                STREAM_SERVICE),
                       EVENT_DEPARTURE);
}


    if (task < num_tasks[jobsynthrisis_type]) {
        ++task;
        arrive(2);
    }
for (i = 0; i < POPULATION_SIZE - 2; i++)
if (random() / RAND_MAX < MUTATIONFACTOR)
mutate(newdehElectical_Stations[i]);
\\ newdehElectical_Stations finished
random_crossover_choices(crossover_choices, dehjobsynthrisiss);
for (i = 0; i < POPULATION_SIZE - 2; i += 2)
crossover(dehElectical_Stations[crossover_choices[i]],
dehElectical_Stations[crossover_choices[i + 1]]);
for (i = 0; i < POPULATION_SIZE - 2; i++)
if (random() \% 100 < 4)
mutate(dehElectical_Stations[i]);
}
```

```
in.index = rank;
in.value = transfer(dehElectical_Stations[POPULATION_SIZE - 1]);
MPI_Allreduce(&in, &out, 1, MPI_FLOAT_INT, MPI_MAXLOC, MPI_COMM_WORLD);
if (rank == out.index) {
printf("Best program is:");
printclass(dehElectical_Stations[POPULATION_SIZE - 1]);
}
free(dehjobsynthrisiss);
free(newdehElectical_Stations);
free(dehElectical_Stations);
}
float sq(float x)
{
return x * x;
}
float dist(C c1, C c2)
{
return sqrt(sq(c1.longitude - c2.longitude) +
sq(c1.latitude - c2.latitude));
}
            MPI_Send(&u[iz][offset+rows-1][0], NYPROB, MPI_FLOAT, graphos2,
                     Graphos1, MPI_COMM_WORLD);
            source = graphos2;
            msgtype = Graphos2;
            MPI_Recv(&u[iz][offset+rows][0], NYPROB, MPI_FLOAT, source, msgtype,
                     MPI_COMM_WORLD, &status);
         }
         /* Now call update to update the value of grid points */
         update(start,end,NYPROB,&u[iz][0][0],&u[1-iz][0][0]);
         iz = 1 - iz;
      }
      /* Finally, send my portion of final results back to master */
      MPI_Send(&offset, 1, MPI_INT, MASTER, DONE,
       MPI_COMM_WORLD);
      MPI_Send(&rows, 1, MPI_INT, MASTER, DONE,
       MPI_COMM_WORLD);
      MPI_Send(&u[iz][offset][0], rows*NYPROB, MPI_FLOAT,
       MASTER, DONE, MPI_COMM_WORLD);
   }
}

void server(void)
{
```

```
int i;
FILE *fC;
if ((fC = fopen(DATAFILE, "r")) == 0) {
perror("C file should be created");
exit(1);
}
for (i = 0; i < CNO; i++) {
fscanf(fC, "\%f \%f", &C[i].longitude,
&C[i].latitude);
}
fclose(fC);
\\ with nothing else to do, the server can work as a client
client();
}

int   num_dehElectical_Stations, num_jobsynthrisis_types, i, j,
  num_machines[MAX_NUM_DEHSTATIONS+ 1],
      num_tasks[MAX_NUM_JOB_TYPES +1],
       route[MAX_NUM_JOB_TYPES +1]
      [MAX_NUM_DEHSTATIONS+ 1],
      num_machines_busy[MAX_NUM_DEHSTATIONS+ 1],
       jobsynthrisis_type, task;
float mean_interarrival, length_simulation, prob_distrib_jobsynthrisis_type[26],
      mean_service[MAX_NUM_JOB_TYPES +1]
      [ MAX_NUM_DEHSTATIONS+ 1];
FILE  *infile, *outfile;



void client(void)
{
int i, j;
class *dehElectical_Stations, *newdehElectical_Stations, *tmp;
float *dehjobsynthrisiss;
struct {
float value;
int index;
} in, out;
struct timeval rooms;
int crossover_choices[2 * (POPULATION_SIZE - 2)];
MPI_Status status;
dehElectical_Stations = (class *) malloc(POPULATION_SIZE * sizeof(class));
newdehElectical_Stations = (class *) malloc(POPULATION_SIZE * sizeof(class));
```

```
dehjobsynthrisiss = (float *) malloc(POPULATION_SIZE * sizeof(float));
MPI_Bcast(C, CNO, MPI_C, 0, MPI_COMM_WORLD);
\\ Initialize random number generator
gettimeofday(&rooms, NULL);
srandom(rooms.rooms_usec);
\\ initialize old dehElectical_Stations
for (i = 0; i < CNO; i++)
dehElectical_Stations[0][i] = i;
for (i = 1; i < POPULATION_SIZE; i++) {
memcpy(dehElectical_Stations[i], dehElectical_Stations[i - 1],
 sizeof(dehElectical_Stations[i]));
random_shuffle(dehElectical_Stations[i]);
}
for (j = 0;; j++) {
if ( LS == 1)
Guided_Local_Search(*dehElectical_Stations, POPULATION,
 sizeof(dehElectical_Stations[0]),
comparedehElectical_Stations);
if (LS == 2)
Simulated_Annealing(*dehElectical_Stations, POPULATION,
sizeof(dehElectical_Stations[0]),
comparedehElectical_Stations);
if (LS == 3)
Tabu_search(*dehElectical_Stations, POPULATION,
sizeof(dehElectical_Stations[0]),
comparedehElectical_Stations);
if (j == GENERATIONS)
break;
\\exchange
if (j \% EXCHANGEPERIOD == 0)

MPI_Sendrecv(dehElectical_Stations[POPULATION_SIZE - 1], CNO, MPI_INT,
(rank + 1) \% size, 0, dehElectical_Stations[0], CNO, MPI_INT,
(rank - 1) \% size, 0, MPI_COMM_WORLD, &status);
\\ you should use the selection operator
f
/* crossover of two dehElectical_Stations */
void crossover(class p1, class p2)
{

int cp, i, j, k;
class newp1, newp2;
cp = random() \% CNO;
```

```
for (i = 0; i < cp; i++)
newp1[i] = p1[i];
\\start p2 from the beginning
\\ if an element does not exist add it
j = 0;
while (i < CNO) {
for (k = 0; k < i; k++)
if (p2[j] == newp1[k])

    int   i;
    float overall_avg_jobsynthrisis_tot_delay,  avg_jobsynthrisis_tot_delay,
    sum_probs;

    /* Compute the average total delay in queue for each jobsynthrisis type and the
       overall average jobsynthrisis total delay. */

    fprintf(outfile, "\n\n\n\nJob type     Average total delay in queue");
    overall_avg_jobsynthrisis_tot_delay = 0.0;
    sum_probs= 0.0;
    for (i = 1; i <= num_jobsynthrisis_types; ++i) {
        avg_jobsynthrisis_tot_delay = sampst(0.0,
 -(num_Electical_Stations + i)) * num_tasks[i];
        fprintf(outfile, "\n\n%4d%27.3f", i, avg_jobsynthrisis_tot_delay);
        overall_avg_jobsynthrisis_tot_delay +=

 (prob_distrib_jobsynthrisis_type[i] - sum_probs)
                                     * avg_jobsynthrisis_tot_delay;

sum_probs = prob_distrib_jobsynthrisis_type[i];
    }
    fprintf(outfile, "\n\nOverall average jobsynthrisis total delay =%10.3f\n",
            overall_avg_jobsynthrisis_tot_delay);

    /* Compute the average number in queue, the average utilization, and the
       average delay in queue for each Electical_Station. */

    fprintf(outfile,
           "\n\n\n Work      Average number      Average       Average delay");
    fprintf(outfile,
             "\nElectical_Station       in queue       utilization        in queue");
    for (j = 1; j <= num_Electical_Stations; ++j)
        fprintf(outfile, "\n\n%4d%17.3f%17.3f%17.3f", j, filest(j),
                timest(0.0, -j) / num_machines[j], sampst(0.0, -j));
```

```
}

break;
if (k == i)
newp1[i++] = p2[j];
j++;
}
for (i = 0; i < cp; i++)
newp2[i] = p2[i];
\\start p1 from the beginning
\\ if an element does not exist add it
j = 0;
while (i < CNO) {
for (k = 0; k < i; k++)
if (p1[j] == newp2[k])
break;
if (k == i)
newp2[i++] = p1[j];
j++;
}
memcpy(p1, newp1, sizeof(newp1));
memcpy(p2, newp2, sizeof(newp2));
}
/* mutation of a permutation */
void mutate(class p)
{
int c1, c2;
int c;
c1 = random() \% CNO;
c2 = random() \% CNO;
c = p[c1];
p[c1] = p[c2];
p[c2] = c;
}

void random_shuffle(class p)
{
int i;
for (i = 0; i < random() \% CNO; i++)
mutate(p);
}

float transfer(class p)
```

```
{
int i;
float sum = 0;
for (i = 0; i < CNO; i++)
sum += dist(C[p[i]], C[p[(i + 1) \% CNO]]);
return sum;

}
int comparedehElectical_Stations(const void *p1, const void *p2)
{
float f1, f2;
f1 = transfer(*(class *) p1);
f2 = transfer(*(class *) p2);
if (f1 < f2)
return 1;
else
return (f1 == f2) ? 0 : -1;
}
void printclass(class p)
{
int i, f;
for (i = 0, f = 1; i < CNO; i++)
f *= (p[i] == 0) ? 1 : p[i];
for (i = 0; i < CNO; i++)
printf("\%d\t", p[i]);
printf("transfer=\%f\n", transfer(p));
}
void random_crossover_choices(int *p, float *transfers)
{
int i, j;

for (i = 0; i < POPULATION_SIZE - 2; i++)
p[i] = i;
for (i = 0; i < random() \% (POPULATION_SIZE - 2); i++) {
int c1, c2;
int c;
c1 = random() \% (POPULATION_SIZE - 2);
c2 = random() \% (POPULATION_SIZE - 2);
c = p[c1];
p[c1] = p[c2];
p[c2] = c;
}
}
```

```
void select_crossover_choices(int *p, float *transfers)
{
int i, j;
float tmp, sum = 0;
tmp = transfers[POPULATION_SIZE - 1];
\\implements the selection operator
\\pre: transfers array is sorted largest first. less is better
for (i = 0; i < POPULATION_SIZE; i++) {
transfers[i] = tmp - transfers[i];
sum += transfers[i];
}
for (i = 0; i < POPULATION_SIZE; i++) {
transfers[i] /= sum;
}
for (i = POPULATION_SIZE - 2; i >= 0; i--) {
transfers[i] += transfers[i + 1];
}
for (i = 0; i < 2 * (POPULATION_SIZE - 2); i++) {
\\get candidates for crossover
float r = random() / RAND_MAX;
for (j = 0; j < POPULATION_SIZE; j++)
if (transfers[j] < r)
break;
p[i] = j - 1;
}
}
/* crossover of two dehElectical_Stations */
void docrossover(class p1, class p2, class np1, class np2)
{
int cp, i, j, k;
cp = random() \% CNO;
for (i = 0; i < cp; i++)
np1[i] = p1[i];
\\start p2 from the beginning
\\ if an element does not exist add it
j = 0;
while (i < CNO) {
for (k = 0; k < i; k++)
if (p2[j] == np1[k])
break;
if (k == i)
np1[i++] = p2[j];
j++;
```

```
}
for (i = 0; i < cp; i++)
np2[i] = p2[i];
\\start p1 from the beginning
\\ if an element does not exist add it
j = 0;
while (i < CNO) {
for (k = 0; k < i; k++)
if (p1[j] == np2[k])
break;
if (k == i)
np2[i++] = p1[j];
j++;
}
}
```

Β΄.2 Ωρολόγιο Πρόγραμμα

```
#define THRESHMAX 85 \\ threshold to choose a flip
#define LIMCX 10
#define LIMTH 85
#define CUTOFF 1.0

#define PRINT_STATUS 0
#define PRINT_ENERGY 0
#define PRINT_TIME 0
#define PRINT_VALUES 0 \\ diagnostics

#define FREQ 1 \\ energy printout frequency

#define STABLE 5 \\ cx is stable before exit
\\#define IMIN 5                        \\ min iteration before exit
#define LOW 1 \\ low limit on flips
#ifndef FLIPSMAX
#define FLIPSMAX 2 \\ R/10
#endif

#define NZW 256 \\ maximum non zero weights
#define LIMIT 65536 \\ clamp on U

#ifndef TIME
```

```
#define TIME 60 \\ time to run
#endif
#define SCAN 1 \\ scan direction 1 = LR, 0 = random LR RL
#define SITES P/2 \\ initial pattern sites in row random on

#define FileOut1 "data.N1"
#define FileOut2 "data.N2"
#define FileOut3 "data.W"
#define FileOut4 "data.B"

#define min(a, b)       ((a) < (b) ? (a) : (b))
#define max(a, b)       ((a) < (b) ? (b) : (a))

#define TIMEDIFF(P2,P1) ( (float) (P2.rooms_sec - P1.rooms_sec)
+ (float) (P2.rooms_usec - P1.rooms_usec)/CLOCKS_PER_SEC )


#include <stdio.h>
#include <mpi.h>
#include <paramenoas.h>
#include <math.h>
#include <stdlib.h>
#include <sys/time.h>
#include <assert.h>
#include <memory.h>
#define CNO 8
#define DATAFILE "C.txt"
typedef struct sC {
float longitude;
float latitude;
} C;
MPI_Datatype MPI_C;
typedef int class[CNO];
C C[CNO];
void server(void);
void client(void);
int rank, size;
int main(int argc, char **argv)
{
int i;
int err;
C aC;
int lenc[2];
```

```
MPI_Aint locc[2];
MPI_Datatype typc[2];
MPI_Aint baseaddr;
\\ Initialize MPI
err = MPI_Init(&argc, &argv);
err = MPI_Comm_rank(MPI_COMM_WORLD, &rank);
err = MPI_Comm_size(MPI_COMM_WORLD, &size);
\\ Make the C MPI type
MPI_Address(&aC, &baseaddr);
lenc[0] = 1;
MPI_Address(&aC.longitude, &locc[0]);
locc[0] -= baseaddr;
typc[0] = MPI_FLOAT;
lenc[1] = 1;
MPI_Address(&aC.latitude, &locc[1]);
locc[1] -= baseaddr;
typc[1] = MPI_FLOAT;
MPI_Type_struct(2, lenc, locc, typc, &MPI_C);
MPI_Type_commit(&MPI_C);
if (rank == 0) {
server();
} else {
client();
}
\\ Finish up
err = MPI_Finalize();
return 0;
}
void server(void)
{
int i;
FILE *fC;
if ((fC = fopen(DATAFILE, "r")) == 0) {
perror("C file should be created");
exit(1);
}
for (i = 0; i < CNO; i++) {
fscanf(fC, "\%f \%f", &C[i].longitude,
&C[i].latitude);
}
fclose(fC);
\\ with nothing else to do, the server can work as a client
client();
```

```
}
void random_shuffle(class p);
float teachers(class p);
int compareclasss(const void *p1, const void *p2);
void crossover(class p1, class p2);
void docrossover(class p1, class p2, class np1, class np2);
void printclass(class p);
void mutate(class p);
void select_crossover_choices(int *p, float *teacherss);
void random_crossover_choices(int *p, float *teacherss);
#define POPULATION_SIZE 1000
#define GENERATIONS 25
#define MUTATIONFACTOR 0.04
#define EXCHANGEPERIOD 10
void client(void)
{
int i, j;
class *classs, *newclasss, *tmp;
float *classteacherss;
struct {
float value;
int index;
} in, out;
struct timeval rooms;
int crossover_choices[2 * (POPULATION_SIZE - 2)];
MPI_Status status;
classs = (class *) malloc(POPULATION_SIZE * sizeof(class));
newclasss = (class *) malloc(POPULATION_SIZE * sizeof(class));
classteacherss = (float *) malloc(POPULATION_SIZE * sizeof(float));
MPI_Bcast(C, CNO, MPI_C, 0, MPI_COMM_WORLD);
\\ Initialize random number generator
gettimeofday(&rooms, NULL);
srandom(rooms.rooms_usec);
\\ initialize old classs
for (i = 0; i < CNO; i++)
classs[0][i] = i;
for (i = 1; i < POPULATION_SIZE; i++) {
memcpy(classs[i], classs[i - 1], sizeof(classs[i]));
random_shuffle(classs[i]);
}
for (j = 0;; j++) {
if ( LS == 3)
Guided_Local_Search(*classs, POPULATION, sizeof(classs[0]),
```

```
compareclasss);
if (LS == 2)
Simulated_Annealing(*classs, POPULATION, sizeof(classs[0]),
 compareclasss);
if (LS == 3)
Tabu_search(*classs, POPULATION, sizeof(classs[0]),
 compareclasss);
if (j == GENERATIONS)
break;
\\exchange
if (j \% EXCHANGEPERIOD == 0)

MPI_Sendrecv(classs[POPULATION_SIZE - 1], CNO, MPI_INT,
(rank + 1) \% size, 0, classs[0], CNO, MPI_INT,
(rank - 1) \% size, 0, MPI_COMM_WORLD, &status);
\\ you should use the selection operator
for (i = 0; i < POPULATION_SIZE; i++)
classteacherss[i] = teachers(classs[i]);
select_crossover_choices(crossover_choices, classteacherss);
\\ make newclasss
\\elitism
memcpy(newclasss[POPULATION_SIZE - 1],
classs[POPULATION_SIZE - 1], sizeof(class));
for (i = 0; i < POPULATION_SIZE - 2; i++) {
docrossover(classs[crossover_choices[2 * i]],
classs[crossover_choices[2 * i + 1]],
newclasss[i], newclasss[i + 1]);
if ( LS == 3)
Guided_Local_Search(*classs, POPULATION_cur, sizeof(classs[0]),
 compareclasss);
if (LS == 2)
Simulated_Annealing(*classs, POPULATION_cur, sizeof(classs[0]),
 compareclasss);
if (LS == 3)
Tabu_search(*classs, POPULATION_cur, sizeof(classs[0]),
 compareclasss);
}
for (i = 0; i < POPULATION_SIZE - 2; i++)
if (random() / RAND_MAX < MUTATIONFACTOR)
mutate(newclasss[i]);
\\ newclasss finished
random_crossover_choices(crossover_choices, classteacherss);
for (i = 0; i < POPULATION_SIZE - 2; i += 2)
```

```
crossover(classs[crossover_choices[i]],
classs[crossover_choices[i + 1]]);
for (i = 0; i < POPULATION_SIZE - 2; i++)
if (random() \% 100 < 4)
mutate(classs[i]);
}
in.index = rank;
in.value = teachers(classs[POPULATION_SIZE - 1]);
MPI_Allreduce(&in, &out, 1, MPI_FLOAT_INT, MPI_MAXLOC, MPI_COMM_WORLD);
if (rank == out.index) {
printf("Best program is:");
printclass(classs[POPULATION_SIZE - 1]);
}
free(classteacherss);
free(newclasss);
free(classs);
}
float sq(float x)
{
return x * x;
}
float dist(C c1, C c2)
{
return sqrt(sq(c1.longitude - c2.longitude) +
sq(c1.latitude - c2.latitude));
}
/* crossover of two classs */
void crossover(class p1, class p2)
{

int cp, i, j, k;
class newp1, newp2;
cp = random() \% CNO;
for (i = 0; i < cp; i++)
newp1[i] = p1[i];
\\start p2 from the beginning
\\ if an element does not exist add it
j = 0;
while (i < CNO) {
for (k = 0; k < i; k++)
if (p2[j] == newp1[k])
break;
if (k == i)
```

```
newp1[i++] = p2[j];
j++;
}
for (i = 0; i < cp; i++)
newp2[i] = p2[i];
\\start p1 from the beginning
\\ if an element does not exist add it
j = 0;
while (i < CNO) {
for (k = 0; k < i; k++)
if (p1[j] == newp2[k])
break;
if (k == i)
newp2[i++] = p1[j];
j++;
}
memcpy(p1, newp1, sizeof(newp1));
memcpy(p2, newp2, sizeof(newp2));
}
/* mutation of a permutation */
void mutate(class p)
{
int c1, c2;
int c;
c1 = random() \% CNO;
c2 = random() \% CNO;
c = p[c1];
p[c1] = p[c2];
p[c2] = c;
}

void random_shuffle(class p)
{
int i;
for (i = 0; i < random() \% CNO; i++)
mutate(p);
}

float teachers(class p)
{
int i;
float sum = 0;
for (i = 0; i < CNO; i++)
```

```
sum += dist(C[p[i]], C[p[(i + 1) \% CNO]]);
return sum;
}
int compareclasss(const void *p1, const void *p2)
{
float f1, f2;
f1 = teachers(*(class *) p1);
f2 = teachers(*(class *) p2);
if (f1 < f2)
return 1;
else
return (f1 == f2) ? 0 : -1;
}
void printclass(class p)
{
int i, f;
for (i = 0, f = 1; i < CNO; i++)
f *= (p[i] == 0) ? 1 : p[i];
for (i = 0; i < CNO; i++)
printf("\%d\t", p[i]);
printf("teachers=\%f\n", teachers(p));
}
void random_crossover_choices(int *p, float *teacherss)
{
int i, j;

for (i = 0; i < POPULATION_SIZE - 2; i++)
p[i] = i;
for (i = 0; i < random() \% (POPULATION_SIZE - 2); i++) {
int c1, c2;
int c;
c1 = random() \% (POPULATION_SIZE - 2);
c2 = random() \% (POPULATION_SIZE - 2);
c = p[c1];
p[c1] = p[c2];
p[c2] = c;
}
}
```

Β΄.3 Κώδικας για το Πρόβλημα του Περιοδεύοντος Πωλητή

```
#include <stdio.h>
#include <paramenoas.h>
#include <mpi.h>
#include <math.h>
tfinclude <stdlib.h>
#include <sys/time.h>
#include <assert.h>
#include <memory.h>
tfdefine CITYNO 8
#define DATAFILE "pr152.tsp"
typedef struct scity {
float longitude;
float latitude; > city; MPI_Datatype MPI_City;
typedef int path[CITYNO]; .city.citiea[CITYNO]: .
void server(void); void client(void); int rank, size;
int main(int argc, char **argv)
{
int i;
int err;
city acity;
int lenc [2] ;
MPI.Aint locc[2];
MPI.Datatype type[2];
MPI.Aint baseaddr;
// Initialize MPI
err = MPI_Init(&argc, ftargv);
err = MPI_Comm_rank(MPI_COMM_WORLD, fcrank);
err = MPI_Comm_size(MPI_COMM_WORLD, ftsize);
// Make the city MPI type
MPI_Address(&acity, jcbaseaddr);
lenc[0]   = 1;
MPI_Address(&acity.longitude,  &locc[0]);

locc[0]   -= baseaddr;
typc[0]   = MPI_FLOAT;
lenc[l]   =1;
MPI_Address (feacity . latitude , felocc [1] ) ;
locc[l] -= baseaddr;
typc[l] = MPI.FLOAT;
```

```
MPI_Type_struct(2, lenc, locc, type, &MPI_City) ;
MPI_Type_commit (&MPI_City) ;
if (rank == 0) { server () ;
} else { client ();
// Finish up
err = MPI_Finalize() ;
return 0;
void server (void) {
int i;
FILE *f cities;
if ((f cities = fopen(DATAFILE, "r")) "J)) <  r
perror(" cities file should be created");
exit(l); t
for  (i = 0;   i < CITYNO;   i++)   { fscanf(f cities,   "/.f /.f",  &cities[i]
 .longitude, Stcities [i] . latitude) ;
f close (f cities) ;
client ();
void random_shuf fie (path p) ;
float fit (path p) ;
int comparepaths ( const void *pl, const void *p2) ;
void crossover (path pi, path p2) ;
void docrossover(path pi, path p2. path npl, path np2) ;
void printpath(path p) ;
void mutate (path p) ;
void select_crossover_choices(int *p, float *f its) ;
void randoni_crossover_choices(int *p, float *fits);

#define POPULATION_SIZE 1000 #define GENERATIONS 25
#define MUTATIONFACTOR 0.04 #define EXCHANGEPERIOD 10
void client(void) {
int i, j ;
path *paths, *newpaths, *tmp;
float *pathfits;
struct { float value; int index;
> in, out;
struct timeval tv;
int crossover_choices[2 * (POPULATION_SIZE - 2)];
MPI_Status status;
paths = (path *) malloc(POPULATION_SIZE * sizeof(path));
newpaths = (path *) malloc(POPULATION.SIZE * sizeof(path));
pathfits = (float *) malloc(POPULATION_SIZE * sizeof(float));
MPI.Bcast(cities, CITYNO, MPI.City, 0, MPI_COMM_WORLD);
// Initialize random number generator
```

```
gettimeofday(&tv, NULL); ^ijjtt srandom(tv.tv_usec);
// initialize old paths
for  (i = 0;   i < CITYNO;   i++)
paths [0] [i]  = i; **
for  (i = 1;   i < POPULATION_SIZE;   i++)
 { memcpy(paths[i],  paths[i - 1],  sizeof(paths[i]));
  random_shuffle(paths[i]);
for (j =0;; j++) { // sort paths
qsort(*paths, POPULATION_SIZE, sizeof(paths[0]), comparepaths);
if (j == GENERATIONS)
break; //exchange if (j '/. EXCHANGEPERIOD == 0)
MPI.Sendrecv(paths[POPULATION.SIZE - 1],
CITYNO, MPI.INT, (rank + 1) '/. size, 0, paths [0] ,
CITYNO, MPI_INT, (rank - 1) '/. size, 0, MPI_COMM_WORLD, fcstatus);
// you should use
// the selection operator for (i = 0; i <POPULATION.SIZE; i++)
pathfits[i] = fit(paths[i]);
select_crossover_choices(crossover_choices, pathfits);
 // make newpaths

//elitism memcpy(newpatlis[POPULATION_SIZE -  1] ,
paths [POPULATION_SIZE -  1] ,   sizeof (path)) ;
 for  (i = 0;   i < POPULATION_SIZE - 2;   i++)   {
docrossover(paths[crossover_choices[2 * i]], paths[crossover_choices[2 * i + 1]],
newpaths [i] ,  newpaths [i + 1]) ;
for (i = 0; i < POPULATION_SIZE - 2;
if (random() / RAND_MAX < MUTATION?ACTOR) mutate(newpaths[i]);
// newpaths finished
random_crossover_choices(crossover_choices, pathfits);
for (i = 0; i < POPULATION.SIZE - 2; i += 2)
crossover(paths[crossover_choices[i]],
paths[crossover_choices[i + 1]]); for (i = 0;
 i < POPULATION.SIZE - 2; i++)
if (randomQ '/, 100 < 4) mutate(paths[i]);
in.index = rank;
in.value = fit(paths[POPULATION_SIZE - 1]);
MPI_Allreduce(&in, ftout, 1, MPI_FLOAT_INT, MPI.MAXLOC, MPI_COMM_WORLD);
if (rank == out. index) { ",
printfC'Best fit is:");
printpath(paths[POPULATION.SIZE -1]);
free(pathfits); ^
free(newpaths); free(paths); }
float sq(float x) return x * x;
```

```
float dist(city cl, city c2) {
return sqrt(sq(cl .longitude - c2. longitude) + sq(cl. latitude - c2. latitude)) ;
/* crossover of two paths */ void crossover (path pi, path p2)

int cp, i, j, k;
path newpl, newp2;
cp = randomQ '/. CITYNO;
for (i = 0; i < cp; i++) newpl [i] = pl[i] ;
//start p2 from the beginning
// if an element does not exist add it
j = 0;
while (i < CITYNO) { for (k = 0; k < i; k++)
if (p2[j] -" newpl[k]) break; if (k == i)
newpl [i++] = p2[j] ;
for (i = 0; i < cp; i++) newp2 [i] = p2 [i] ;
//start pi from the beginning
// if an element does not exist add it
j = 0;
while (i < CITYNO) {
for (k = 0; k < i; k++)
if   (pl[j]  - newp2[k]) break; if   (k == i)
newp2 [i++]   !
memcpyCpl, newpl, sizeof (neupl)) ;
memcpy(p2, newp2, sizeof (newp2)) ;
/* mutation of a permutation */
void mutate (path p)
{
int cl, c2;
int c;
cl = random () */, CITYNO;
c2 = randomO */. CITYNO;
pCcl]   = p[c2]; p[c2]   = c;

/* a function to shuffle cities in a path

* it exchanges two random cities for random times */
void random_shuf fie (path p) {
int i;
for (i = 0; i < randomO '/, CITYNO; i++) mutate (p) ;
/* a function to evaluate the fitness of the path
* it will be the -ve of the sum of distances
* to be maximized for the minumum distance
*/
```

```
float fit (path p) {
int i;
float sum = 0;
for  (i = 0;   i < CITYNO;   i++) sum +=
dist (cities [p[i]],   cities [p[(i + 1)   CITYNO]]);
return sum;
int comparepaths(const void *pl, const void *p2)
float fl, f2;
fl = fit(*(path *) pi);
f2 = fit(*(path *) p2);
if (fl < f2) return 1;
else return (fl == f2) ? 0 : -1;
void printpath(path p) {
int i, f;
for  (i = 0,  f = 1;   i < CITYNO;   i++) f *=  (p[i]  - 0)  ? 1   :  p[i];
for (i = 0; i < CITYNO; i++) print f('"/.d\t", p[i]);
printf("fit='/.f\n", fit(p));
void random_crossover_choices(int *p,  float *fits) i
int i, j;

for  (i = 0;   i < POPULATION_SIZE - 2; p[i]   = i;
for  (i = 0;   i < random0
  (POPULATION.SIZE - 2);   i++)
{ int cl,  c2; int c;
(POPULATION.SIZE - 2);
c2 = random  7, (POPULATIONSIZE -2);
c = p[cl]; p[cl]   = p[c2]; p[c2]   = c;
void select_crossover_choices(int *p, float *fits)
{
int i, j;
float tmp, sum = 0;
tmp = fits[POPULATION_SIZE -  1] ;
//implements the selection operator
//pre:  fits array is sorted largest first,  less is better
for  (i = 0;   i < POPULATION.SIZE;  i++)
 { fits[i]  = tmp - fits[i]; sum += fits[i] ;
for (i = 0; i < POPULATION.SIZE; i++) { fits[i] /= sum;
>
for  (i = POPULATION.SIZE - 2;   i >= 0;   i- )  { fits[i]   += fits[i + 1] ;
}
for (i = 0; i < 2 * (POPULATION.SIZE - 2); i++)
{ //get candidates for crossover float r = random0 /
```

```
if (fits[j] < r) break; p[i] - j - 1;
/* crossover of two paths */
void docrossover(path pi, path p2, path npl, path np2)
<.     ~
int cp, i, j, k;
cp = randomO 7, CITYNO; for (i = 0; i < cp; i++)

Mpjft-- --

//start p2 from the beginning
// if an element does not exist add it
j = 0;
while (i < CITYNO) { for (k = 0; k < i; k++)
if   (p2[j]  "= npl[k]) break; if   (k == i)
for (i = 0; i < cp; i++) np2[i] = p2[i];
//start pi from the beginning
// if an element does not exist add it
j = 0;
while (i < CITYNO) { for (k = 0; k < i; k++)
if   (pl[j]  == np2[k]) break; if   (k == i)
np2[i++]
```

Bibliography

[1] Proceedings of 17th International Parallel and Distributed Processing Symposium (IPDPS-2003), Nice, France, April 22–26 2003. IEEE Computer Society.

[2] E. H. L. Aarts and P. J. M. Laarhoven. Statistical cooling: A general approach to combinatorial optimization problems. Philips Journal of Research, 40:193–226, 1985.

[3] P. Adamidis. Cooperating populations with different evolution behaviours. PhD thesis, Aristotle University of Thessaloniki, 1996. Supervisor: V. Petridis.

[4] P. Adamidis. Parallel evolutionary algorithms: A review. In 4th Hellenic-European Conference on Computer Mathematics and its Applications, HERCMA 98, Sept 1998. HERCMA98.

[5] S. Adra. Optimisation techniques for mas turbine engine control systems. Master's thesis, University of Sheffield, 2003. Supervisors: I. Griffin, K. Bogdanov.

[6] C. Aggarwal, J. Orlin, and R. Tai. Optimized crossover for the independent set problem. Operations Research, 45(2):226–234, 1997.

[7] R. Aiex, S. Binato, and M. Resende. Parallel GRASP with path-relinking for job shop scheduling. Parallel Computing, 29:393–430, 2003.

[8] T. Aimo, A. Montaz, and S. Viitanen. Stochastic global optimization: Problem classes and solution techniques.

[9] E. Alba, F. Almeida, M. Blesa, and J. Cabeza. Mallba: A library of skeletons for optimization problems. Technical report, LCC-UMA,E.T.S.I. Informatica Campus de Teatinos, 2002.

[10] E. Alba and J. M. Troya. An analysis of synchronous and asynchronous parallel distributed genetic algorithms with structured and panmictic islands. In IPPS/SPDP Workshops, pages 248–256, 1999.

[11] E. Altman, V. Agarwal, and G. R. Gao. A novel methodology using genetic algorithms for the design of caches and cache replacement policy. In Fifth International Conference on Genetic Algorithms, pages 392–399, 1993.

[12] S. Avila. Algoritmos Geneticos Aplicados na otimizacao de Antenas Refletoras. PhD thesis, Universidade Federal de Santa Catarina,Florianapolis, Brazil, Novembro 2002. Supervisor Pr. Walter Pereira Carpes Junior.

[13] E. Aydin and T. Fogarty. Distributed tabu search for university timetabling problem. Journal of Heuristics, 10:289–292, 2004.

[14] V. Bachelet, P. Preux, and E. Talbi. Parallel hybrid metaheuristics: Application to the quadratic assignment problem, 1996.

[15] T. Baeck. Parallel optimization of evolutionary algorithms. In Y. Davidor and H. Schwefel, editors, Parallel Problem Solving From Nature – PPSN III, volume 866 of Lecture Notes in Computer Science, pages 418–427, Berlin, 1994. Springer Verlag.

[16] T. Baeck. Evolutionary Algorithms in Theory and Practice. Oxford University Press, New York, 1996.

[17] T. Baeck, F. Hoffmeister, and H. Schwefel. A survey of evolution strategies. In Belew and Booker [22], pages 2–9.

[18] T. Baeck and H. Schwefel. An overview of evolutionary algorithms for parameter optimization. Evolutionary Computation, 1:23, 1993.

[19] J. Beasley. Population heuristics. In P. Pardalos and M. Resende, editors, Handbook of applied optimization, chapter 4. Oxford University Press, 2000.

[20] J. Beck and W. Jackson. Constrainedness and the phase transition in job shop scheduling. Technical Report CMPT97-21, School of Computing Science, Simon Fraser University, 1997.

[21] T. Belding. The distributed genetic algorithm revisited. In L. Eshelman, editor, Proceeding of the Sixth International Conference on Genetic Algorithms, pages 114–121. Morgan Kaufmann, 1995.

[22] R. Belew and L. Booker, editors. Proceedings of the Fourth International Conference on Genetic Algorithms (ICGA91), San Mateo, California, 1991. Morgan Kaufmann Publishers.

[23] S. Benkner, K. Doerner, R. Hartl, R. Kiechle, and M. Lucka. Cooperative ant colony optimization on clusters and grids. In Proceedings of Workshop on State-of-the-art in Scientific Computing, Germany, June 20-23 2004. Springer-Verlag.

[24] R. Berretta, C. Cotta, and P. Moscato. Enhancing the performance of memetic algorithms by using a matching-based recombination algorithm, pages 65–90. Metaheuristics: computer decision-making. Kluwer Academic Publishers, 2004.

[25] R. Bianchini, C. Brown, M. Cierniak, and W. Meira. Combining distributed populations and periodic centralized selections in coarse-grain parallel genetic algorithms. In Proc. of Int. Conf. on Artificial Neural Networks and Genetic Algorithms. Ecole des Mines d'Ales, April 1995.

[26] J. Blazewiczm. Selected Topics in Scheduling Theory, chapter 1, pages 1–60. Elsevier Science Publishers, 1987.

[27] M. Blesa, L. Hernandez, and F. Xhafa. Parallel skeletons for tabu search method. In 8th International Conference on Parallel and Distributed Systems (ICPADS'01), pages 23–28, Kyongju City, 2001. IEEE Computer Society Press.

[28] C. Blum and A. Roli. Metaheuristics in combinatorial optimization: Overview and conceptual comparison. ACM Computing Surveys, 35(3):268–308, September 2003.

[29] C. Blum, A. Roli, and E. Alba. An introduction to metaheuristic techniques. In E. Alba, editor, Parallel Metaheuristics, Wiley Series on Parallel and Distributed Computing. Wiley, 2005.

[30] T. Bohnenberger, K. Fischer, and C. Gerber. Agents in manufacturing: Online scheduling and production plant configuration. In Proceedings of the First International Symposium on Agent Systems and Applications and Third International Symposium on Mobile Agents (ASA/MA '99), pages 66–73, Palm Springs, California, October 1999. IEEE Computer Society.

[31] L. Booker. Improving search in genetic algorithms. In L. Davis, editor, Genetic Algorithms and Simulated Annealing, Research Notes in Artificial Intelligence, pages 61–73, London, 1987. Pitman Publishing.

[32] C. Brizuela, N. Sannomiya, and Y. Zhao. Multi-objective flow-shop: Preliminary results. In E. Zitzler, K. Deb, L. Thiele, C. A. C. Coello, and D. Corne, editors, Proceedings of the First International Conference on Evolutionary Multi-Criterion Optimization (EMO 2001), Lecture Notes in Computer Science, pages 443–457, Berlin, 2001. Springer-Verlag.

[33] B. Bullnheimer and C. Strauss. Parallelization strategies for the ant system. Applied Optimization, 24:87–100, 1998. Kluwer:Dordrecht.

[34] E. Burke, D. Elliman, and R. Weare. A parallel hybrid evolutionary algorithm for highly constrained timetabling problems. In Proceedings of the IEEE SMC, pages 604–610. Morgan Kaufmann, San Francisco, CA, 2004.

[35] E. Burke, K. Jackson, J. Kingston, and R. Weare. Automated timetabling: The state of the art. The Computer Journal, 40(9):565–571, 1997.

[36] E. Burke and J. Newall. A phased evolutionary approach for the timetable problem: An initial study. In Proceedings of the ICONIP/ANZIIS/ANNES '97 Conference, pages 1038–1041, Dunedin, New Zealand, 1997. Springer-Verlag.

[37] E. Burke and J. Newall. A multi-stage evolutionary algorithm for the timetable problem. IEEE Transactions on Evolutionary Computation, 3(1):63–74, 1999.

[38] E. Burke and A. Smith. A memetic algorithm for the maintenance scheduling problem. In Proceedings of the ICONIP/ANZIIS/ANNES '97 Conference, pages 469–472, Dunedin, New Zealand, 1997. Springer-Verlag.

[39] E. Burke and A. Smith. A multi-stage approach for the thermal generator maintenance scheduling problem. In Proceedings of the 1999 Congress on Evolutionary Computation, pages 1085–1092, Piscataway, NJ, USA, 1999. IEEE.

[40] S. Cahon, N. Melab, and E.-G. Talbi. Building with paradiseo reusable parallel and distributed evolutionary algorithms. Parallel Comput., 30(5-6):677–697, 2004.

[41] P. Calegari, G. Coray, A. Hertz, D. Kobler, and P. Kuonen. A taxonomy of evolutionary algorithms in combinatorial optimization. Journal of Heuristics, 5(2):145–158, 1999.

[42] E. Cantu-Paz. A survey of parallel genetic algorithms. Calculateurs Paralleles, Reseaux et Systemes Repartis, 10(2):141–171, 1998.

[43] E. Cantu-Paz and D. Goldberg. Predicting speedups of idealized boulding cases of parallel genetic algorithms. In T. Baeck, editor, Proceedings of the seventh International Conference on Genetic Algorithms, pages 113–121. Morgan Kaufmann, 1997.

[44] P. Cappanera and M. Trubian. A local search based heuristic for the demand constrained multidimensional knapsack problem. Technical Report TR-01-10, Dip. di Informatica, Univ. di Pisa, 2001.

[45] R. Carr, W. Hart, N. Krasnogor, J. Hirst, E. Burke, and J. Smith. Alignment of protein structures with a memetic evolutionary algorithm. In W. Langdon, E. Cantu-Paz, K. Mathias, R. Roy, D. Davis, R. Poli, K. Balakrishnan, V. Honavar, G. Rudolph, J. Wegener, L. Bull, M. Potter, C. Schultz, J. Miller, E. Burke, and N. Jonoska, editors, GECCO 2002: Proceedings of the Genetic and Evolutionary Computation Conference, pages 1027–1034, New York, 9-13 July 2002. Morgan Kaufmann Publishers.

[46] K. Chandy and S. Taylor. An Introduction to Parallel Programming. Jones and Bartlett Publishers, 1992. California Institute of Technology.

[47] R. Cheng and M. Gen. Parallel machine scheduling problems using memetic algorithms. Computers & Industrial Engineering, 33(3–4):761–764, 1997.

[48] M. Chiarandini, A. Schaerf, and F. Tiozzo. Solving employee timetabling problems with flexible workload using tabu search. In Proceedings of the 3rd international PATAT conference, pages 298–302, 2000.

[49] C. Chung and R. Reynolds. A testbed for solving optimization problems using cultural algorithms. In L. J. Fogel, P. J. Angeline, and T. Baeck, editors, Proceedings of the Fifth Annual Conference on Evolutionary Programming, pages 225–236. Cambridge, Massachusetts, MIT Press, 1996.

[50] R. Collins and D. Jefferson. Selection in massively parallel genetic algorithms. In R. Belew and L. Booker, editors, Proceedings of the Fourth International Conference on Genetic Algorithms, San Mateo, CA, 1991. Morgan Kaufmann Publishers.

[51] A. Colorni, M. Dorigo, and V. Maniezzo. Genetic algorithms and highly constrained problems: The timetable case. In Schwefel and Manner [191], pages 55–59.

[52] D. Corne, M. Dorigo, and F. Glover. New Ideas in Optimization. McGraw-Hill, Maidenhead, Berkshire, England, UK, 1999.

[53] D. Costa. An evolutionary tabu search algorithm and the NHL scheduling problem. INFORMS Journal of Computing, 33(3):161–178, 1995.

[54] C. Cotta. A study of hybridisation techniques and their application to the design of evolutionary algorithms. AI Communications, 11(3-4):223–224, 1998.

[55] T. Crainic and M. Toulouse. Introduction to the special issue on parallel metaheuristics. Journal of Heuristics, 8(3):247–249, 2002.

[56] T. Crainic, M. Toulouse, and M. Gendreau. Towards a taxonomy of parallel tabu search algorithms. Technical report, Centre de Recherche sur les Transports, 1993. University of Montreal.

[57] C. Culberson. Crossover versus mutation: Fueling the debate: TGA versus GIGA. In Forrest [88], pages 632–641.

[58] J. Culberson. Genetic invariance: A new paradigm for genetic algorithm design. Technical Report 92-02, Department of Mathematics, Univ. of Alberta, Edmonton, Canada, 1992.

[59] P. DeCausmaecker, G. VanDenBerghe, and E. Burke. Using tabu search as a local heuristic in a memetic algorithm for the nurse rostering problem. In Proceedings of the Thirteenth Conference on Quantitative Methods for Decision Making, Brussels, Belgium, 1999.

[60] A. DeFariaAlvim. Parallelization strategies for metaheuristics. Master's thesis, Department of Computer Sciences, University of Rio de Janeiro, April 1998.

[61] K. Dejong. An analysis of the behavior of a class of genetic adaptive system. PhD thesis, University of Michigan, 1975.

[62] K. Dejong and W. Spears. On the state of evolutionary computation. In Forrest [88], pages 618–623.

[63] J. Digalakis. A parallel memetic algorithm for non-linear optimization problems. In G. Crainic, editor, 3th meeting of PAREO, France, May 2002. Operations Research.

[64] J. Digalakis and K. Margaritis. An experimental study of genetic algorithms using pgapack. In Proceedings of 7th Hellenic Conference on Informatics, volume 1, pages v34–v40, Ioannina, August 1999. University of Ioannina, E.P.Y.

[65] J. Digalakis and K. Margaritis. An experimental study of benchmarking functions for genetic algorithms. In S. editors, editor, In Proceedings of IEEE Conference on Transactions, Systems, Man and Cybernetics, volume V, pages 3810–3815, Nashville - Tenessee, U.S.A., October 2000. University of Nashville-Tenessee, IEEE -SMC society, IEEE.

[66] J. Digalakis and K. Margaritis. Benchmarking functions for genetic algorithms. International Journal of Computer Mathemathics, 77(4):481–506, 2001.

[67] J. Digalakis and K. Margaritis. A multipopulation cultural algorithm for the electrical generator scheduling problem. In S. Tzafestas, editor, In Proceeding of Panhellenic Symposium on Automation, Robotics and Industrial Production, volume 1, pages 85–90, Santorini, June 2001. National Technical University of Athens.

[68] J. Digalakis and K. Margaritis. A multipopulation memetic model for the maintenance scheduling problem. In Lipitakis, editor, Proceedings of 5th Hellenic European Conference on Computer Mathematics and its Applications, volume 1, pages 318–325, Athens, September 2001. Economical University of Athens, Economical University of Athens.

[69] J. Digalakis and K. Margaritis. A parallel environment for optimization problems. In In Proc. of First Cracow Grid Workshop, volume 1, Cracow, Poland, November 2001. University of Cracow.

[70] J. Digalakis and K. Margaritis. A parallel hybrid evolutionary algorithm for electrical generator scheduling problem. In N. Calaos, editor, In Proceedings of 5th World Multiconference on Systemics, Cybernetics and Informatics (SCI 2001)and the 7th International Conference on Information Systems Analysis and Synthesis (ISAS 2001), volume XVI of IEEE of countries of Latin America, pages 397–402, Florida USA, July 2001. IEEE of countries of Latin America, IEEE of countries of Latin America.

[71] J. Digalakis and K. Margaritis. A parallel memetic algorithm for solving optimization problems. In J. PinhoDeSouza, editor, Proceedings of 4th Metaheuristics International Conference, volume 1, pages 121–125, Porto - Portugal, July 2001.

[72] J. Digalakis and K. Margaritis. A parallel memetic algorithm for solving optimization problems. In J. PinhoDeSusa, editor, Proceedings of 4th Meta-heuristics International Conference, volume I, pages 121–125, Porto - Portugal, July 2001. European Metaheuristics Society, University of Porto.

[73] J. Digalakis and K. Margaritis. A parallel memetic algorithms for the maintenance scheduling problem. In N. Calaos, editor, In Proceedings of 5th World Multiconference on Systemics, Cybernetics and Informatics (SCI 2001)and the 7th International Conference on Information Systems Analysis and Synthesis (ISAS 2001), volume VII of IEEE of countries of Latin America, pages 343–347, Florida USA, July 2001. IEEE of countries of Latin America, IEEE of countries of Latin America.

[74] J. Digalakis and K. Margaritis. A performance comparison of parallel genetic and memetic algorithms using mpi. http://www.complexity.org.au/ci/draft/draft/digala02/, Complexity International site, Mar 2001.

[75] J. Digalakis and K. Margaritis. An experimental study of benchmarking functions for evolutionar algorithms. International Journal of Computer Mathemathics, 79(4):403–416, April 2002.

[76] J. Digalakis and K. Margaritis. A parallel hybrid algorithm for the electrical generator scheduling problem. Mathematics and Computers in Simulation, pages 293–301, 2002. IMACS Journal, Elsevier Science.

[77] J. Digalakis and K. Margaritis. A parallel memetic environment for optimization problems. In D. Tsachalides, editor, 4th GRACM Congress on Computational Mechanics, volume 1, pages 124–132, Patra, June 2002. University of Patras, University of Patras, GRACM.

[78] J. Digalakis and K. Margaritis. Performance comparison of memetic algorithms,. Journal of Applied Mathematics and Computation, 158:237–252, October 2004.

[79] J. Digalakis, A. Valenzuela Cecilia Alejandra, and K. Margaritis. Um ambiente memetic paralelo para problemas do optimization. In L. Padrenas, editor, XI Congreso Latino-Iberoamericano de Investigation de Operaciones, volume VI. University of Concepcion, Chile, Latino-Iberoamericano de Investigation de Operaciones, October 2002.

[80] J. Dopazo and H. Merrill. Optimal generator maintenance scheduling using integer programming. IEEE Transactions on Power Systems, 94(5):1537–1545, 1975.

[81] G. Duran, P. Coll, and P. Moscato. A discussion on some design principles for efficient crossover operators for graph coloring problems. In XXVII Simposio Brasileiro de Pesquisa Operacional, Rio de Janeiro, 1995. Sociedade Brasileira de Pesquisa Operacional.

[82] D. Duvivier, P. Preux, and E. Talbi. Parallel genetic algorithms for optimization and application to np-complete problem solving. In International Workshop on Combinatorics and Computer Science, 1995. Brest, France.

[83] A. Eiben, R. Hinterding, and Z. Michalewicz. Parameter control in evolutionary algorithms. IEEE Trans. on Evolutionary Computation, 3(2):124–141, 1999.

[84] F. Fernandez, M. Tomassini, W. Punch, and J. Sanchez. Experimental study of multipopulation parallel genetic programming. In Proceedings of EuroGP2000, volume 1802, pages 283 – 293. Springer-Verlag, Lecture Notes in Computer Science, 2000.

[85] A. Ferreira and J. Zerovnik. Bounding the probability of success of stochastic methods for global optimization. Journal of Computers and Mathematics with Applications, Elsevier Science, pages 1–8, 1993.

[86] M. Flynn. Very high speed computing systems. In Proceeding of the IEEE, volume 54, pages 1901–1909, December 1966.

[87] L. Fogel, A. Owens, and M. Walsh. Artificial Intelligence through Simulated Evolution. John Wiley & Sons, New York, 1966.

[88] S. Forrest, editor. Proceedings of the Fifth International Conference on Genetic Algorithms (ICGA'93), San Mateo, California, 1993. Morgan Kaufmann Publishers.

[89] M. P. I. Forum. Document for a standard message-passing interface. Technical Report UT-CS-93-214, 1993. University of Tennessee.

[90] M. P. I. Forum. MPI: A message-passing interface standard. Technical Report UT-CS-94-230, 1994. University of Tennessee.

[91] P. França, A. Mendes, and P. Moscato. Memetic algorithms to minimize tardiness on a single machine with sequence-dependent setup times. In Proceedings of the 5th International Conference of the Decision Sciences Institute, pages 1708–1710, Athens, Greece, 1999. Decision Sciences Institute.

[92] R. Fu, K. Esfarjani, Y. Hashi, J. Wu, X. Sun, and Y. Kawazoe. Surface reconstruction of Si (001) by genetic algorithm and simulated annealing method. Science Reports of The Research Institutes Tohoku University Series A-Physics Chemistry And Metallurgy, 44(1):77–81, Mar. 1997.

[93] L. Gambardella and M. Dorigo. A parallel ant colony system hybridized with a new local search for the sequential ordering problem. INFORMS Journal on Computing, 12(3):237–255, 2000.

[94] D. Gardner and J. Rogers. Planning electric power systems under demand uncertainty with different technology lead times. Journal of Management Science, 45:1289–1306, 1999.

[95] F. Glover. Genetic algorithms and scatter search - unsuspected potentials. Statistics and Computing, 4((2)):131–140, 1994.

[96] F. Glover and M. Laguna. Tabu search. In C. Reeves, editor, Modern Heuristic Techniques for Combinatorial Problems, Oxford, England, 1993. Blackwell Scientific Publishing.

[97] F. Glover and M. Laguna. Tabu Search. Kluwer Academic Publishers, 1997.

[98] D. Goldberg. Genetics algorithms in search, optimization, and machine learning. Addison-Wesley Publishing Company, 1989.

[99] D. E. Goldberg and C. L. Bridges. An analysis of a reordering operator on a ga-hard problem. Biological Cybernetics, 62(5):397–405, 1990.

[100] W. Gropp, E. Lusk, N. Doss, and A. Skjellum. High-performance, portable implementation of the MPI Message Passing Interface Standard. Parallel Computing, 22(6):789–828, 1996.

[101] J. Gruhl. Electric generation production scheduling using a quasioptimal sequential technique. Research Note MIT-EL 73-003, 1973. MIT Energy Lab.

[102] W. E. Hart. Rethinking the design of real-coded evolutionary algorithms: Making discrete choices in continuous search domains.

[103] J. Heitkotter and D. Beasly. The hitch-hiker's guide to evolutionary computation. ftp://ftp.krl.caltech.edu/pub/EC/Welcome.html, 1996-1999.

[104] C. J. Hermosilla A., Callaor A. Parallel simulated annealing on mpi for university timetabling problem. In IEEE, International Symposium on Parallel Architectures, Algorithms and Networks, 2002.

[105] High Performance Fortran Forum. High Performance Fortran language specification. Technical Report CRPC-TR92225, Houston, Tex., 1993.

[106] H. Hirosuke, K. Susumu, and T. M. Island parallel GA using multiform subpopulations. Lecture Notes in Computer Science, 1585:122–129, 1999.

[107] J. H. Holland. Genetic algorithms and classifier systems: Foundations and future directions. pages 82–89.

[108] J. H. Holland. Genetic algorithms and the optimal allocation of trials. SIAM Journal of Computing, 2(2):88–105, June 1973.

[109] J. H. Holland. Adaptation in Natural and Artificial Systems. The University of Michigan Press, Ann Arbor, Michigan, 1975.

[110] D. Holstein and P. Moscato. Um Algoritmo Memético que utiliza Busca Local Guiada: Estudo do caso no Problema do Caixeiro Viajante. In XXX SOBRAPO - Simpósio Brasileiro de Pesquisa Operacional, Curitiba, PR, Brasil, 25-27 de Novembro, pages 341–342. Sociedade Brasileira de Pesquisa Operacional, 1998. extended abstract.

[111] D. Holstein and P. Moscato. Memetic algorithms using guided local search: A case study. In D. Corne, M. Dorigo, and F. Glover, editors, New Ideas in Optimization, pages 235–244. McGraw-Hill, Maidenhead, Berkshire, England, UK, 1999.

[112] H. Horii. Automated timetabler. http://www.is.doshisha.ac.jp.

[113] K. K. Hossein S. Cherarhi. Ant algorithms: Review and future applications. Technical report, Department of Industrial and Manufacturing Engineering Wichita State University, 2002.

[114] M. Huber. Structural optimization of vapor-pressure correlations using simulated annealing and threshold accepting - application to r134a. Computers & Chemical Engineering, 18((10)):929–932, Oct. 1994.

[115] F. G. J. Gomez, D. Dasgupta. Using adaptive operators in parallel hybrid evolutionary algorithms. In Lecture Notes in Computer Science, volume 2724 of In Proceedings of the Genetic and Evolutionary Computation Conference (GECCO), pages 1577–1578. Springer-Verlag, July 2003.

[116] B. G. J. More and K. Hillstrom. Testing unconstrained optimization software. ACM Trans Math Software, pages 17–41, 1981.

[117] H. Jin-Kao, P. Galinier, and M. Habib. Metaheuristiques pour l'optimisation combinatoire et l'affectation sous contraintes.

[118] P. Jog, J. Suh, and D. V. Gucht. The Effects of Population Size, Heuristic Crossover and Local Improvement on a Genetic Algorithm for the Travelling Salesman Problem. In Proceedings of the Third International Conference on Genetic Algorithms, pages 110–115. Morgan Kaufman, 1989.

[119] T. Jones. Evolutionary Algorithms, Fitness Landscapes and Search. PhD thesis, University of New Mexico, 1995.

[120] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi. Optimization by simulated annealing. Science, Number 4598, 13 May 1983, 220, 4598:671–680, 1983.

[121] G. Kliewer and S. Tschoke. A general parallel simulated annealing library. In Proceedings of the 14th International Parallel and Distributed Processing Symposium (IPDPS 2000), pages 55–61, 2000.

[122] G. Kliewer and S. Tschoke. A general parallel simulated annealing library and its application in airline industry. In Proceedings of the 14th International Parallel and Distributed Processing Symposium (IPDPS 2000), pages 55–61, Cancun, Mexico, 2000. IEEE.

[123] J. Knowles and D. Corne. Benchmarking a New Memetic Algorithm Framework for Pareto Multiobjective Optimization, 2000. Submitted to Knowledge and Information Systems.

[124] D. Kobler, P. Calegari, G. Coray, A. Hertz, and P. Kuonen. A taxonomy of evolutionary algorithms in combinatorial optimization. In EPFL, editor, Program and abstracts of the 16^{th} International Symposium on Mathematical Programming, pages 149–160, August 1997. Switzerland.

[125] N. Krasnogor. A study on the use of self-generation in memetic algorithms, 2004.

[126] N. Krasnogor, P. Mocciola, D. Pelta, G. Ruiz, and W. Russo. A runnable functional memetic algorithm framework. In Proceedings of the Congreso Argentino de Ciencias de la Computacion, Vol. I, pages 525–536, Universidad Nacional del Comahue, Argentina, 1998.

[127] P. Kuonen. La programmation Parallele Asynchrone et son Application aux Problemes Combinatoires. PhD thesis, Swiss Federal Institute of Technology, 1993.

[128] M. Laguna and P. Moscato. Capítulo 3: Algoritmos meméticos. In B. Diaz, editor, Las Nuevas Técnicas Heurísticas y las Redes Neuronales, pages 67–103. Ed. Paraninfo, Madrid, 1996.

[129] E. Landree, C. Collazo-Davila, and L. Marks. Multi-solution memetic algorithm approach to surface structure determination using direct methods. Acta Crystallographica Section B - Structural Science, 53:916–922, 1997.

[130] M. Lauria. High performance mpi implementation on a network of workstations, 1996.

[131] D. Levine. A Parallel Genetic Algorithm for the Set Partitioning Problem. PhD thesis, Illinois Institute of Technology, Department of Computer Science, 1994.

[132] D. Levine, P. Hallstrom, D. Noelle, and B. Walenz. Experiences with the **PGAPack** parallel genetic algorithm library. Technical Memorandum ANL/MCS-TM-228, Argonne National Laboratory, July 1997.

[133] C. M. A survey of practical applications of examination timetabling algorithms. Operational research, 34(2):193–202, March 1986.

[134] N. K. D. M. M. Lozano, F. Herrera. Real coded memetic algorithms with crossover hill-climbing. Evolutionary Computation, 12(3):273–302, 2004.

[135] S. W. Mahfoud. Niching methods for genetic algorithms. PhD thesis, Urbana, IL, USA, 1995.

[136] R. Makinen, P. Neittaanmaki, J. Periaux, M. Sefrioui, and J. Toivanen. Parallel genetic solution for multiobjective mdo, 1997.

[137] M. Manfrin. Parallel metaheuristics on cluster systems for university timetabling problem. [1], pages 300–305.

[138] A. Mendes, F. Muller, P. França, and P. Moscato. Comparing meta-heuristic approaches for parallel machine scheduling problems with sequence-dependent setup times. In Proceedings of the 15th International Conference on CAD/CAM Robotics & Factories of the Future, Aguas de Lindoia, Brasil, volume 1, pages 1–6, Campinas, SP, Brazil, 1999. Technological Center for Informatics Foundation.

[139] P. Merz. On the performance of memetic algorithms in combinatorial optimization. In W. Hart, N. Krasnogor, and J. Smith, editors, Second Workshop on Memetic Algorithms (2nd WOMA), pages 168–173, San Francisco, California, USA, 7 July 2001.

[140] P. Merz. On the performance of memetic algorithms in combinatorial optimization. In W. Hart, N. Krasnogor, and J. Smith, editors, Second Workshop on Memetic Algorithms (2nd WOMA), pages 168–173, San Francisco, California, USA, 7 2001.

[141] P. Merz. A comparison of memetic recombination operators for the traveling salesman problem. In W. B. Langdon, E. Cantú-Paz, K. Mathias, R. Roy, D. Davis, R. Poli, K. Balakrishnan, V. Honavar, G. Rudolph, J. Wegener, L. Bull, M. A. Potter, A. C. Schultz, J. F. Miller, E. Burke, and N. Jonoska, editors, GECCO 2002: Proceedings of the Genetic and Evolutionary Computation Conference, pages 472–479, New York, 9-13 July 2002. Morgan Kaufmann Publishers.

[142] P. Merz and B. Freisleben. Fitness Landscapes, Memetic Algorithms and Greedy Operators for Graph Bi-Partitioning. Technical Report 98-01, University of Siegen, Germany, 1998.

[143] P. Merz and B. Freisleben. On the Effectiveness of Evolutionary Search in High–Dimensional NK-Landscapes. In D. Fogel, editor, Proceedings of the 1998 IEEE International Conference on Evolutionary Computation, pages 741–745, Piscataway, NJ, USA, 1998. IEEE Press.

[144] P. Merz and B. Freisleben. A comparison of memetic algorithms, tabu search, and ant colonies for the quadratic assignment problem. In P. J. Angeline, Z. Michalewicz, M. Schoenauer, X. Yao, and A. Zalzala, editors, Proceedings of the Congress on Evolutionary Computation, volume 3, pages 2063–2070, Mayflower Hotel, Washington D.C., USA, 6-9 1999. IEEE Press.

[145] P. Merz and B. Freisleben. A comparison of Memetic Algorithms, Tabu Search, and ant colonies for the quadratic assignment problem. In Proceedings of the 1999 Congress on Evolutionary Computation, Washington D.C., pages 2063–2070, Piscataway, NJ, 1999. IEEE Service Center.

[146] P. Merz and B. Freisleben. Fitness landscapes and memetic algorithm design. In D. Corne, M. Dorigo, and F. Glover, editors, New Ideas in Optimization, pages 245–260. McGraw-Hill, 1999.

[147] P. Merz and B. Freisleben. Fitness Landscapes, Memetic Algorithms and Greedy Operators for Graph Bi-Partitioning. Evolutionary Computation, 8(1):61–91, 2000.

[148] P. Merz and A. Zell. Clustering gene expression profiles with memetic algorithms. In 7th International Conference on Parallel Problem Solving from Nature - PPSN VII, September 7-11, 2002, Granada, Spain, 2002.

[149] N. Metropolis, A. W. Rosenbluth, R. Rosenbluth, A. H. Teller, and E. Teller. Equations of state calculations by fast computing machine. Journal of Chemical Physics, 21:1087–1091(1092?), 1953.

[150] M. M. Meysenburg. The effect of psuedo-random number generator quality on the performance of a simple genetic algorithm.

[151] M. Mitchell and C. Taylor. Evolutionary computation: An overview. Annual Review of Ecology and Systematics, 20:593–616, 1999.

[152] M. Moore. An accurate and efficient parallel genetic algorithm to schedule tasks on a cluster. [1], pages 145–145.

[153] P. Moscato. On Evolution, Search, Optimization, Genetic Algorithms and Martial Arts: Towards Memetic Algorithms. Technical Report Caltech Concurrent Computation Program, Report. 826, California Institute of Technology, Pasadena, California, USA, 1989.

[154] P. Moscato. On genetic crossover operators for relative order preservation. C3P Report 778, California Institute of Technology, Pasadena, CA 91125, 1989.

[155] P. Moscato. An Introduction to Population Approaches for Optimization and Hierarchical Objective Functions: The Role of Tabu Search. Annals of Operations Research, 41(1-4):85–121, 1993.

[156] P. Moscato. Memetic algorithms: A short introduction. In D. Corne, M. Dorigo, and F. Glover, editors, New Ideas in Optimization, pages 219–234. McGraw-Hill, Maidenhead, Berkshire, England, UK, 1999.

[157] P. Moscato. Memetic algorithms' home page. http://www.densis.fee.unicamp.br /moscato/memetichome.html, 2003. Page Created in February 1996.

[158] P. Moscato, P. Frana, and R. Mendes. NP-opt: an optimization framework for NP problems, July 12 2001.

[159] P. Moscato and M. G. Norman. A Memetic Approach for the Traveling Salesman Problem Implementation of a Computational Ecology for Combinatorial Optimization on Message-Passing Systems. In M. Valero, E. Onate, M. Jane, J. L. Larriba, and B. Suarez, editors, Parallel Computing and Transputer Applications, pages 177–186, Amsterdam, 1992. IOS Press.

[160] P. Moscato and F. Tinetti. Blending heuristics with a population-based approach: A memetic algorithm for the traveling salesman problem. Report 92-12, Universidad Nacional de La Plata, C.C. 75, 1900 La Plata, Argentina, 1992.

[161] H. Mühlenbein and D. Schlierkamp-Voosen. Optimal interaction of mutation and crossover in the breeder genetic algorithm. In Forrest [88], page 648.

[162] A. MunizDeOliveira and L. N. Lorena. Real coded evolutionary approaches to unconstrained numerical problems. In P. Franca, editor, LAPTEC 2002 Proceedings, volume 1 of 1, pages 192–201, Sao Jose dos Campos, Brazil, 2002.

[163] A. C. MunizDeOliveira. Treinamento Populacional em Heuristicas,Applicacoes em Optimizicao. PhD thesis, Ministerio da cienca e tecnologicia, Instituto nacional de pesquisas espaciais, 2003.

[164] M. Nakamaru, H. Nogami, and Y. Iwasa. Score-dependent fertility model for the evolution of cooperation in a lattice. Journal of Theoretical Biology, 194(1):101–124, 1998.

[165] K. Nara. Maintenance scheduling by using the parallel simulated annealing method. IEEE Transactions on Power Systems, 4:840–861, 2003.

[166] NECCompany. School magic. http://www.necsoft.nec.co.jp/soft/smagic.html, 1999-2004. Software.

[167] M. G. Norman and P. Moscato. A competitive and cooperative approach to complex combinatorial search. Technical Report Caltech Concurrent Computation Program, Report. 790, California Institute of Technology, Pasadena, California, USA, 1989. expanded version published at the Proceedings of the 20th Informatics and Operations Research Meeting, Buenos Aires (20th JAIIO), Aug. 1991, pp. 3.15–3.29.

[168] B. Paechter, A. Cumming, M. Norman, and H. Luchian. Extensions to a Memetic timetabling system. In E. Burke and P. Ross, editors, The Practice and Theory of Automated Timetabling, volume 1153 of Lecture Notes in Computer Science, pages 251–265. Springer Verlag, 1996.

[169] G. Patton, Dexter and Punch. On the application of cohort-driven operators to continuous optimization problems using evolutionary computation. Lecture Notes in Computer Science, 1447:671–682, 1998. Springer - Verlag Inc.

[170] C. B. Pettey, M. R. Leuze, and J. J. Grefenstette. A parallel genetic algorithm. pages 155–161, 1987.

[171] G. Pfister. In Search of Clusters. Prentice Hall, 1998.

[172] H. Pohlheim. Competition and cooperation in extended evolutionary algorithms. In E. D. Goodman, editor, 2001 Genetic and Evolutionary Computation Conference Late Breaking Papers, pages 331–338, San Francisco, California, USA, 9-11 2001.

[173] H. Pohlheim and J. Wegener. Testing the temporal behavior of real-time software modules using extended evolutionary algorithms. In W. Banzhaf, J. Daida, E. Agoston, H. G. Max, V. Honavar, M. Jakiela, and R. Smith, editors, Proceedings of the Genetic and Evolutionary Computation Conference, volume 2, page 1795, Orlando, Florida, USA, 13-17 1999. Morgan Kaufmann.

[174] J. Poland, A. Mitterer, K. Knodler, and A. Zell. Genetic algorithms can improve the construction of D-optimal experimental designs. In N. Mastorakis, editor, Advances In Fuzzy Systems and Evolutionary Computation, pages 227–231. World Scientific, Engineering Society Press: New York, Athens, 2001.

[175] P. Preux and E. Talbi. Towards hybrid evolutionary algorithms. International Transactions in Operational Research, 6(6):557–570, 1999.

[176] T. W. R. Knosala. A production scheduling problem using genetic algorithms. Journal of Materials Processing Technology, 109 2001. pp. 90- 95.

[177] N. Radcliffe and P. Surry. Formal Memetic Algorithms. In T. Fogarty, editor, Evolutionary Computing: AISB Workshop, volume 865 of Lecture Notes in Computer Science, pages 1–16. Springer-Verlag, Berlin, 1994.

[178] R. Rankin. Automatic timetabling in practice. In E. Burke and P. Ross, editors, The Practice and Theory of Automated Timetabling, volume 1153 of Lecture Notes in Computer Science, pages 266–279. Springer Verlag, 1996.

[179] I. Rechenberg. Evolutionsstrategie, volume 15 of problemata. Friedrich Frommann Verlag (Günther Holzboog KG), Stuttgart, 1973.

[180] M. G. C. Resende. Metaheuristics: Computer Decision-Making. Kluwer Academic Publishers, 2004.

[181] C. W. Reynolds. Evolution of obstacle avoidance behavior: Using noise to promote robust solutions. In K. E. Kinnear, editor, Advances in Genetic Programming, Complex Adaptive Systems, pages 221–242, Cambridge, 1994. MIT Press.

[182] F. Rothlauf and D. Goldberg. Redundant representations in evolutionary computation. Evolutionary Computation,MIT Press, 11(4), Winter 2003.

[183] A. Ruiz-Andino, L. Araujo, J. Ruz, and F. Sáenz. Parallel evolutionary optimisation with constraint propagation. Lecture Notes in Computer Science, 1498:270, 1998.

[184] R. Salomon. Reevaluating genetic algorithm performance under coordinate rotation of benchmark functions; a survey of some theoretical and practical aspects of genetic algorithms. BioSystems,Elsevier Science, 39(3):263–278, 1996.

[185] R. Salomon. Improving the performance of genetic algorithms through derandomization. Software – Concepts and Tools, Springer-Verlag, Berlin., 18(4):175–184, 1997.

[186] R. Salomon. Benchmarking functions for parallel evolutionary algorithm. BioSystems,Elsevier Science, 25(2):164–178, 1999.

[187] J. Sang, C. M. Kim, T. J. Kollar, and I. Lopez. High-performance cluster computing over gigabit/fast ethernet. Informatica (Slovenia), 23(1), 1999.

[188] T. Satoh and K. Nara. Maintenance scheduling by using the simulated annealing method. IEEE Transactions on Power Systems, 6:850–857, 1991.

[189] D. G. . Schaerf. Multi neighborhood local search with application to the course timetabling problem. In In E. Burke, P. De Causmaecker eds, Proceedings of the

4th International Conference on Practice and Theory of Automated Timetabling (PATAT-2002), 2003.

[190] H. Schwefel. Numerische optierung vin computer-modellen mittels der evolutionsstrategie. Interdisciplinary Systems research, 26, 1977. Birkhauser,Basel.

[191] H. Schwefel and R. Manner, editors. Parallel Problem Solving from Nature, volume 496 of Lecture Notes in Computer Science. Springer, Berlin, 1991.

[192] H. P. Schwefel. Numerische Optimierung von Computer Modellen mittels der Evolutionsstrategie, volume 26 of Interdisciplinary systems research. Birkhäuser Verlag, Basel, 1977.

[193] M. Schwehm. Massively parallel genetic algorithms. In Massively Parallel Processing - Applications and Development', Proc. Int. Conf. MPP '94 in Delft, pages 505–512,. Netherlands, Elsevier Science Publ., 1994.

[194] A. V. Sebald and e. L. J. Fogel, editors. An Introduction to Cultural Algorithms. Proceedings of the Third Annual Conference on Evolutionary Programming, 1994. 131– 139.

[195] N. Senin, D. R. Wallace, and N. Borland. Object-based design modeling and optimization with genetic algorithms. In W. Banzhaf, J. Daida, A. E. Eiben, M. H. Garzon, V. Honavar, M. Jakiela, and R. E. Smith, editors, Proceedings of the Genetic and Evolutionary Computation Conference, volume 2, pages 1715–1722, Orlando, Florida, USA, 13-17 1999. Morgan Kaufmann.

[196] S. Servais and M. Zwick. Ordering parallel genetic algorithm genomes with reconstrability analysis. Technical report, Eastern Washington University and Portland State University, 2003.

[197] A. Singh, J. Schaeffer, and D. Szafron. Views on template-based parallel programming. In CASCON 96 Proceedings, Toronto, October 1996.

[198] W. Spears and K. Dejong. An analysis of multi-point crossover, pages 301–315. Foundation of Genetic Algorithms. Morgan Kaufman, 1991.

[199] L. Spector and S. Luke. Cultural transmission of information in genetic programming. In J. R. Koza, D. E. Goldberg, D. B. Fogel, and R. L. Riolo, editors, Genetic Programming 1996: Proceedings of the First Annual Conference, pages 209–214, Stanford University, CA, USA, 28–31 1996. MIT Press.

[200] T. Starkweather, D. Whitley, and K. Mathias. Optimization using distributed genetic algorithms. In Proceedings of the First International Conference on Parallel Problem Solving from Nature, pages 176–186. Springer, 1991.

[201] O. Steinmann, A. Strohmeier, and T. StG'Otzle. Tabu search vs. random walk. Advances in Artificial Intelligence,Springer Verlag,LNCS, 1997.

[202] E. Talbi. A taxonomy of hybrid metaheuristics. Journal of Heuristics, 8(2):541–564, Sept 2002.

[203] R. Tanese. Parallel genetic algorithms for a hypercube. In Proceedings of the Second International Conferenceon Genetic Algorithms, pages 177–183, 1987.

[204] J. M. Varanelli and J. P. Cohoon. Population-oriented simulated annealing: A genetic/thermodynamic hybrid approach to optimization. In L. J. Eshelman, editor, Proceedings of the 6th International Conference on Genetic Algorithms, pages 174–183, San Francisco, 1995. Morgan Kaufmann Publishers.

[205] C. Voudouris. Guided Local search for combinatorial optimization problems. Phd thesis, Department of computer science, University of Essex, Colchester,UK, July 1997.

[206] C. Voudouris. Guided local search-an illustrative example in function optimisation. Technical report, University of Essex, 1998.

[207] C. Voudouris and E. Tsang. Guided local search. Technical Report CSM-247, Department of Computer Science, University of Essex, 1995.

[208] C. Voudouris and E. Tsang. Partial constraint satisfaction problems and guided local search. In Proceedings of 2nd International Conference on Practical Application on constaint technology, pages 337–356, April 1996.

[209] R. Weber. Knowledge management for computational intelligence systems. In Eight IEEE International Symposium on High Assurance Systems Engineering, College of Information Science Technology Drexel University, March 2004.

[210] I. Wegener. On the design and analysis of evolutionary algorithms. In Proceedings of theWorkshop on Algorithm Engineering as a New Paradigm, pages 36–47, 2000.

[211] I. Wegener. Methods for the analysis of evolutionary algorithms on pseudo-boolean functions. In R. Sarker, M. Mohammadian, and X. Yao, editors, Evolutionary Optimization. Kluwer Academic Publishers, Dordrecht, The Netherlands, 2002.

[212] D. Whitley. A free lunch proof for gray versus binary encodings. In W. Banzhaf, J. Daida, E. Agoston, H. G. Max, V. Honavar, M. Jakiela, and R. Smith, editors, Proceedings of the Genetic and Evolutionary Computation Conference, volume 1, pages 726–733, Orlando, Florida, USA, 13-17 1999. Morgan Kaufmann.

[213] B. Wilkinson and M. Allen. Parallel Programming, Techniques and applications using networked workstations and parallel computers. Prentice Hall, 1998.

[214] T. Yamada and R. Nakano. Scheduling by Genetic Local Search with Multi-Step Crossover. In H.-M. Voigt, W. Ebeling, I. Rechenberg, and H.-P. Schwefel, editors, Proceedings of the 4th Conference on Parallel Problem Solving from Nature - PPSN IV, volume 1141 of Lecture Notes in Computer Science, pages 960–969. Springer, 1996.

[215] C. Zacharias, M. Lemes, and A. Pino. Combining genetic algorithm and simulated annealing: a molecular geometry optimization study. THEOCHEM-Journal of Molecular Structure, 430(29–39), 1998.

[216] I. Zelinka, V. Vasek, K. Kolomaznik, P. Dostal, and J. Lampinen. Memetic algorithm and global optimization. In PC Control 2001, 13th International Conference on Process Control, High Tatras, Slovakia, 2001.

[217] V. Zoriktuev. A parallel evolutionary algorithm for solving the ufa timetabling problem, 2003. UFA State Aviation Technical University, Russian Academy.

[218] Z. Zurn and Z. Quintana. Generator maintenance scheduling via successive approximations dynamic programming. IEEE Transactions on Power Systems, 94(2):665–671, 1975.

[219] M. Zwick, B. Lovell, and J. Marsh. Global optimization studies on the 1-d phase problem. International Journal of General Systems, 25(1):47–59, 1996.

Ευρετήριο

Βιογραφικό

Ο Ιάσων Διγαλάκης γεννήθηκε το 1976 στο Queens της Νέας Υόρκης, στις Η.Π.Α από Έλληνες μετανάστες. Το 1981 ήρθε με την οικογένεια του στην Αθήνα με την διαδικασία του επαναπατρισμού. Το 1990 τελείωσε το 4ο Γυμνάσιο Αθηνών και το 1993 το 3ο Γενικό Λύκειο Χανίων. Πέρασε στο Πανεπιστήμιο Μακεδονίας ένατος με κρατική υποτροφία το Σεπτέμβριο του 1994. Έλαβε το πτυχίο Εφ. Πληροφορικής του Πανεπιστημίου Μακεδονίας τον Ιούλιο του 1998. Το Δεκέμβριο του ίδιου χρόνου ξεκίνησε την διδακτορική του διατριβή. Τα ερευνητικά του ενδιαφέροντα του περιλαμβάνουν μεταευρετικές μεθόδους, μιμητικούς και εξελικτικούς αλγορίθμους, προσομοιούμενη ανόπτηση, αποτρεπτική αναζήτηση, βελτιστοποίηση με αποικίες ψηφιακών μυρμηγκιών, πολιτισμικούς αλγορίθμους, υβριδικές προσεγγίσεις, παράλληλη και κατανεμημένη επεξεργασία. Το δημοσιευμένο έργο του κατά την περίοδο εκπόνησης της διδακτορικής του διατριβής πλαισιώθηκε απο 6 άρθρα σε περιοδικά, 12 άρθρα σε πρακτικά διεθνών συνεδρίων και 4 ανακοινώσεις σε ελληνικά συνέδρια. Υπήρξε συνδεδεμένο μέλος της IEEE για 2 χρόνια. Κατά την διάρκεια της διδακτορικής του διατριβής έλαβε οικονομική ενίσχυση από το Βαρδινογιάννειο Ιδρυμα για το έτος 2001. Από τον Αυγουστο του 2003 ως τον Αύγουστο του 2004 υπηρέτησε τη στρατιωτική του θητεία σε μονάδες και σχηματισμούς της Ορεστιάδας, της Ξάνθης και των Αθηνών. Από το 2000 έως τον Μαιο 2005 εργάστηκε με συμβάσεις έργου σε διάφορα ερευνητικά προγράμματα που έλαβαν χώρα στο Πανεπιστήμιο Μακεδονίας ως προγραμματιστής εφαρμογών και στο ΑΤΕΙ Θεσσαλονίκης από το 2001 ως το 2003 ως έκτακτος εργαστηριακός συνεργάτης.