ΠΟΛΥΤΕΧΝΕΙΟ ΚΡΗΤΗΣ
Τμήμα Ηλεκτρονικών Μηχανικών & Μηχανικών Υπολογιστών

Ανάπτυξη Μεθοδολογίας και Συστήματος Χρονοπρογραμματισμού Ευφυών Πρακτόρων με Χρήση Μεθόδων Εμπνευσμένων από την Φύση.

Διπλωματική Εργασία

Περιστερόπουλος Αθανάσιος

Εξεταστική Επιτροπή:
Σταυρακάκης Γεώργιος (Επιβλέπων)
Λαγουδάκης Μιχαήλ
Ματσατσίνης Νικόλαος

Ευχαριστίες

Η παρούσα διπλωματική εργασία εκπονήθηκε στο Τμήμα Ηλεκτρονικών Μηχανικών και Μηχανικών Υπολογιστών του Πολυτεχνείου Κρήτης , στα πλαίσια των ερευνητικών δραστηριοτήτων του Εργαστηρίου Σχεδιασμού & Ανάπτυξης Συστημάτων Υποστηριξης Αποφάσεων.

Ολοκληρώνοντας αυτή την διπλωματική εργασία αισθάνομαι την ανάγκη να ευχαριστήσω θερμά τον καθηγητή κ. Νικόλαο Ματσατσίνη, για την ευκαιρία που μου έδωσε να εργαστώ σε ένα σύγχρονο αντικείμενο και για την διάθεση συνεργασίας και την μεγαλοθυμία που μου έδειξε καθόλη την διάρκεια εκπόνησης αυτης της εργασίας.

Τέλος, ευχαριστώ την οικογένειά μου για την υποστήριξή της καθόλη τη διάρκεια της φοιτητικής μου πορείας.

Αθανάσιος Περιστερόπουλος,
Ιούνιος 2012

Πρόλογος

Στη κλάση των μεθευρετικών αλγορίθμων ο ACO αποτελεί ένα σχετικά νέο πεδίο.Παρά την επιτυχία του σε εφαρμογές προβλημάτων συνδυαστικής βελτιστοποίησης ,όπως το Πρόβλημα του Πλανώδιου Πωλητή και το Πρόβλημα της Τετραγωνικής Ανάθεσης έχει διεξαχθεί περιορισμένη έρευνα στο πλαίσιο του JSP .Η ελαχιστοποίηση της πλειοψηφίας των κριτιρίων για ένα JSP,όπως η ελαχιστοποίηση του makespan και της συνολικής βραδύτερης περάτωσης, είναι σχετικά απλή από μαθηματικής άποψης και εύκολη προς την μοντελοποίηση της ,ωστόσο λόγω των πολλών « αυστηρών» περιορισμών της ,το JSP έχει λάβει την φήμη ενός ιδιαίτερα δύσκολου προβλήματος προς λύση.Συνεπώς αποτελεί σημείο αναφοράς στην ακαδημαική έρευνα, καθώς αντικατοπτρίζει με συνέπεια την υπολογιστική δυσκολία που συναντάται στην εύρεση του βέλτιστου χρονοπρογράμματος.Η ελαχιστοποίηση λοιπόν των προαναφερθέντων κριτηρίων συνιστά ένα σημαντικό μοντέλο στη θεωρία χρονοπρογραμματισμού που παρέχει το έδαφος για την δοκιμή νέων αλγοριθμικών ιδεών και ένα σημείο εκκίνησης για νέα ,περισσότερο πρακτικά μοντέλα.
Στην συγκεκριμένη διπλωματική εφαρμόσαμε μια κλασσική προσέγγιση των ACO αλγόριθμων ,τον ACS(Ant Colony System) ,στο JSP στοχεύοντας στην ελαχιστοποίηση δύο αντικειμενικών συναρτήσεων, του makespan και της συνολικής βραδύτερης περάτωσης ενός χρονοπρογράμματος.Κατόπιν προτείναμε μια μέθοδο βασισμένη στον single-objective ACS για την επίλυση

του πολυκριτηριακού(multi-objective) JSP, με σκοπό την ταυτόχρονη ελαχιστοποίηση των δύο κριτηρίων.Η μέθοδος αναζητά ευρετικά σχεδόν-βέλτιστα χρονοπρογράμματα τα οποία βελτιστοποιούν ταυτόχρονα τα ζητούμενα κριτήρια ,επιστρέφοντας ένα σύνολο από Pareto βέλτιστες λύσεις.
Υπολογιστικά πειράματα εκτελέστηκαν στο πλαίσιο των προτεινόμενων αλγόριθμων και η απόδοσή τους δοκιμάστηκε σε 7 γνωστά προβλήματα χρονοπρογραμματισμού της βιβλιογραφίας.Τα αποτελέσματα έδειξαν ότι ο ACS μπορεί να προσεγγίσει σε ικανοποιητικό βαθμό τις βέλτιστες θεωρητικές τιμές κάθε κριτηρίου ξεχωριστά ,αλλά και σε πολυκριτηριακό επίπεδο δύναται να βρεί ένα καλά κατανεμημένο σύνολο λύσεων που παρέχει μια ευρεία γκάμα επιλογών στον λήπτη αποφάσεων.

Περιεχόμενα

1 Βασικές Έννοιες 7

2 Υφιστάμενη κατάσταση 8

2.1 Η Βελτιστοποίηση ως Πρόβλημα 8

2.2 Ευφυής πράκτορες & Αλγόριθμοι εμπνευσμένοι από την φύση .9

2.2.1 Ευφυία σε κοινωνικές αποικίες 12

2.2.1.1 Ιστορική αναδρομή και σχετικές θεωρίες.....12

2.2.1.2 Η αυτοοργάνωση ως βιολογική βάση της θεωρίας της νοημοσύνης του σμήνους 15

2.2.1.2 Στιγμεργία ……16

2.2.2 Αλγόριθμοι εμπνευσμένοι από αποικίες μυρμηγκιών …….17

2.2.3 Intergration of MAS and EAs 23

2.2.4 Agent-based Nature inspired algorithms 24

2.3 Χρονοπρογραμματισμός Παραγωγής 25

2.3.1 Χρονοπρογραμματισμός στο Σύστημα παραγωγής 26

2.3.2 Αντικειμενικά Κριτήρια 27

2.4 Προηγούμενη Κατάσταση 29

2.5 Στόχοι της διπλωματικής 34

3 Θεωρητικό υπόβαθρο ……36

3.1 Μοντελοποίηση του Προβλήματος 36

3.2 Αναπαράσταση του Job Shop Problem 39

3.2.1 Γραφική Αναπαράσταση του Job Shop Problem (disjunctive graph model) ……40

3.3 Κανόνες Προτεραιότητας (Dispathcing Rules) ……..42

3.4 Job Shop Benchmark Problems ……43

3.4.1 Στιγμιότυπα των Fisher και Thompson …….....44

3.4.2 Στιγμιότυπα του Lawrence …….....45

3.5 Τοπική Αναζήτηση 46

3.6 Πολυκριτηριακή βελτιστοποίηση …….....50

3.6.1 Ολική Βελτιστοποίηση.....50

3.6.1.1 Αντικειμενικές Συναρτήσεις ενός κριτηρίου (Single Objective Functions……...50

3.6.1.2 Αντικειμενικές Συναρτήσεις πολλαπλών κριτηρίων ……...51

3.6.1.3 Χειρισμός περιορισμών…….....55

3.6.2 Μέτρα Απόδοσης(Performance metrics).....55

3.6.2.1 Μέτρα σύγκλισης…….....55

3.6.2.2 Μέτρα διαφοροποίησης 58

3.6.2.3 Επιπλέον μέτρα.....59

3.6.3 ACO για πολυκριτηριακή βελτιστοποίηση με περιορισμούς 60

3.6.3.1 Αλγοριθμικά συστατικά ενός MOACO αλγορίθμου.....62

3.6.3.1.1 Πολλαπλές αποικίες μυρμηγκιών 63

3.6.3.1.2 Πληροφορία φερομόνης και ευρετική πληροφορία 63

3.6.3.1.3 Προσάθροιση φερομόνης και ευρετικής πληροφορίας….....63

3.6.3.1.4 Αξιολόγηση Pareto μετώπου …… 67

3.6.3.1.4 Ανανέωση φερομόνης ……68

3.6.3.1.5 Αρχείο Pareto …….....69

4 Μεθοδολογία Μοντελοποίησης προτεινόμενου συστήματος

4.1 Σχεδίαση του ACO-πράκτορα 71

4.2 Εφαρμογή του Συστήματος πρακτόρων στο JSP 75

4.3 Κωδικοποίηση λύσεων και έξοδος του συστήματος.....81

4.4 Μοντελοποίηση του συστήματος για πολυκριτηριακή βελτιστοποίηση.....83

5 Ανάλυση και Σχεδίαση συστήματος

5.1 Δεδομένα στο πρόβλημα μας 91

5.2 Ανάλυση και Σχεδίαση Μονοκριτηριακού σύστηματος.....91

5.2.1 Κωδικοποίηση των λύσεων και υπολογισμός των κριτηρίων.....101

5.3 Ανάλυση και Σχεδίαση του Πολυκριτηριακού σύστηματος.....103

6 Πειράματα

6.1 Εφαρμογές στο μονοκριτηριακό σύστημα.....107

6.2 Εφαρμογές στο πολυκριτηριακό σύστημα.....126

7 Συμπεράσματα- Μελλοντική Έρευνα 133

8 Βιβλιογραφία 137

Παράρτημα Α …….141

Παράρτημα Β …….145

Κεφάλαιο 1 Βασικές έννοιες

Ο σκοπός της συγκεκριμένης διπλωματικής εργασίας είναι η εφαρμογή συστήματος βελτιστοποίησης χρονοπρογραμματισμού ευφυών πρακτόρων σε σύστημα παραγωγής κατά παραγγελία (job-shop,JSP). Χρησιμοποιήθηκε η τεχνική των ACO αλγορίθμων με κάποιες μικρές παραλλαγές ως προς τον τρόπο υπολογισμού των πιθανοτήτων μετάβασης ανάμεσα στους κόμβους του κατασκευαστικού γράφου, τον ορισμό της ευρετικής πληροφορίας για κάθε κόμβο, αλλά και την μέθοδο εξάτμισης και ανανέωσης της φερομόνης στα διάφορα μονοπάτια του γράφου .Στη συνέχεια ακολούθησε η προσπάθεια επίλυσης 7 από τα πιο γνωστά πολύπλοκα προβλήματα χρονοπρογραμματισμού με αριθμό μηχανών μεγαλύτερο από 6, ενώ στη δεύτερη φάση του πειράματος σχεδιάστηκε και υλοποιήθηκε μια πολυκριτηριακή εκδοχή του ACO αλγόριθμου για την ταυτόχρονη βελτιστοποίηση των κριτηρίων υπο εξέταση για τα προάναφερθέντα προβλήματα.

Για δεδομένα συστήματα παραγωγής (μ εργασίες, ν μηχανές), έχοντας επιλέξει ως αντικειμενικές συνάρτήσεις το χρόνο ολοκλήρωσης και τον συνολικό χρόνο βραδύτερης περάτωσης όλων των εργασιών, επιδιώκουμε με τη χρήση των ACO αλγορίθμων την ελαχιστοποίησή τους. Αρχική μας πεποίθηση, η οποία φυσικά επιβεβαιώνεται με την εργασία αυτή, ήταν ότι συνδυάζοντας ένα πολύ αποτελεσματικό, αλλά ταυτόχρονα πολύ απαιτητικό ως προς το χρόνο επεξεργασίας ACO αλγόριθμο θα καταφέρουμε να πάρουμε ικανοποιητικές λύσεις με πολύ γρήγορο τρόπο. Το μοντέλο που προτείνεται, μπορεί εύκολα να επεκταθεί και σε προβλήματα μεγαλύτερου μεγέθους, και δεν παραλείπεται η επιβεβαίωση του ότι, σε μια τέτοια περίπτωση, τα αποτελέσματα θα είναι αναλόγως ικανοποιητικά.

*Η παρούσα διπλωματική ακολουθεί την εξής δομή: στο **Κεφάλαιο 2** δίνεται ένας ορισμός του χρονοπρογραμματισμού ως προβλήματος συνδυαστικής βελτιστοποίησης(Π.Σ.Β) και στη συνέχεια ορίζουμε το μοντέλο ενός Π.Σ.Β .Κατόπιν αναλύεται οι έννοια του πράκτορα και ο ρόλος του σε συστήματα πολυπρακτορικών μεθόδων εμπνευσμένων απο την φύση.Ακολούθως γίνεται ιστορική αναδρομή στην εξέλιξη αυτών των μεθόδων, όπως της «θεωρίας του σμήνους» και παρουσιάζονται δύο βασικά τους χαρακτηριστικά:η αυτοοργάνωση και η στιγμεργία.Στο υπόλοιπο κεφάλαιο εισάγουμε τον αναγνώστη στην θεωρία των αλγόριθμων εμπνευσμένων από αποικίες μυρμηγκιών, συνεχίζουμε με την παρουσίαση του γενικού μεθευρετικού αλγορίθμου ACO(Ant Colony Optimization) και κλείνουμε κάνοντας μια συνοπτική παρουσίαση των συστημάτων χρονοπρογραμματισμού παραγωγής.Αναλύονται τα διάφορα αντικειμενικά κριτήρια που μπορεί να θέσουμε ως στόχο βελτιστοποίησης σε ένα τέτοιο σύστημα.Επίσης γίνεται μια εισαγωγική αναφορά στο ειδικό πρόβλημα του χρονοπρογραμματισμού παραγωγής σε περιβάλλον job-shop, το οποίο θα μας απασχολήσει στο υπόλοιπο της διπλωματικής εργασίας.Τέλος ολοκληρώνουμε αυτό το κεφάλαιο με μια ευρεία περίληψη της υφιστάμενη κατάστασης για το JSP τοσο σε μονοκριτηριακο όσο και σε πολυκριτηριακό επίπεδο.*

*Στο **Κεφάλαιο 3** δίνεται η μοντελοποίηση του βασικού JSP ως προβλήματος γραμμικού χρονοπρογραματισμού και παρατίθονται δύο εναλλακτικές αναπαραστάσεις του προβλήματος αυτές που βασίζονται σε διαγράμματα Gantt και σε disjunctive γράφους.Επιπρόσθετα επιχειρείται μια σύντομη ανασκόπηση των κανόνων προτεραιοτήρων και της λειτουργικότητας τους στα συστήματα παραγωγής,κάθώς αποτελούν βασικό συστατικό της υλοποίησης ενός ACO αλγορίθμου σε περιβάλλον job-shop. Στη συνέχεια δίνουμε μια περιγραφή των πιο γνωστών benchmarks της βιβλιογραφίας και εισάγουμε τον αναγνώστη στις βασικές αρχές της τοπικής αναζήτησης στον χρονοπρογραμματισμο παραγωγής ως σκέψη για μελλοντική εργασία.*

Ακολούθως γίνεται μια προσπάθεια προσέγγισης ενός πολυκριτηριακού προβλήματος συνδυαστικής βελτιστοποίησης. Αρχικά παρουσιάζουμε τις βασικές

έννοιες της ολικής και πολυκριτηριακής βελτιστοποίησης και ακολούθως παραθέτουμε ένα σύνολο συγκριτικών μετρων αξιολόγησης που χρησιμοποιούνται ευρέως στην βιβλιογραφία.Γίνεται επίσης μια εκτενής ανάλυση των αλγοριθμικών συστατικών ενός πολυκριτηριακού ACO αλγόριθμου.

*Στο **Κεφάλαιο 4** ξεκινά το πρακτικό κομμάτι της διπλωματικής όπου παρουσιάζουμε την βασική μεθοδολογία επίλυσης προβλημάτων job shop με χρήση των πολυπρακτορικών ACO αλγόριθμων στα πλαίσια της οποίας θα αναλύσουμε την μεθοδολογία μας στο 5^{ο} κεφάλαιο.Το προτεινόμενο σύστημα εφαρμόζεται και στις δύο περιπτώσεις ενός μονοκριτηριακού και πολυκριτηριακού job shop συστήματος.*

*Στο **Κεφάλαιο 5** παρουσιάζουμε την ανάλυση και σχεδίαση του προτεινόμενου συστήματος που αναπτύξαμε στο προηγούμενο καφάλαιο , τις διάφορες σχεδιαστικές λεπτομέρειες και τον τρόπο υλοποίησης τους.*

*Στο **Κεφάλαιο 6** στη 1^{η} φάση προχωρήσαμε στη εκτέλεση της μεθοδολογίας μας το προτεινόμενο ACO πολυπρακτορικό σύστημα στο πρόβλημα χρονοπρογραμματισμού εργασιών για 7 διαφορετικά προβλήματα χρονοπρογραμματισμού Ft06, Ft10 ,La01, La06, La10, La21 και Orb04.Υπολογίσαμε τα σχετικά σφάλματα για την περίπτωση της ελαχιστοποίησης του makespan και συγκρίναμε τα αποτελεσματά μας με μια ευρετική μέθοδο παραγωγής χρονοπρογραμμάτων στην περίπτωση του συνολικού χρόνου βραδύτερης περάτωσης, βάση των οποίων κάναμε κάποιες γενικές παρατηρήσεις,όσον αφορά την απόδοση του ACO σε σχέση με τις διαστάσεις ενός JSP.Στη 2^{η} φάση εφαρμόσαμε την πολυκριτηριακή εκδοχή του ACO συστήματος στα παραπάνω προβλήματα και αξιολογήσαμε την απόδοσή του βάση συγκεκριμένων ποιοτικών κριτηρίων.*

*Ολοκληρώνουμε την διπλωματική εργασία με το **Κεφάλαιο 7** ,όπου εξάγουμε κάποια γενικά συμπεράσματα για την ποιότητα του ACO συτήαμτός μας τόσο για τη μονοκριτηριακή όσο και για την πολυκριτηριακή περίπτωση.Τέλος παραθέτουμε σκέψεις για μελλοντική έρευνα πάνω στο αντικείμενο με σκοπό την βελτίωση της απόδοσης του συστήματος σε προβλήματα χρονοπρογραμματισμού παραγωγής*

Κεφάλαιο 2 Υφισταμένη Κατάσταση

2.1 Η Βελτιστοποίηση ως Πρόβλημα

Ο όρος 'χρονοπρογραμματισμός έχει οριστεί ως "η κατανομή δεδομένων πόρων στη διάρκεια του χρόνου με σκοπό την ολοκλήρωση ενός συνόλου εργασιών". Τα προβλήματα χρονοπρογραμματισμού δεν αφορούν αποκλειστικά συστήματα παραγωγής.Παρόμοιες καταστάσεις συμβαίνουν σε μεγάλο βαθμό σε εκδοτικούς οίκους , πανεπιστήμια, νοσοκομεία, μεταφορικές εταιρείες κλπ. Τα προβλήματα χρονοπρογραμματισμού τυπικά θεωρούνται NP-hard, δηλαδή η εύρεση μιας λύσης είναι αδύνατη χωρίς την χρήση απαριθμητικών αλγορίθμων ,ενω ο χρόνος εκτέλεσης του αλγορίθμου αυξάνει εκθετικά με το μέγεθος του προβλήματος.Ο χρονοπρογραμματισμός ,σε γενικές γραμμές , ασχολείται με την ανάθεση πόρων (resources) σε εργασίες(tasks) στη διάρκεια του χρόνου(Siang Lyn,2005).Μπορεί να θεωρηθεί ως μια διαδικασία λήψης αποφάσεων με σκοπό την βελτιστοποίηση μιας ή περισσοτέρων αντικειμενικών συναρτήσεων και διαδραματίζει σημαντικό ρόλο σε κατασκευαστικά συστήματα όπου οι μηχανές ,το εργατικό δυναμικό ,οι εγκαταστάσεις και ο χρόνος αποτελούν κρίσιμους πόρους στις διαδικασία παραγωγής.Η ιδιαίτερη αυτή σημασία του χρονοπρογραμματισμού τον έχει καταστήσει ώς ένα από τα περισσότερο μελετημένα προβλήματα συνδυαστικής βελτιστοποίησης(Π.Σ.Β).
Η επίλυση ενός Π.Σ.Β έγγυται στην εύρεση των βέλτιστων λύσεων από ένα πεπερασμένο ή άπειρο αριθμό εναλλακτικών λύσεων (Papadimitriou and Steiglitz, 1982).Ένα Π.Σ.Β μπορεί να είναι πρόβλημα ελαχιστοποίησης ή μεγιστοποίησης και το μοντέλο του ορίζεται ως εξής(Dorigo & Socha,2006):

Ορισμός :*Το μοντέλο $P=(\mathbf{S}, \mathbf{\Omega}, f)$ ενός Π.Σ.Β περιλαμβάνει:*

- *ένα χώρο αναζήτησης* $\mathbf{S}$ *ορισμένο σε ένα πεπερασμένο σύνολο διακριτών μεταβλητών απόφασης και ένα σύνολο περιορισμών* $\mathbf{\Omega}$ *ανάμεσα στις μεταβλητές.*
- *μια αντικειμενική συνάρτηση* $f : \mathbf{S} \rightarrow \mathrm{R}_0^+$ *προς βελτιστοποίηση.*

Ο χώρος αναζήτησης $\mathbf{S}$ *ορίζεται ακολούθως: Έστω δίνεται ένα σύνολο διακριτών μεταβλητων* $\boldsymbol{X_i}$ *,* $i=1, \ldots, n$ *με τιμές* $u_i^j \in \mathbf{D_i} = \{ u_i^1, \ldots, u_i^{|\mathbf{Di}|} \}$.
Το στιγμιότυπο μιας μεταβλητής ,δηλαδή η ανάθεση μιας τιμής u_i^j *σε μια μεταβλητή* X_i *παρίσταται ως* $X_i \leftarrow u_i^j$.
Μια λύση λοιπόν $s \in \mathbf{S}$ *– πχ. μια πλήρης ανάθεση των τιμών στις μεταβλητές απόφασης – η οποία ικανοποιεί όλους τους περιορισμούς του συνόλου* $\mathbf{\Omega}$ *,συνιστά μια εφικτή λύση του Π.Β.Σ.Εάν το σύνολο* $\mathbf{\Omega}$ *είναι κενό,το* P *καλείται μη περιοριστικό μοντέλο, διαφορετικά λέγεται ότι είναι περιοριστικό.*
Μια λύση $s^* \in \mathbf{S}$ *καλείται ολικό βέλτιστο ,αν,* $f(s^*) \leq f(s)$ *,*$\forall\ s \in \mathbf{S}$ *θεωρώντας προβλήματα ελαχιστοποίησης και αντίστοιχα θα ισχύει ,* $f(s^*) \geq f(s)$ *,*$\forall\ s \in \mathbf{S}$ *για προβλήματα μεγιστοποίησης.Το σύνολο όλων των ολικών βέλτιστων λύσεων συμβολίζεται ως* $\mathbf{S}^* \subseteq \mathbf{S}$*. Συνοψίζοντας η λύση ενός Π.Β.Σ έγκειται στην εύρεση τουλάχιστον ενός* $s^* \in \mathbf{S}^*$.

Συνδυαστικά προβλήματα βελτιστοποίησης μπορούν να λυθούν χρησιμοποιώντας ακριβής ή πλήρης αλγόριθμους ,ενώ η δεύτερη προσέγγιση χαρακτηρίζεται απο την χρήση προσεγγιστικών αλγορίθμων,οι οποίοι διακρίνονται σε ευρετικούς και

μεθευρετικούς.Οι ακριβής αλγόριθμοι ανήκουν σε μια ευρύτερη κατηγορία αυτή των ντετερμινιστικών αλγορίθμων με κυριότερα παραδείγματα τους branch and bound αλγόριθμους μικτού ακέραιου προγραμματισμού(mixed integer programming) αλγόριθμοι και μέθοδους αποσύνθεσης(decomposition methods),ενώ από την άλλη χαρακτηριστικά δείγματα προσεγγιστικών αλγορίθμων είναι οι Γενετικοί αλγόριθμοι και οι παραλλαγές τους.

Γενικά οι αλγόριθμοι βελτιστοποίησης χωρίζονται σε δύο κλάσεις(Weisse,2009):

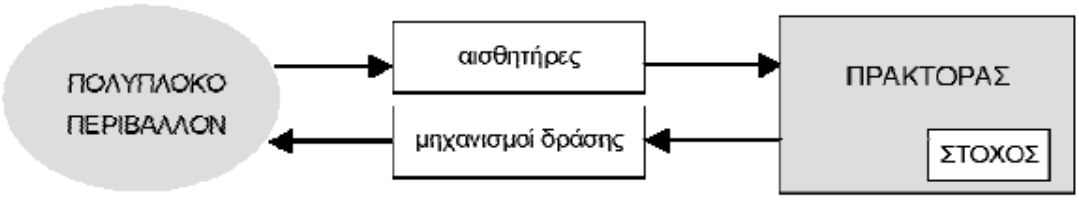
- *Τους **ντετερμινιστικούς αλγόριθμους** ,οι οποίοι χρησιμοποιούνται συχνότερα όταν υπάρχει μια σαφή σχέση ανάμεσα στα χαρακτηριστικά των πιθανών λύσεων και της χρησιμότητάς τους, δεδομένου ενός συγκεκριμένου προβλήματος.Τότε ο χώρος αναζήτησης μπορεί να εξερευνηθεί επαρκώς με χρήση ενός «διαίρει και βασίλευε» σχήματος(divide and conquer).Εάν ωστόσο η σχέση ανάμεσα σε μια υποψήφια λύση και τη «καταλληλότητάς» της δεν είναι τόσο ξεκάθαρη ή οι διαστάσεις του χώρου λύσεων είναι πολλές ,είναι δύσκολο για ένα πρόβλημα να λυθεί ντετερμινιστικά ,καθώς οδηγούμαστε σε μια πλήρη αναζήτηση του χώρου λύσεων που ίσως να μην είναι εφικτή ακόμα και για προβλήματα μικρού μεγέθους.*
- *Τους **στοχαστικούς αλγόριθμους** ,οι οποίοι βασίζονται σε προσεγγίσεις Monte-Carlo.Οι αλγόριθμοι αυτοί θυσιάζουν την βελτιστότητα της λύσης για να επιτύχουν μικρότερους υπολογιστικούς χρόνους χωρίς αυτό να σημαίνει ότι το αποτέλεσμα θα είναι λάθος –μπορεί να καταλήξουν σε ένα τοπικό βέλτιστο.*

*Οι **ευρετικές συναρτήσεις** στην συνδιαστική βελτιστοποίηση είναι συναρτήσεις που χρησιμοποιούνται για να καθορίζουν πιο σύνολο πιθανών λύσεων θα εξεταστεί επόμενα κατα την διάρκεια της αναζήτησης.Οι ντετερμινιστικοί αλγόριθμοι συνήθως τις χρησιμοποιούν ούτως ώστε να καθορίσουν τη σειρά επεξεργασίας των υποψήφιων λύσεων,ενώ οι στοχαστικές μεθόδοι από την άλλη για να εξετάσουν μόνο συγκεκριμένα στοιχεία του χώρου λύσεων που έχουν επιλεχθεί απο τον ευρετική συνάρτηση.*
***Ορισμός :**Μια ευρετική συνάρτηση αποτελεί το μέρος εκείνο ενός αλγόριθμου βελτιστοποίησης που χρησιμοποιεί την υπάρχουσα πληροφορία που έχει συλλεχθεί εως τώρα για να επιλέξει την επόμενη προς εξέταση υποψήφια λύση ή το πως αυτή θα παραχθεί.*
***Ορισμός**: Μεθευρετική είναι η μέθοδος επίλυσης πολύ γενικών κλάσεων προβλημάτων .Συνδυάζει τις διάφορες αντικειμενικές ή ευρετικές συναρτήσεις κατα ένα αφαιρετικό και αποδοτικό τρόπο χωρίς να εισχωρεί βαθύτερα στη δομή τους.*

2.2 Ευφυής πράκτορες & Αλγόριθμοι εμπνευσμένοι από την φύση

Η τεχνητή νοημοσύνη είναι το πεδίο της επιστήμης των υπολογιστών που αφορά στο συνδυασμό της υπολογιστικής ευφυΐας και της βιολογικής νοημοσύνης. Έχει τις ρίζες τις στην βιολογική κατανόηση και χρησιμοποιεί «αρχές» από τη φύση προκειμένου να καταφέρει να δώσει λύσεις. Χρησιμοποιώντας, λοιπόν, αυτή την κατανόηση της φύσης ο βασικός στόχος της τεχνητής νοημοσύνης είναι να καταφέρει να κάνει τους υπολογιστές πιο χρήσιμους στην επίλυση προβλημάτων.
Ο τομέας έχει δείξει ιδιαίτερο ενδιαφέρον στην μελέτη και σχεδίαση των χαρακτηριστικών ενός ευφυούς πράκτορα,όπως την εκμάθηση ,σχεδιασμό,λήψη αποφάσεων και την αντίληψη. Ως πράκτορας ή αντιπρόσωπος (agent) ορίζεται η οντότητα εκείνη που αντιλαμβάνεται το περιβάλλον μέσα στο οποίο βρίσκεται με τη

βοήθεια αισθητήρων (sensors), είναι μέρος του περιβάλλοντος αυτού, κανει συλλογισμούς για το περιβάλλον και δρα πάνω σε αυτό με τη βοήθεια μηχανισμών δράσης (actuators), για την επίτευξη των ζητούμενων στόχων. [Wooldridge&Jennings,1999]. Με την βοήθεια των πρακτόρων ο χρήστης δεν επικοινωνεί πλέον άμεσα με την εφαρμογή αλλά αφήνει εκείνους ως αντιπροσώπους του να τον διευκολύνουν ιδιαίτερα σε χρονοβόρες διαδικασίες και διαδικασίες ρουτίνας. Οι πράκτορες χρησιμοποιούνται σε πληθώρα εφαρμογών, όπως στα προβλήματα έξυπνης παροχής υπηρεσιών βοήθειας και στα προβλήματα χρονοπρογραμματισμού.



Εικόνα 2.1 : Αλληλέπιδραση με το περιβάλλον(Βλαχαβάς,2006).

Η σχεδίαση ενός πράκτορα εξαρτάται άμμεσα από τους προτιθέμενους σκοπούς του και το περιβάλλον στα πλαίσια του οποίου ενεργεί.Οι Russel&Norvig(2003) πρότειναν το σχήμα Αντιλήψεις(Percepts), Ενέργειες(Actions), Στόχοι(Goals) και Περιβάλλον(Enviroments), (PAGE) για να περιγράψουν τις προθέσεις ενός πράκτορα και να δομήσουν την σχεδιασή του.Οι Αντιλήψεις ορίζονται ως η πληροφορία την οποία ένας πράκτορας είναι ικανός να αποσπάσει από το περιβάλλον του.Οι Ενέργειες συνιστούν τα μέσα που χρησιμοποιεί ο πράκτορας για να επιτύχει τους σκοπούς του.Οι Στόχοι είναι οι επιδράσεις των ενεργειών του στο περιβάλλον και τα μέτρα βάση των οποίων αξιολογείται η απόδοσή του.Τέλος το Περιβάλλον περιλαμβάνει την πληροφορία που αντιλαμβάνεται ο πράκτορας και αποτελεί το σκηνικό μέσα στο οποίο ενεργεί.Τέλος ένα μέτρο απόδοσης(performance measure) ενσωματώνει το κριτήριο για την επιτυχημένη συμπεριφορά ενός πράκτορα.Ένας ορθολογικός πράκτορας λοιπόν θα πρέπει να εφοδιαστεί με την γνώση των ενεργειών οι οποίες θα μεγιστοποιήσουν το μέτρο απόδοσής του ,δοσμένου ενός συνόλου αντιλήψεων.Αυτή η γνώση ορίζει το πως οι αντιλήψεις αντιστοιχίζουν σε (mapped onto) ενέργειες και είναι απαραίτητη για ένα πράκτορα ώστε να διατηρήσει τις ιδιότητες της ορθολογικότητας και αυτονομίας(Russel&Norvig,2003).Η PAGE περιγραφή θα χρησιμοποιηθεί στο 3^{ο} κεφάλαιο ώστε να αναγνωρίσουμε μια κατάλληλη αρχιτεκτονική και περιγραφή του περιβάλλοντος ενός πράκτορα εμπνευσμένου από τις αποικίες μυρμηγκιών, πάνω στην οποία θα σχεδιάσουμε το μοντέλο του πράκτορά μας.

Ο σχεδιασμός ενός πράκτορα εξαρτάται σε μεγάλο βαθμό από το περιβάλλον στο οποίο ενεργεί(Wooldridge, 2002).Οι Russel&Norvig(2003,σελ. 45) ορίζουν συνοπτικά τη σχέση ενός πράκτορα και του περιβάλλοντος του ως εξής:"οι ενέργειες περατώνονται από τον πράκτορα στο περιβάλλον,το οποίο ακολούθως παρέχει αντιλήψεις στον πράκτορα ".Πρός αυτή την κατεύθυνση θα πρέπει να κατανοήσουμε πλήρως τις ιδιότητες του μοντέλου περιβάλλοντος ,ούτως ώστε να σχεδιάσουμε έναν πράκτορα με τους κατάλληλους αισθητήρες και μηχανισμούς δράσης για την επίτευξη των στόχων του.

Σύμφωνα με τους Russel&Norvig(2003) η ποικιλία των περιβαλλόντων εργασιών που μπορεί να προκύψουν στη τεχνητή νοημοσύνη είναι προφανώς τεράστια.Μπορούμε όμως να προσδιορίσουμε ένα σχετικά μικρό αριθμό διαστάσεων με βάση τις οποίες

μπορούν τα περιβάλλοντα εργασιών να καταταχθούν σε κατηγορίες.Οι διαστάσεις αυτές προσδιορίζουν σε μεγάλο βαθμό την σχεδίαση κατάλληλων πρακτόρων και την καταλληλότητα της κάθε κύριας οικογένειας τεχνικών για την υλοποίηση πρακτόρων.Παρακάτω παραθέτουμε έναν άτυπο ορισμό αυτών των διαστάσεων :

- ***Πλήρως παρατηρήσιμο*** *ή* ***μερικώς παρατηρήσιμο****.*

Αν οι αισθητήρες ενός πράκτορα του παρέχουν πρόσβαση στην πλήρη κατάσταση του περιβάλλοντος σε κάθε χρονική στιγμή ,τότε λέμε ότι το περιβάλλον εργασιών είναι πλήρως παραρτηρήσιμο(fully observable).Από την άλλη σε ένα μερικώς παρατηρήσιμο περιβάλλον ο πράκτορας δεν έχει πλήρη γνώση λόγω της ανακρίβειας των αισθητήρων ή επειδή απλώς κάποια μέρη της κατάστασης λείπουν από τα δεδομένα των αισθητήρων.

- ***Αιτιοκρατικό*** *ή* ***στοχαστικό***

Αν η επόμενη κατάσταση του περιβάλοντος προσδιορίζεται πλήρως από την τρέχουσα κατάσταση και από την ενέργεια που εκτελείται από τον πράκτορα ,τότε λέμε ότι το περιβάλλον είναι αιτιοκρατικό(deterministic) αλλίως το περιβάλλον είναι στοχαστικό.Θεωρητικά σε ένα πλήρως παρατηρήσιμο αιτιοκρατικό περιβάλλον ένας πράκτορας δεν έχει να αντιμετωπίσει καμία αβεβαιότητα.Αν όμως το περιβάλλον είναι μερικώς παρατηρήσιμο ,τότε μπορεί να εμφανίζεται ως στοχαστικό.Αυτο ισχύει ιδιαίτερα αν το περιβάλλον είναι περίπλοκο,πράγμα που δυσκολεύει την παρακολούθηση όλων των μη παρατηρούμενων απόψεων.Για αυτό είναι συχνά καλύτερο να χαρακτηρίζουμε ένα περιβάλλον αιτιοκρατικό ή στοχαστικό από την άποψη του πράκτορα.

- ***Επεισοδιακό*** *ή* ***ακολουθιακό***

Σε ένα επεισοδιακό περιβάλλον εργασιών η εμπειρία του πράκτορα αποτελείται από ατομικά επεισόδια.Σε κάθε επεισόδιο ο πράκτορας αντιλαμβάνεται και έπειτα πραγματοποιεί μια μεμονωμένη ενέργεια.Στα επεισοδιακά περιβάλλοντα τη επιλογή της ενέργειας σε κάθε επείσόδιο εξαρτάται μόνο από το ίδιο το επεισόδιο.Στα ακολουθιακά περιβάλλοντα από την άλλη η τρέχουσα απόφαση μπορεί να επηρεάζει όλες τις μελλοντικές αποφάσεις.Συνοψίζοντας τα επεισοδιακά περιβάλλοντα είναι πολύ απλούστερα από τα ακολουθιακά,επειδή ο πράκτορας δεν χρειάζεται να σκέπτεται για το μέλλον.

- ***Στατικό*** *ή* ***δυναμικό***

Αν το περιβάλλον μπορεί να αλλάζει ενώ ένας πράκτορας σκέπτεται τότε λέμε ότι το περιβάλλον είναι δυναμικό για αυτόν τον πράκτορα ,διαφορετικά το περιβάλλον είναι στατικό.Τα στατικά περιβάλλοντα αντιμετωπίζονται ευκολότερα επειδή ο πράκτορας δεν χρειάζεται να κοιτάζει τον κόσμο ενώ αποφασίζει για μια ενέργεια,ούτε χρειάζεται να τον απασχολεί το πέρασμα του χρόνου.Τα δυναμικά περιβάλλοντα από την άλλη ζητούν συνεχώς από τον πράκτορα να αποφασίσει τι θα κάνει –αν δεν έχει αποφασίσει ακόμα τότε αυτό θεωρείται ως απόφαση να μην κάνει τίποτα.

- ***Διακριτό*** *ή* ***συνεχές***

Η διάκριση μεταξύ διακριτού και συνεχούς μπορεί να εφαρμοστεί στην κατάσταση του περιβάλλοντος στον τρόπο που αντιμετωπίζεται ο χρόνος και στίς αντιλήψεις και τις ενέργειες του πράκτορα.

2.2.1 Ευφυία σε κοινωνικές αποικίες

Όπως αναφέραμε παραπάνω ,χαρακτηριστικά όπως συνεργασία, εκμάθηση και αυτονομία αποτελούν θεμελιώδη στοιχεία ενός πρακτόρα τα οποία μπορούν προσομοιωθούν μέσα από μεθόδους εμπνευσμένες από βιολογικές διαδικασίες.Τέτοιες φυσικές διαδικασίες ,ιδιαίτερα κοινωνικές αποικίες(social colonies), εμπνέουν τους επιστήμονες για την εύρεση νέων αλγόριθμων που βοηθούν στην μελέτη της τεχνητής νοημοσύνης για την δημιουργία ευφυών πρακτόρων ή πραγματικών συστημάτων που μπορούν να προσομειώσουν συμπεριφορές τις οποίες οι άνθρωποι θεωρούν ευφυείς(εκμάθηση ,λήψη αποφάσεων κλπ.,(Onet Vladu,2010).
*Στη συνέχεια θα παρουσιάσουμε κάποιους από τους βασικότερους αλγόριθμους οι οποίοι αποτελούν σημείο αναφοράς σε αυτές τις μελετές:Particle Swarm Optimization(PSO),Genetic Algorithms,Ant Colony Optimization(ACO).Ο κάθε ένας από τους αλγόριθμους ακολουθεί κάποιο μοντέλο φυσικού φαινομένου ή φυσικής διαδικασίας βασισμένο στο πληθυσμό(**population-based**) και σκοπός τους είναι να βρούν το ολικό βέλτιστο μιας αντικειμενικής συνάρτησης ορισμένη σε δοσμένο χώρο(χώρος αναζήτησης).Οι αλγόριθμοι PSO και ACO ανήκουν σε μια ευρύτερη οικογένεια μεθευρετικών αλγορίθμων αυτών της θεωρίας «νοημοσύνης του σμήνους»,ενώ οι γενετικοί αλγόριθμοι απότελούν υποκατηγορία της οικογένειας των Εξελικτικών Αλγορίθμων.*
Η βασική ιδέα των μεθόδων αυτών είναι η εξής: πράκτορες που αποτελούν στοιχεία μιας κοινωνικής ομάδας διατηρούν μια γνώμη που αποτελεί με τη σειρά της στοιχείο ενός χώρου πεποίθησης(χώρος αναζήτησης) προσβάσιμο από όλο τον πληθυσμό.Οι πράκτορες δύναται να τροποποιήσουν την παρούσα κατάσταση πεποιθήσεως βασιζόμενα στους εξής παράγοντες(Onet Vladu,2010):

- *Την γνώση του περιβάλλοντος(η τιμή της συνάρτησης καταλληλότητας)*
- *Την πρωτύτερη ιστορία καταστάσεων του πράκτορα(την μνήμη του).*
- *Την πρωτύτερη ιστορία καταστάσεων των γειτονικών πρακτόρων.*

Η γειτονιά ενος πράκτορα μπορεί να οριστεί αναλόγως με τον ορισμό του «κοινωνικού δικτύου» του πράκτορα.Ακολουθώντας λοιπόν συγκεκριμένους κανόνες αλληλεπίδρασης οι πράκτορες στον πληθυσμό προσαρμόζουν σταδιακά τις πεποιθήσεις τους ,σε αυτές που επιδεικνύουν την μεγαλύτερη επιτυχία στο κοινωνικό τους δίκτυο.Ως αποτέλεσμα κατα τη διάρκεια το χρόνου δημιουργείται ένας νέος πληθυσμός στον οποίο οι γνώμες των διαφόρων πρακτόρων είναι στενά συσχετισμένες.Οι βασικές αρχές της θεωρίας σμήνους θα παρουσιαστούν στη συνέχεια.

2.2.1.1 Ιστορική αναδρομή και σχετικές θεωρίες

Η «νοημοσύνη του σμήνους» (swarm Intelligence) δεν είναι η πρώτη απόπειρα μίμησης φυσικών ή βιολογικών συστημάτων στην επίλυση προβλημάτων. Τα νευρωνικά δίκτυα και οι γενετικοί αλγόριθμοι αποτελούν ίσως τις πιο γνωστές μεθόδους επίλυσης προβλημάτων με βιολογική έμπνευση.Ο όρος «νοημοσύνη του σμήνους» χρησιμοποιήθηκε από τους Beni & Wang το 1989 ως ο τίτλος μιας εργασίας τους που παρουσιάστηκε σε συνάντηση των επιστημόνων της ρομποτικής. Στο άρθρο τους αυτό οι Beni & Wang αναφερόταν στα κυψελωτά ρομποτικά συστήματα (cellular robotic systems) τα οποία βασίζονται στα κυψελωτά

αυτόματα (cellular automata). Ένα κυψελωτό αυτόματο αποτελείται από ένα άπειρο πλέγμα κυψελών.Κάθε μία κυψέλη βρίσκεται κάθε χρονική στιγμή (ο χρόνος στα κυψελωτά αυτόματα είναι διακριτός) σε μία από τις καταστάσεις που περιγράφονται σε ένα πεπερασμένο σύνολο καταστάσεων. Η κατάσταση αυτή της κυψέλης εξαρτάται άμεσα από την κατάσταση των γειτονικών του κυψελών στην αμέσως προηγούμενη χρονική στιγμή. Οι καταστάσεις όλων των κυψελών επηρεάζονται από τους ίδιους κανόνες και ανανεώνονται σε κάθε αλλαγή του χρόνου. Τα κυψελωτά αυτόματα θεωρούνται πρόγονος της νοημοσύνης του σμήνους επειδή σε αυτά προβάλλεται η αυτοοργάνωση ως μέθοδος επίλυσης προβλημάτων και επιπλέον επειδή αποτέλεσαν έμπνευση για τη θεωρία της «βελτιστοποίησης σμήνους σωματιδίων (Particle Swarm Optimization). Επειδή η ελληνική μετάφραση δεν φαντάζει δόκιμη, στη συνέχεια αυτής της εργασίας θα αναφερόμαστε στη θεωρία αυτή με τα αρχικά PSO. Η PSO λοιπόν, θεωρείται (μαζί με τον αλγόριθμο Ant Colony Optimization που περιγράφουμε παρακάτω στο κεφάλαιο) από τις κυριότερες εκφάνσεις της νοημοσύνης του σμήνους.

- ***Particle Swarm Optimization***

Η PSO είναι μία στοχαστική τεχνική βελτιστοποίησης η οποία βασίζεται στην εξέλιξη ενός πληθυσμού πρακτόρων και αναπτύχθηκε από τους Eberhart και Kennedy το 1995 οι οποίοι εμπνεύστηκαν από την κοινωνική συμπεριφορά που επιδεικνύει ένα σμήνος πουλιών ή ένα κοπάδι ψαριών.Σαν θεωρία η PSO έχει αρκετά κοινά σημεία με άλλους εξελικτικούς αλγορίθμους, όπως οι γενετικοί αλγόριθμοι: στην αρχική του κατάσταση, στο σύστημα υπάρχει ένας τυχαίος πληθυσμός πρακτόρων που αντιπροσωπεύουν τυχαίες λύσεις ο οποίος αξιολογείται από μία συνάρτηση προσαρμογής, ενώ η αναζήτηση του βέλτιστου πραγματοποιείται εξελίσσοντας τον πληθυσμό αυτό.Βέβαια, σε αντίθεση με τους γενετικούς αλγορίθμους η PSO δεν εφαρμόζει κάποιους εξελικτικούς τελεστές όπως η διασταύρωση ή η μετάλλαξη. Στην PSO οι πράκτορες, οι οποίοι αποκαλούνται σωματίδια (Particles), «ίπτανται» στο χώρο των λύσεων ακολουθώντας τα τρέχοντα βέλτιστα σωματίδια. Κάθε σωματίδιο έχει μία θέση στον πολυδιάστατο χώρο και μία ταχύτητα. Το σωματίδιο καταγράφει τις συντεταγμένες του στο χώρο.Οι συντεταγμένες αυτές αναφέρονται στη βέλτιστη λύση που έχει βρει το συγκεκριμένο σωματίδιο μέχρι τώρα. Η τιμή της βέλτιστης λύσης υπολογίζεται με μία συνάρτηση προσαρμογής (fitness) και καταγράφεται επίσης. Η τιμή αυτή ονομάζεται "p-best". Επιπλέον το σύστημα υπολογίζει και μία άλλη τιμή, την "l-best" η οποία είναι η βέλτιστη ανάμεσα στις τιμές "p-best" του σωματιδίου και των γειτονικών του σωματιδίων. Αν βέβαια θεωρήσουμε ως γειτονικά σωματίδια το σύνολο των σωματιδίων, τότε η τιμή αυτή είναι ολικά βέλτιστη και καλείται "g-best". Τα σωματίδια ενός σμήνους επικοινωνούν μεταξύ τους διαδίδοντας τη γνώση τους για τις βέλτιστες λύσεις.Το σύστημα εξελίσσεται καθώς τα σωματίδια ανανεώνουν τη θέση και την ταχύτητα τους σε κάθε επανάληψη σύμφωνα με τις παρακάτω σχέσεις:

$$x \leftarrow x + \upsilon \qquad (2.1)$$

$$\upsilon \leftarrow \omega\, \upsilon + c_1 r_1 (\quad - x) + c_2 r_2 (\quad {}_g - x) \qquad (2.2)$$

Όπου

- x *είναι η θέση του σωματιδίου*
- υ *είναι η ταχύτητα του*
- ω *είναι η σταθερά της αδράνειας.Συνήθως οι τιμές της είναι λίγο μικρότερες της μονάδας.*

- c_1 και c_2 είναι σταθερές που προσανατολίζουν το σωματίδιο προς τις βέλτιστες λύσεις.Οι τιμές τους είναι επίσης κοντά στη μονάδα.
- r_1 και r_2 είναι τυχαίες τιμές στο διάστημα [0,1].
- είναι η τιμή p-best που αναφέραμε προηγουμένως.
- $_g$ είναι η τιμή g-best. Αν λαμβάνουμε υπόψη μόνο ορισμένους γείτονες τότε η τιμή αυτή μπορεί να είναι και η l-best.

Παρατηρούμε πως η θεωρία στηρίζει την αποτελεσματικότητα της στη συνολική γνώση της κοινωνίας (εν προκειμένω του σμήνους)και στις αλληλεπιδράσεις (επικοινωνία) μεταξύ των μελών της. Η PSO φαντάζει ελκυστική ως τεχνική βελτιστοποίησης αφενός γιατί το υπολογιστικό της κόστος είναι μικρότερο από άλλες μεθόδους και γιατί οι παράμετροι που πρέπει να υπολογιστούν είναι σχετικά λίγες και αφετέρου γιατί η εφαρμογή της στη φύση φαίνεται αποτελεσματική.Όπως ένα σμήνος πτηνών βρίσκει τροφή (δηλαδή βέλτιστη τοποθεσία με τις κατάλληλες παραδοχές) χωρίς να γνωρίζει κανένα στοιχείο εκ των προτέρων για την τοποθεσία αυτή, έτσι οι επιστήμονες του χώρου φιλοδοξούν να ευρίσκουν βέλτιστες λύσεις σε προβλήματα με τεράστιους χώρους λύσεων.

- ***Γενετικοί Αλγόριθμοι - Genetic Algorithms***

Οι Γενετικοί Αλγόριθμοι αποτελούν μια υποκατηγορία των Εξελικτικών Αλγορίθμων οι οποίοι βασίζονται στη θεωρία εξέλιξης των ειδών του Δαρβίνου. Σύμφωνα με τη θεωρία του Δαρβίνου, η φυσική εξέλιξη των ατόμων ενός πληθυσμού βασίζεται στη φυσική διαλογή (σε όλα τα είδη των ζώων πιθανότερο είναι να επιβιώσουν τα πιο ικανά, γρήγορα κτλ), την αναπαραγωγή (τα ζώα αναπαράγονται μεταφέροντας τα γονίδιά τους στα παιδιά τους) και τη μετάλλαξη (κατά το πέρασμα των αιώνων όλα τα είδη των ζώων υφίστανται μεταλλάξεις που διαφοροποιούν ορισμένα χαρακτηριστικά τους όχι πάντοτε προς το καλύτερο).
Όμοια με το βιολογικό ανάλογο, οι Γενετικοί Αλγόριθμοι βασίζονται στην ύπαρξη ενός πληθυσμού πρακτόρων, οι οποίοι αποτελούν και λύσεις του προβλήματος.
Ο πληθυσμός αυτός των πρακτόρων μεταβάλλεται από γενιά σε γενιά (δηλαδή σε κάθε επανάληψη του αλγορίθμου) καθώς υφίσταται: (α) διαλογή (sorting) ανάλογα με την καταλληλότητά του, (β) διασταύρωση (crossover) έτσι ώστε να μεταφέρει τα χαρακτηριστικά του στην επόμενη γενιά και (γ) μετάλλαξη (mutation) έτσι ώστε να καλύπτεται όλος ο χώρος των λύσεων. Να αναφέρουμε, δε, ότι με το μηχανισμό της διαλογής πραγματοποιείται εκμετάλλευση (exploitation) του χώρου των λύσεων ενώ με τους μηχανισμούς της διασταύρωσης και της μετάλλαξης εξερεύνηση (exploration).

Η βασική δομή ενός κανονικού (standard) γενετικού αλγορίθμου περιγράφεται από τα ακόλουθα πέντε βήματα:

1. *άρχισε με έναν τυχαία παραγόμενο πληθυσμό n πρακτόρων (bitstrings) που ο κάθε ένας αναπαριστάνει μια λύση.Στη συνέχεια θα ανάφερόμαστε στους πράκτορες ως άτομα.*
2. *υπολόγισε την καταλληλότητα για κάθε άτομο x (fitness(x)) στον πληθυσμό*
3. *επανέλαβε [n/2] φορές, κάθε φορά παράγοντας δύο άτομα:*

(a) επέλεξε ένα ζευγάρι γονέων (τοποθέτησέ τους πίσω μετά από την επιλογή), η πιθανότητα της επιλογής αυξάνει ανάλογα με την ικανότητα του ατόμου

(b) με την πιθανότητα διασταύρωσης να είναι p_c, εφάρμοσε διασταύρωση για να δημιουργηθούν δύο απόγονοι, διαφορετικά κλονοποίησε απλά τους γονείς (πάρε ένα ίδιο αντίγραφο από κάθε γονέα)

(c) μετάλλαξε και τους δύο απογόνους σε κάθε γονίδιο με πιθανότητα μετάλλαξης p_m εάν το n είναι περιττός, απόρριψε ένα μέλος της νέας γενιάς τυχαία

4. *αντικατέστησε τον τρέχοντα πληθυσμό με νέο*
5. *εάν το καλύτερο άτομο δεν είναι αρκετά κατάλληλο, πήγαινε στο βήμα 2*

Αυτός ο βασικός γενετικός αλγόριθμος που παρουσιάζεται παραπάνω μοντελοποιεί την εξελικτική θεωρία που συναντάται στη φύση. Μόλις δημιουργηθεί η νέα γενιά αντικαθιστά την παλαιά γενιά (βήμα 4) που παύει έπειτα να υπάρχει. Αυτό είναι επιθυμητό στη φύση, όπου ο στόχος είναι να κυριαρχήσουν οι υγιείς πληθυσμοί.

Μια περίληψη ενός ψευδοκώδικα του απλού γενετικού αλγορίθμου (SGA) που περιγράφεται από τον Goldberg παρουσιάζεται στην Εικόνα 2.2.Ο πληθυσμός στο χρόνο t αναπαριστάνεται από τη χρονικά εξαρτημένη μεταβλητή P, με τον αρχικό πληθυσμό των τυχαίων εκτιμήσεων να είναι P(0).

```
procedure GA
begin
        t = 0;
        αρχικοποίησε τον P(t);
        εκτίμησε τον P(t);
        while not finished do
        begin
                t = t + 1;
                επέλεξε τον P(t) από τον P(t-1);
                Δημιούργησε ζευγάρια στον P(t);
                εκτίμησε τον P(t);
        end
end.
```

Εικόνα 2.2:Ψευδοκώδικας ενός απλού γενετικού αλγορίθμου (SGA).

2.2.1.2 Η αυτοοργάνωση ως βιολογική βάση της θεωρίας της νοημοσύνης του σμήνους

Η αυτοοργάνωση (self-organization) είναι ένα σύνολο δυναμικών μηχανισμών μέσω του οποίου εμφανίζονται σε ένα σύστημα δομές υψηλού επιπέδου οι οποίες με τη σειρά τους έχουν προκύψει από αλληλεπιδράσεις των συστατικών του, από στοιχεία δηλαδή χαμηλού επιπέδου (Bonabeau, Dorigo, Theraulaz,1999).Οι κανόνες οι οποίοι

προσδιορίζουν τις αλληλεπιδράσεις αυτές αφορούν αποκλειστικά την τοπική πληροφόρηση που μπορούν να έχουν τα συστατικά στοιχεία, ενώ αγνοούν τη συνολική κατάσταση του συστήματος. Αυτή προκύπτει σαν άθροισμα μεμονωμένων συμπεριφορών.Τέσσερα είναι τα βασικά χαρακτηριστικά της αυτοοργάνωσης:

1. ***Θετική ανάδραση.*** *Πρόκειται για υποδείξεις συμπεριφοράς οι οποίες προωθούν τη δημιουργία επιθυμητών δομών.Τέτοια παραδείγματα θετικής ανάδρασης αποτελούν η στρατολόγηση (recruitment) και η ενίσχυση (reinforcement).Έτσι, η στρατολόγηση εντόμων προς συλλογή μιας συγκεκριμένης τροφής αποτελεί περίπτωση θετικής ανάδρασης και επιτυγχάνεται π.χ. με την έκκριση φερομόνης στα μυρμήγκια ή με το «χορό» στις μέλισσες.*
2. ***Αρνητική ανάδραση.*** *Η αρνητική ανάδραση αντισταθμίζει τη θετική και λειτουργεί ως σταθεροποιητής στο σχηματισμό της δομής. Τέτοιου είδους ανάδραση μπορεί να είναι ο κορεσμός, η εξάντληση ή ακόμη και ο ανταγωνισμός.Για παράδειγμα στη συλλογή τροφής, η διαδικασία ανακόπτεται από τον περιορισμένο αριθμό των διαθέσιμων εντόμων, την εξάντληση της τροφής, το συνωστισμό στη συγκεκριμένη πηγή ή τον ανταγωνισμό μεταξύ δύο ισοδύναμων πηγών τροφής.*
3. ***Τυχαιότητα.*** *Η τυχαιότητα είναι κρίσιμη για την ευστάθεια του συστήματος.Εξαιτίας της μπορούν να ανακαλυφθούν νέες βελτιωμένες λύσεις και αποφεύγεται ο αποκλεισμός των δομών σε «τοπικά» βέλτιστα.*
4. ***Πολλαπλές αλληλεπιδράσεις.****Για να είναι δυνατή η βελτιστοποίηση χρειάζεται μία ελάχιστη πυκνότητα οντοτήτων.Τα όντα αυτά πρέπει να αφήνουν το στίγμα τους προς τα υπόλοιπα ενώ ταυτόχρονα να εκμεταλλεύονται τα αποτελέσματα από τις αναζητήσεις των υπολοίπων*

Συνήθως, η αυτοοργάνωση προσδίδει στα φαινόμενα που λαμβάνει χώρα τις παρακάτω χαρακτηριστικές ιδιότητες:

1. Τη δημιουργία δομών σε ένα αρχικά ομογενές μέσο. *Για παράδειγμα η αρχιτεκτονική μιας φωλιάς, η χωροθέτηση των κυψελών σε ένα μελίσσι.*
2. Η πιθανή συνύπαρξη πολλαπλών καταστάσεων ευστάθειας. *Αφού οι δομές που προκύπτουν, προκύπτουν λόγω τυχαιότητας, το σύστημα μπορεί να συγκλίνει σε μία από τις πολλές πιθανές ευσταθείς καταστάσεις, ανάλογα με τις αρχικές συνθήκες.*
3. Η ύπαρξη διακλαδώσεων συμπεριφοράς κάτω από ορισμένες παραμέτρους. *Η συμπεριφορά των εντόμων είναι δυνατόν να αποκλίνει από τη φυσιολογική (ή καλύτερα την αναμενόμενη) κάτω από ειδικές περιπτώσεις. Ενδεικτικά αναφέρουμε κάποιο είδος τερμιτών οι οποίοι όταν χτίζουν τη φωλιά τους εναποθέτουν τους σβώλους στα σημεία που υπάρχει μεγάλη έκκριση φερομόνης μέχρι ενός σημείου και από εκεί και έπειτα στα σημεία όπου η φερομόνη πέφτει ανάμεσα σε κάποια ελαττωμένα όρια.*

2.2.1.2 Στιγμεργία – Stigmergy

Ως στιγμεργία μεταφράζουμε τον αγγλικό όρο «Stigmergy» ο οποίος προέκυψε με τη σειρά του από την ένωση δύο ελληνικών λέξεων: «στίγμα» + «έργο». Ο όρος εισήχθη από τον Grassé το 1959 και αποτελεί περισσότερο μία θεωρητική σύλληψη παρά μία βιολογική λειτουργία. Η στιγμεργία εκφράζει τη γενική ιδέα της χρήσης της έμμεσης επικοινωνίας (μέσω στίγματος) για την οργάνωση μιας ομάδας. Αποτελεί το έτερο θεωρητικό θεμέλιο (μαζί με την αυτοοργάνωση) της θεωρίας της νοημοσύνης του σμήνους.
Σύμφωνα με τη στιγμεργία, ο συντονισμός των εργασιών σε μία ομάδα δεν προκύπτει

από άμεση επικοινωνία μεταξύ των εργατών αλλά ως αποτέλεσμα των εργασιών τους. Δεν κατευθύνει ο εργάτης την εργασία του αλλά η εργασία τον εργάτη. Ο Grassé παρατηρώντας ένα είδος τερμιτών έδειξε πως λειτουργεί η προσέγγιση αυτή στο χτίσιμο της φωλιάς τους.Κάθε μυρμήγκι εναποθέτει σβώλους οι οποίοι επιφέρουν μία μεταβολή στο περιβάλλον.Τα μυρμήγκια που ακολουθούν απαντούν σε αυτήν την εναλλαγή διαθέτοντας κάποιο εσωτερικό κίνητρο, το οποίο ο Grassé ισχυρίζεται πως είναι γενετικά καθορισμένο .
Είναι σημαντικό να σημειώσουμε πως η στιγμεργία δεν εξηγεί τους μηχανισμούς τους οποίους οδηγούν τα όντα να συμπεριφερθούν με αυτόν τον τρόπο. Πρόκειται όμως για τη θεωρητική μοντελοποίηση του γενικού πλαισίου της συμπεριφοράς αυτής: Η ατομική συμπεριφορά μεταβάλλει το περιβάλλον,το οποίο με τη σειρά του μεταβάλλει τη συμπεριφορά των ατόμων.
Κάτι τέτοιο συμβαίνει και στον αλγορίθμο που περιγράφουμε στην επόμενη ενότητα: τα μυρμήγκια εκκρίνουν φερομόνη μεταβάλλοντας το περιβάλλον ώστε τα υπόλοιπα μέλη της κοινωνίας να συμπεριφέρνονται ανάλογα με αυτές τις μεταβολές.
Το σημαντικό που προκύπτει από την προσέγγιση αυτή είναι πως μπορούμε να σχεδιάσουμε αυτόνομες οντότητες οι οποίες θα μπορούν να επιλύουν προβλήματα χωρίς να επικοινωνούν άμεσα, αρκεί αυτές να μπορούν να αντιληφθούν και να μεταβάλλουν το περιβάλλον τους. Σύμφωνα με την στιγμεργία το περιβάλλον λειτουργεί ως εξωτερική μνήμη του συστήματος, ενώ το έργο μπορεί να συνεχιστεί από οποιοδήποτε άτομο. Τέλος, η στιγμεργία προσδίδει ιδιαίτερη ευελιξία σε μία κοινότητα. Αν υπάρξει μία διαταραχή στο περιβάλλον, τότε η κοινότητα μπορεί να ανταποκριθεί συλλογικά σε αυτή τη διαταραχή.Το γεγονός αυτό είναι πολύ σημαντικό, αν φανταστούμε ότι μας προσφέρει μία βάση να σχεδιάσουμε τεχνητές αυτόνομες οντότητες οι οποίες θα μπορούν να ανταποκριθούν αυτόματα σε μία διαταραχή, δηλαδή χωρίς να είναι απαραίτητο να προγραμματιστούν για τη διαταραχή αυτή.

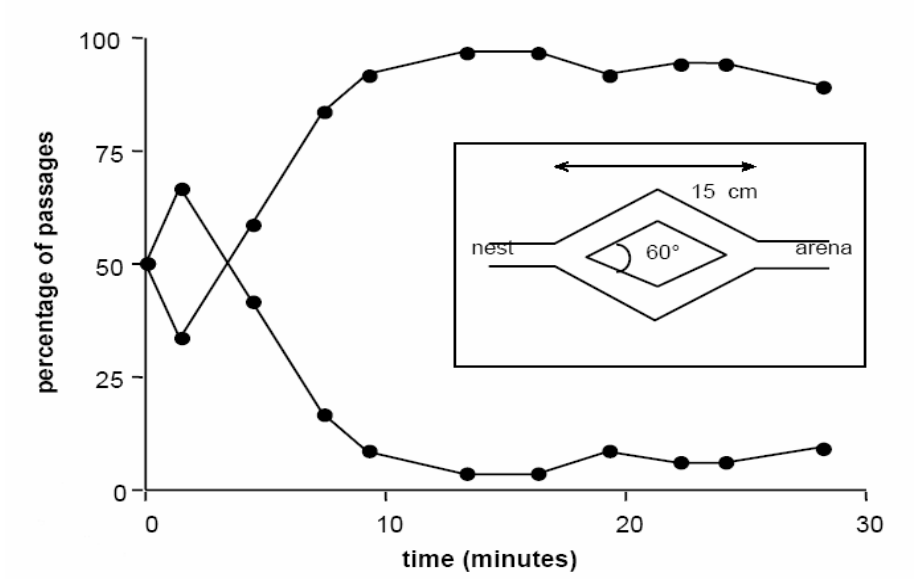
2.2.2 Αλγόριθμοι εμπνευσμένοι από αποικίες μυρμηγκιών

Οι βασικοί αλγόριθμοι της θεωρίας εμπνέονται από τη συλλογική αναζήτηση τροφής των μυρμηγκιών. Οι αποικίες των μυρμηγκιών αυτοοργανώνονται για να βρουν και να εκμεταλλευτούν τις τροφές που είναι οι πιο κοντινές στη φωλιά τους. Προκειμένου να επιλυθούν προβλήματα βελτιστοποίησης, σχεδιάζονται τεχνητές οντότητες (πράκτορες) οι οποίες μιμούνται τη συμπεριφορά των αληθινών μυρμηγκιών.

Η βελτιστοποίηση στην αναζήτηση τροφής από τα μυρμήγκια

Τα περισσότερα είδη μυρμηγκιών υιοθετούν τη συμπεριφορά «Αφήνω ίχνη – ακολουθώ ίχνη» (trail-laying trail-following) κατά την αναζήτηση τροφής (Hölldpbler, B., & E. O. Wilson,1990).. Το κάθε μυρμήγκι εκκρίνει μία χημική ουσία καθώς κινείται από μία πηγή τροφής προς τη φωλιά του.Η ουσία αυτή καλείται φερομόνη.Τα μυρμήγκια που αναζητούν τροφή ακολουθούν αυτά τα ίχνη φερομόνης, ενώ η διαδικασία σύμφωνα με την οποία τα μυρμήγκια επηρεάζονται από τα ίχνη αυτά για τις επιλογές τους, ονομάζεται στρατολόγηση (recruitment). Όταν οι επιλογές τους κατευθύνονται αποκλειστικά από τα ίχνη αυτά η διαδικασία ονομάζεται μαζική στρατολόγηση (mass recruitment). Ο Deneubourg και οι συνεργάτες του (Deneubourg, J. L., S. Aron, S. Goss, and J. M. Pasteels,1990), παρουσίασαν με ένα πείραμα πως η επιλογή διαδρομής για την αναζήτηση τροφής στα μυρμήγκια βασίζεται στην αυτοοργάνωση. Το πείραμα αυτό θεωρείται πλέον κλασσικό και είναι το πείραμα της δυαδικής γέφυρας.
Μία πηγή τροφής διαχωρίζεται από τη φωλιά των μυρμηγκιών μέσω μιας γέφυρας η οποία διαθέτει δύο κλάδους ίσου μήκους, τον Α και τον Β (Εικόνα 2.3).Αρχικά δεν υπάρχει καθόλου φερομόνη σε κανέναν από τους κλάδους, οπότε τα μυρμήγκια

μπορούν να επιλέξουν και τους δύο με την ίδια πιθανότητα. Ωστόσο, λόγω τυχαιότητας κάποιος κλάδος θα επιλεγεί αρχικά από έναν μεγαλύτερο αριθμό μυρμηγκιών. Καθώς όμως τα μυρμήγκια εκκρίνουν φερομόνη στο πέρασμα τους, αυτό θα παρακινήσει ολοένα και περισσότερα μυρμήγκια να επιλέγουν το συγκεκριμένο κλάδο εις βάρος του άλλου.



Εικόνα 2.3: Το πείραμα της δυαδικής γέφυρας με ίσους κλάδους(Deneubourg et. al , 1990).

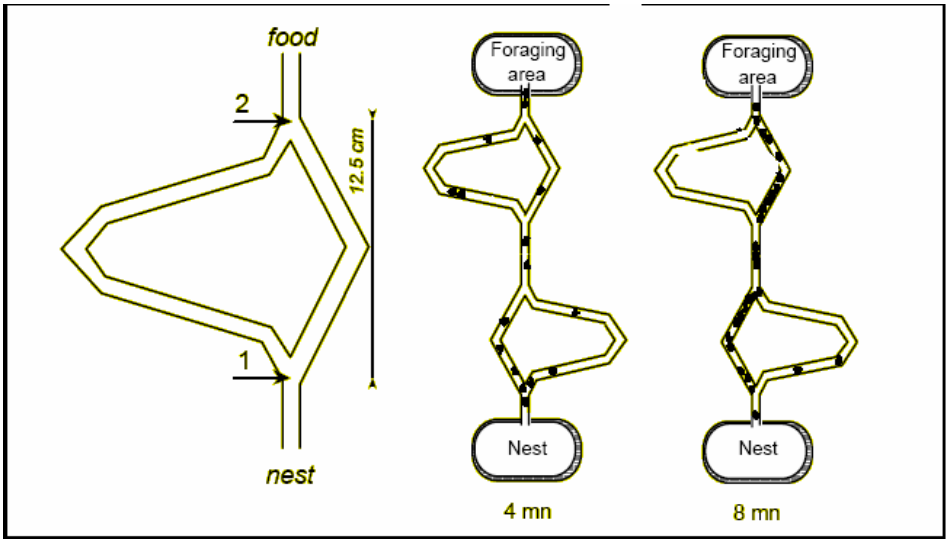
Ο Deneubourg και οι συνεργάτες του αναπτύξανε ένα μοντέλο για το φαινόμενο αυτό, ώστε η συμπεριφορά του μοντέλου να είναι πολύ κοντά στις πειραματικές παρατηρήσεις με τα φυσικά όντα.Ας υποθέσουμε πως το ποσό της φερομόνης που υπάρχει σε έναν κλάδο είναι ανάλογο με τον αριθμό των μυρμηγκιών που επέλεξαν αυτόν τον κλάδο για να διασχίσουν τη γέφυρα.Ας κάνουμε επιπλέον την παραδοχή πως η φερομόνη δεν εξατμίζεται καθόλου στη διάρκεια του πειράματος.Στο μοντέλο λοιπόν που αναπτύξανε οι ερευνητές η πιθανότητα επιλογής ενός κλάδου σε μία συγκεκριμένη στιγμή εξαρτάται από το συνολικό αριθμό των μυρμηγκιών που επιλέξανε αυτή τη διαδρομή μέχρι εκείνη τη στιγμή.Πιο συγκεκριμένα, έστω Ai και Bi το πλήθος των μυρμηγκιών που έχουν επιλέξει τους κλάδους A και B αντίστοιχα όταν i μυρμήγκια έχουν διασχίσει τη γέφυρα. Η πιθανότητα P_A το (i+1)-οστό μυρμήγκι να επιλέξει τον κλάδο A είναι:

$$P_A = \frac{\qquad\qquad}{\qquad\qquad} = 1 - P_B \qquad (2.3)$$

Η παραπάνω εξίσωση δείχνει ποσοτικά πως η μεγαλύτερη συγκέντρωση μυρμηγκιών στον κλάδο A παράγει μεγαλύτερη πιθανότητα επιλογής για τον κλάδο, ανάλογα με τις τιμές A_i και B_i. Η παράμετρος n καθορίζει το βαθμό μη γραμμικότητας της συνάρτησης επιλογής. Αν είναι μεγάλη, τότε ακόμη και μία μικρή υπεροχή στην ποσότητα φερομόνης στον ένα κλάδο μπορεί να παράγει μεγάλη πιθανότητα επιλογής. Η παράμετρος k δίνει το μέτρο ελκυστικότητας για έναν κλάδο ο οποίος δεν έχει ακόμα επιλεγεί. Όσο μεγαλύτερη είναι τόσο περισσότερο τυχαία γίνεται η επιλογή.

Η ίδια ομάδα ερευνητών, με επικεφαλής τον Goss αυτή τη φορά πραγματοποίησε μία παραλλαγή του πειράματος αυτού αλλάζοντας το μήκος των κλάδων, ώστε ο ένας

να είναι σημαντικά μακρύτερος από τον άλλον(Εικόνα 2.4).



Εικόνα 2.4: Το πείραμα της δυαδικής γέφυρας με άνισους κλάδους(Goss et. al ,1989).

Το προηγούμενο μοντέλο εφαρμόζει και σε αυτήν την περίπτωση. Επειδή τα μυρμήγκια που επιλέγουν τη σύντομη διαδρομή επιστρέφουν πιο γρήγορα στη φωλιά, είναι φυσικό η φερομόνη να αυξάνεται γρηγορότερα στον κοντύτερο κλάδο. Έτσι τα μυρμήγκια επιλέγουν ολοένα και περισσότερο το σύντομο κλάδο. Μάλιστα τα πειράματα έδειξαν πως η πιθανότητα επιλογής του σύντομου κλάδου αυξάνεται όσο μεγαλώνει ο λόγος μήκος μακρύτερου κλάδου/μήκος σύντομου κλάδου.Βέβαια οι ερευνητές πειραματίστηκαν αφήνοντας στα μυρμήγκια το μακρύ κλάδο ως μόνη επιλογή για 30 λεπτά παρουσιάζοντας το σύντομο στη συνέχεια.Αυτό που παρατηρήσανε είναι πως τα μυρμήγκια ακολουθούνε τις εκκρίσεις τις φερομόνης και εγκλωβίζονται στη λάθος επιλογή.Αυτό συμβαίνει γιατί η εξάτμιση της φερομόνης πραγματοποιείται με έναν ρυθμό που της επιτρέπει να παραμένει για αρκετό καιρό (από μερικές ώρες μέχρι μερικούς μήνες).Στο πεδίο βέβαια της βελτιστοποίησης, ο ρυθμός εξάτμισης έχει επιταχυνθεί προς αποφυγήν του εγκλωβισμού σε τοπικά βέλτιστες λύσεις (Bonabeau,1996). Μπορούμε να εξηγήσουμε γιατί η φύση δεν έχει καταλήξει σε μία παρόμοια λύση, αν αναλογιστούμε πως η βέλτιστη λύση για τα έντομα δεν εξαρτάται μόνο από το μήκος της διαδρομής αλλά και από άλλους παράγοντες όπως η προστασία των μυρμηγκιών και της φωλιάς τους.

- ***Ant Colony Optimization(ACO)***

Αλγόριθμοι βελτιστοποίησης και ελέγχου που εμπνέονται από τα πρότυπα της εύρεσης τροφής στα μυρμήγκια ήταν απροσδόκητα επιτυχείς και έχουν γίνει γνωστοί τα τελευταία χρόνια ως Βελτιστοποίηση Αποικιών Μυρμηγκιών (Ant Colony Optimisation - ACO) και Δρομολόγηση Αποικιών Μυρμηγκιών (Ant Colony Routing - ACR). Το πρώτο σύστημα ACO εισήχθη από τον Marco Dorigo στη διδακτορική του διατριβή και ονομάστηκε σύστημα μυρμηγκιών (Ant System - AS) και από τότε έχει χρησιμοποιηθεί για την λύση πολλών προβλημάτων συνδυαστικής βελτιστοποίησης. Η βασική ιδέα των αλγορίθμων που έχουν προταθεί έκτοτε παραμένει ίδια και στηρίζεται στη λειτουργία της «θετικής ανάδρασης» που αναφέραμε στη προηγούμενη ενότητα.Οι αλγόριθμοι αυτοί θα αποτελέσουν και το αντικείμενο μελέτης της συγκεκριμένης διπλωματικής και τούς αναλύουμε λεπτομερώς παρακάτω.

Η συμπεριφορά «αφήνω-ίχνη ακολουθώ-ίχνη» πρέπει να ενισχυθεί με τέτοιο τρόπο ώστε να προωθούνται οι καλές λύσεις.Μία εικονική φερομόνη χρησιμοποιείται ως μέσο ενίσχυσης και επιτρέπει να αποθηκεύονται οι καλές λύσεις στη μνήμη του αλγορίθμου, έτσι ώστε η αναζήτηση να προχωρά προς ακόμη καλύτερες λύσεις. Βέβαια, οι λύσεις που πρέπει να ενισχύονται είναι στην πραγματικότητα οι «πολύ καλές» και όχι οι απλά «καλές».Για το λόγο αυτό χρησιμοποιείται και η λειτουργία της «αρνητικής ανάδρασης» με τη μορφή της εξάτμισης της φερομόνης. Η εξάτμιση επιτυγχάνεται προσθέτοντας και τη διάσταση του χρόνου στον αλγόριθμο.Βέβαια η εξάτμιση οφείλει να είναι τόσο αργή όσο να επιτρέπεται στα τεχνητά μυρμήγκια να αναπτύξουν τη συμπεριφορά τους και τόσο γρήγορη ώστε να αποφεύγονται εγκλωβισμοί σε τοπικά βέλτιστα και πρώιμη σύγκλιση του αλγορίθμου.Το κλειδί της επιτυχίας του αλγορίθμου είναι το ίδιο με αυτό της επιτυχίας της συμπεριφοράς των αληθινών μυρμηγκιών, δηλαδή η συνεργατική συμπεριφορά, ώστε το κάθε μυρμήγκι να εκμεταλλεύεται τα αποτελέσματα της αναζήτησης των συμβιωτών του.

Συνοψίζοντας ο ACO υλοποιείται ως μια ομάδα ευφυών πρακτόρων οι οποίοι προσομειώνουν την συμπεριφορά των μυρμηγκιών μετακινούμενοι στον χώρο αναζήτησης για την εύρεση μιας λύσης χρησιμοποιώντας μηχανισμούς συνεργασίας και προσαρμογής. Κατα την φάση συνεργασίας κάθε πληθυσμός λύσεων εξετάζεται ούτως ώστε να ανανεώσει μια ολική μνήμη,την φερομόνη .Τότε η ατομική φάση προσαρμογής του πράκτορα περιλαμβάνει μια κατασκευαστική μέθοδο η οποία χρησιμοποιεί πληροφορία που βρίσκεται στην ολική μνήμη για την κατασκευή νέων λύσεων.

Δοθέντος λοιπόν ενός Π.Σ.Β το πρώτο βήμα εφαρμογής σε αυτό ενός ACO αλγορίθμου ,είναι ο ορισμός ενός επαρκούς μοντέλου για την αναπαράσταση του προβλήματος.Το μοντέλο αυτό αποτελεί τη βάση για τον καθορισμό του βασικού συστατικού ενός ACO, του μοντέλου φερομόνης(Dorigo & Socha,2006):

Χρησιμοποιώντας το μοντέλο ενός Π.Σ.Β όπως το ορίσαμε στην Ενότητα 2.1 εξάγουμε το μοντέλο φερομόνης ενός ACO αλγόριθμου.Αρχικά καλούμε ως συστατικό μιας λύσης το στιγμιότυπο μιας μεταβλητής απάφασης $X_i = u_i^j$ (πχ. ,μια μεταβλητή X_i που της έχει ανατεθεί μια τιμή u_i^j από την περιοχή $\mathbf{D_i}$) και το συμβολίζουμε ως c_{ij}.Το σύνολο όλων των πιθανών συστατικών μιας λύσης συμβολίζεται ως $\mathbf{C}$.
Ορίζουμε τότε μια συσχέτιση ανάμεσα σε μια παράμετρο ενός μονοπατιού φερομόνης T_{ij} και ένα συστατικό c_{ij} και συμβολίζουμε το σύνολο όλων αυτών των παραμέτρων ως $\mathbf{T}$. Η τιμή μιας παράμετρου φερομόνης T_{ij} , 'τ_{ij}' καλείται τιμή φερομόνης ,χρησιμοποιείται και ανανεώνεται από τον ACO αλγόριθμο κατα την διάρκεια της αναζήτησης στον $\mathbf{S}$. Επιπλέον μας επιτρέπει να μοντελοποιήσουμε την στοχαστική κατανομή των διαφόρων συστατικών μιας λύσης.

Στην ACO τεχνητοί πράκτορες(μυρμήγκια) κατασκευάζουν την λύση ενός Π.Σ.Β μετακινούμενα στον λεγόμενο κατασκευαστικό γράφο(construction graph), $G_C(\mathbf{V},\mathbf{E})$. Κάθε πράκτορας στο χρόνο t επιλέγει τον επόμενο κόμβο που θα επισκεφτεί στο χρόνο $t+1$. Γι' αυτό, εάν γίνει μία επανάληψη του αλγορίθμου ACO, όπου m κινήσεις γίνονται από m πράκτορες στο διάστημα $(t, t+1)$, τότε σε κάθε n επαναλήψεις του αλγορίθμου (που αποκαλείται ένας κύκλος) κάθε πράκτορας θα έχει ολοκληρώσει μια διαδρομή (tour).

Ο γράφος είναι πλήρως συνδεδεμένος και περιλαμβάνει ένα σύνολο κόμβων $\mathbf{V}$ και ένα σύνολο ακμών $\mathbf{E}$.Το σύνολο των συστατικών του $\mathbf{C}$ μπορεί να συσχετίζονται είτε με το συνόλο των ακμών ή το σύνολο των κόμβων του γράφου, ενώ οι πράκτορες μετακινούνται από κόμβο σε κόμβο κατα μήκος των ακμών του γραφήματος,

κατασκευάζοντας επαυξητικά μια μερική λύση (partial solution).Επιπλέον εναποθέτουν ορισμένη ποσότητα φερομόνης στα συστατικά της λύσης, δηλαδή είτε στους κόμβους ή στις ακμές των οποίων διέρχονται.Η ποσότητα της φερομόνης $\Delta\tau$ που εναποτίθεται κατα τη διάρκεια της εκτέλεσης εξαρτάται άμμεσα από την ποιότητα της παραγόμενης λύσης –παράλληλα διαδοχικά μυρμήγκια αντλούν πληροφορία από τη συσσωρευμένη φερομόνη προκειμένου να μετακινηθούν σε περισσότερο υποσχόμενες περιοχές του χώρου αναζήτησης.

Ο ψευδο-κώδικας για την βασική μεθευρετική ACO διαδικασία(Αλγόριθμος 2.1) παρουσιάζεται και αναλύεται παρακάτω:

Αρχικά όλα τα στοιχεία που θα χρησιμοποιηθούν από τον αλγόριθμο αρχικοποιούνται και κατόπιν ακολουθεί ένας βρόγχος που περιλαμβάνει τρία αλγοριθμικά συστατικά.Σε μια λοιπόν επανάληψη του βρόγχου λαμβάνουν χώρα η κατασκευή των λύσεων από τους πράκτορες, η (προαιρετική) βελτίωσή τους με χρήση ενός αλγόριθμου τοπικής αναζήτησης και η ανανέωση των φερομόνων στα διάφορα μονοπάτια του γράφου.Πιο συγκεκριμένα κάθε αλγοριθμικό συστατικό έχει ως εξής:

Αλγόριθμος 2.1:Η βασική ACO διαδικασία για ένα Π.Β.Σ(Dorigo & Socha,2006).

Algorithm 1 Ant colony optimization metaheuristic

```
Set parameters, initialize pheromone trails

while termination conditions not met do

  ConstructAntSolutions

  ApplyLocalSearch        {optional}

  UpdatePheromones

end while
```

ConstructAntSolutions: Κατά τη διάρκεια της διαδικασίας ένα σύνολο m τεχνητών πρακτόρων κατασκευάζει λύσεις από τα στοιχεία ενός πεπεπερασμένου συνόλου των διαθέσιμων συστατικών μιας λύσης $\mathbf{C} = \{ c_{ij} \}, i = 1,\ldots,n, j=1,\ldots,|\mathbf{D_i}|$. *Η κατασκευή της λύσης αρχίζει από μια κενή μερική λύση* $s^p = \emptyset$ *και σε κάθε βήμα της η τρέχουσα μερική λύση* s^p *εκτείνεται ,προσθέτωντας κάποιο εφικτό συστατικό από το σύνολο των εφικτών γειτόνων* $\mathbf{N}(s^p) \subseteq \mathbf{C}$. *Η διαδικασία της δημιουργίας μιας λύσης ουσιαστικά αντιπροσωπεύει ένα μονοπάτι στον γράφο* $G_C(\mathbf{V},\mathbf{E})$ *,ενώ τα επιτρεπόμενα μονοπάτια στον* G_C *καθορίζονται από συγκεκριμένο μηχανισμό κατασκευής λύσεων ,ο οποίος ορίζει το σύνολο* $\mathbf{N}(s^p)$ *,λαμβάνοντας πάντα υπόψη τη μερική λύση* s^p.
Η επιλογή του συστατικού μιας λύσης από το $\mathbf{N}(s^p)$ *γίνεται στοχαστικά σε κάθε βήμα της κατασκευής - οι ακριβείς κανόνες για την επιλογή αυτή ποικίλουν ανάμεσα στις διάφορες παραλλαγές του ACO. Ο πιό γνωστός κανόνας χρησιμοποιείται στον Ant System (AS) αλγόριθμο και είναι ο εξής:*

$$\frac{\qquad\qquad}{\qquad\qquad}, \qquad (2.3)$$

,όπου τ_{ij} *είναι η τιμή της φερομόνης που σχετίζεται με το συστατικό* c_{ij} *και* $n(\cdot)$ *μια*

συνάρτηση η οποία αναθέτει σε κάθε βήμα μια ευρετική τιμή σε όλα τα εφικτά συστατικά $c_{ij} \in \mathbf{N}(s^p)$ *.Η τιμή αυτή συχνά αναφέρεται ως ευρετική πληροφορία ή ορατότητα . Επίσης τα* α *και* β *είναι θετικές παράμετροι οι οποίες καθορίζουν τη σχετική βαρύτητα της φερομόνης έναντι της ευρετικής πληροφορίας.*

ApplyLocalSearch:Μετά την ολοκλήρωση της κατασκευής των λύσεων και πρίν την ανανέωση των φερομόνων συχνά λαμβάνουν χώρα κάποιες προαιρετικές διαδικασίες(daemon actions) ,οι οποίες εξυπηρετούν στην περαίωση συγκεκριμένων ενεργειών που δεν μπορούν να αναλάβουν οι πράκτορες και άπτονται κυρίως της φύσης του προβλήματος.Έχουν συγκεντρωτικό/αποσυγκεντρωτικό χαρακτήρα,ενώ οι πιο συχνά χρησιμοποιούμενες περιλαμβάνουν την εφαρμογή μεθόδων τοπικής αναζήτησης στις λύσεις που έχουν προκύψει από το προηγούμενο βήμα.Κατόπιν οι τοπικά βελτιστοποιημένες λύσεις χρησιμοποιούνται για την ανανέωση των φερομόνων.

UpdatePheromones: Ο σκοπός αυτής της διαδικασίας είναι να ενισχύσει τις τιμές φερομόνης που σχετίζονται με καλές λύσεις και να μειώσει εκείνες που σχετίζονται με πιό 'φτωχές'.Αυτό επιτυγχάνεται i) ελλατώνοντας όλες τις τιμές φερομόνης μέσω της εξάτμισης φερομόνης και ii) αυξάνοντας τα επίπεδα φερομόνης που αντιστοιχούν σε ένα σύνολο καλών λύσεων $\mathbf{S}_{upd}$ *:*

$$\tau_{ij} \leftarrow (1-\rho) \cdot \tau_{ij} + \rho \cdot \quad , \qquad (2.4)$$

,όπου $\mathbf{S}_{upd}$ *είναι το σύνολο των λύσεων που χρησιμοποιούνται για την ανανέωση,* $\rho \in (0,1]$ *μια παράμετρος που καλείται συχνότητα εξάτμισης ,και* $F : \mathbf{S} \rightarrow \mathrm{R}_0^+$ *είναι μια συνάρτηση για την οποία* $f(s) < f(s')$ $F(s) \geq F(s'), \forall s \neq s' \in \mathbf{S}$. *Η* $F(\cdot)$ *είναι γνωστή ως συνάρτηση καταλληλότητας (fitness function) ή αντικειμενική συνάρτηση.*

Η εξάτμιση της φερομόνης είναι απαραίτητη για αν αποφύγουμε μια πρόωρη σύγκλιση της μεθόδου, καθώς αγνοεί ως ένα βαθμό την υπάρχουσα πληροφορία ,ενισχύοντας έτσι την εξερεύνηση νέων περιοχών του χώρου αναζήτησης.Φυσικά ανάλογα με τις διάφορες παραλλαγές των ACO αλγόριθμων(ACS,MAX-MIN ,Ant System,(MMAS)) ,διαφέρει και ο τρόπος με τον οποίο ανανεώνεται η φερομόνη. Διαφορετικά στιγμιότυπα του κανόνα ανανέωσης της Εξισ. 2.4 προκύπτουν ανάλογα με τον ορισμό του συνόλου $\mathbf{S}_{upd}$ *,που σε αρκετές περιπτώσεις είναι υποσύνολο του* $\mathbf{S}_{iter}$ $\{s_{bs}\}$*, όπου το* $\mathbf{S}_{iter}$ *είναι το σύνολο m λύσεων που παράχθηκαν στη τρέχουσα επανάληψη(κύκλο) του αλγόριθμου από τους m πράκτορες και* s_{bs} *(best-so-far) η καλύτερη λύση που έχει βρεθεί από την εκκίνηση του αλγόριθμου. Ένα ευρέως γνωστό παράδειγμα είναι ο κανόνας ανανέωσης του Ant System, AS- update rule ,όπου:*

$$\mathbf{S}_{upd} \leftarrow \mathbf{S}_{iter} \qquad (2.5)$$

Ένας πιό συχνά χρησιμοποιούμενος κανόνας ανανέωσης είναι ο IB-update rule(Iteration-Best):

$$\mathbf{S}_{upd} \leftarrow arg\ max_{s \in \mathrm{Siter}} \qquad (2.6)$$

Ο IB- update rule εισάγει μια ενισχυμένη πόλωση κοντά στις καλές λύσεις, και παρότι αυξάνεται η ταχύτητα εύρεσης καλών λύσεων, αυξάνεται παράλληλα και η πιθανότητας πρόωρης σύγκλισης σε τοπικό βέλτιστο.Μια ακόμα πιο δυνατή πόλωση

εισάγει ο BS-update rule , όπου BS παραπέμπει στη χρήση της καλύτερης εώς τώρα λύσης s_{bs} (best-so-far).Σε αυτή την περίπτωση $\mathbf{S}_{upd} \leftarrow \{ s_{bs} \}$. Συμπερασματικά μια ACO διαδικασία που κάνει χρήση των IB- update και BS-update κανόνων και περιλαμβάνει ένα μηχανισμό για την αποφυγή πρόωρης σύγκλισης ,επιτυγχάνει στη πράξη καλύτερα αποτελέσματα από τον AS- update κανόνα.

2.2.3 Ενσωμάτωση συστημάτων πρακτόρων και μεθόδων εμπνευσμένων από την φύση.

Σε αυτή την ενότητα θα συζητησουμε τις πτυχές των πρακτορικών αλγόριθμων βασιζόμενων σε μεθόδους εμπνευσμένες απο την φύση,των φυσικών μεθόδων βασιζόμενων σε συστήματα πρακτόρων και στην ακολουθιακή χρήση πολυπρακτορικών συστημάτων και φυσικών μεθόδων.Υπάρχουν τριών τύπων αλγόριθμοι ή αλγοριθμικά πλαίσια(framework) βάση των οποίων μπορούμε να συσχετίσουμε τα πολυπρακτορικών συστημάτων(MAS) με φυσικές μεθόδους(Ruhul,Sarker & Ray,2010):

- *Ο πράκτορας είναι υπέυθυνος για ενέργειες και συμπεριφορές,*
- *Ο πράκτορας αντιπροσωπεύει μια υποψήφια λύση και*
- *Τα MAS ,οι φυσικές μέθοδοι και οι παραλλαγές τους χρησιμοποιούνται ακολουθιακά.*

Στον πρώτο τύπο ο πράκτορας αντιπροσωπεύει τους τύπους των ενεργειών και συμπεριφορών που οδηγουν το σύστημα στη επίλυση του προβλήματος.Σε αυτή την περίπτωση ο πράκτορας μπορεί να χρησιμοποιήσει φυσικές μεθόδους για την εκμάθηση ή βελτίωση συγκεκριμένων λειτουργιών.Ο Vacher et al(2004) πρότεινε ένα πολυπρακτορικό σύστημα όπου ο πράκτορας συντίθεται από συναρτήσεις ενεργειών ,γνώσης και συμπεριφορών.Στο συστημά τους γίνεται χρήση ενός γενετικού αλγόριθμου για τον καθορισμό ενός συνόλου συναρτήσεων για κάθε πράκτορα που οδηγεί το σύστημα στην επαρκή επίλυση του προβήματος.Οι Nunes & Oliveira(2002) και οι Iantovics & Enachescu(2006) χρησιμοποίησαν φυσικές μεθόδους ως αλγόριθμους εκμάθησης μέσα στο MAS.Οι Milano & Roli(2008) πρότειναν μια πολυπρακτορική μεθευρετική αρχιτεκτονική όπου ο πράκτορας ως ένα σύστημα είναι ικανός να κατασκευάσει μια λύση ,να κινηθεί σε ένα τοπίο(landscape),να επικοινωνήσει με άλλους πράκτορες ,να είναι ενεργός και πιθανώς προσαρμόσιμος.Κατα αυτόν τον τρόπο μπορεί να σχεδιαστεί ένας νέος αλγόριθμος (ο οποίος συνδυάζει διάφορους μεθευρετικούς) επιλέγωντας τους εμπλεκόμενους πράκτορες και ορίζοντας τις αλληλεπιδράσεις τους.

Στον δεύτερο τύπο ο πράκτορας αντιπροσωπεύει μια υποψήφια λύση .Σε μια φυσική μέθοδο το κάθε άτομο ορίζεται ώς ένας πράκτορας υπό συγκεκριμένες προυποθέσεις.Αρα ο πληθυσμός των ατόμων μπορεί να θεωρηθεί ως ένας πληθυσμός πρακτόρων(Liu et al ,2010). Όπως όρισε ο Barkat Ullah et al(2009) ένας πράκτορας περιέχει εκτός της υποψήφιας λύσης και συγκεκριμένες πληροφορίες,όπως είναι οι τεχνικές και ο ρυθμός εκμάθησης του. Οι πράκτορες κατοικούν σε 2-D κυψελωτά/δικτυωτά περιβάλλοντα(Zhang et al,2009) ,σε δομές γράφων(Drezewski et al,2008) ,αλυσιδωτά ή δακτύλια περιβάλλοντα.Ο κάθε πράκτορας στοχεύει στην αύξηση της ενεργειάς του μέσω της συνεργασίας ή του ανταγωνισμού του με τους γείτονές του.Στην περίπτωση για παράδειγμα των εξελικτικών διαδικασιών , πχ τους γενετικούς αλγόριθμους ,μπορούν να εφαρμόστούν σε ένα πράκτορα γενετικές διαδικασίες όπως ανταγωνισμός, συνεργασία ,μετάλλαξη και αυτο-εκμάθηση.Ο ανταγωνισμός χρησιμοποιεί ένα είδος τουρνουά για αναγνωρίσει έναν πράκτορα σαν νικητή ή χαμένο και στη περίπτωση

που χάνει αντικαθίσταται από το σύστημα με ένα νεό πράκτορα.Η συνεργασία αφορά κυρίως τον μηχανισμό ανταλλαγής πληροφοριών μέσω του τελεστή διασταύρωσης ,ενώ η αυτοεκμάθηση θα μπορούσε να βασιστεί σε τεχνικές τοπικής αναζήτησης.Επίσης σε ορισμένα δικτυωτά(lattice) περιβάλλοντα ο πράκτορας είναι τοποθετημένος σε συγκεκριμένο σημείο(lattice point) και αλληλεπιδρά μόνο με τους γειτονές του (Liu et al ,2010).Ο αριθμός των γειτόνων ποικίλει(Barkat Ullah et al,2009) ενώ σε άλλες περιπτώσεις οι πράκτορες κινούνται και συναντάνε άλλους πράκτορες σε ολόκληρο τον χώρο του περιβάλλοντος.

Τέλος στον τρίτο τύπο τα MAS και οι φυσικές μεθόδοι μπορούν να εφαρμοστούν είτε ακολουθιακά είτε επαναληπτικά για την επίλυση συγκεκριμένων κλάσεων προβλημάτων.Για την λύση ενός δυναμικού job-shop προβλήματος χρονοπρογραμματισμού οι Li & Du(2009) πρότειναν την αρχική κατανομή των εργασιών μέσω ενός πολυπρακτορικού συστήματος και εφάρμοσαν στη συνέχεια έναν υβριδικό γενετικό αλγόριθμο για το βέλτιστο επαναχρονοπρογραμματισμό τους.Οι Liu et al(2005) πρότειναν ένα πολυπρακτορικό σύστημα όπου ο γενετικός αλγόριθμος χρησιμοποιείται ως μια συνεχής γεννήτρια εναλλακτικών και όχι ως βελτιστοποιητής.Μια τέτοια γεννήτρια διεγείρει την φαντασία των πρακτόρων και επεκτείνει τους ορίζοντες σκέψης τους προς όφελος της βέλτιστης λήψης αποφάσεων.Τέλος στο σύστημα που προτάθηκε από τον Guo(2006) διαφορετικά είδη πρακτόρων έχουν διαφορετικές λειτουργίες και κάθε είδος κάνει χρήση ενός αλληλεπιδραστικού γενετικού αλγόριθμου ώς τμήμα της ολικής διαδικασίας λήψης αποφάσεων.

2.2.4 Agent-based Nature inspired algorithms

Σε σύγκριση με τα MAS ,τα άτομα σε μια φυσική μέθοδο δεν έχουν την απαραίτητη αυτονομία που είναι πρωταρχικής σημασίας στα πολυπρακτορικά συστήματα.Ωστόσο κληρονομούν κάποιες ιδιότητες των MAS όπως ο πληθυσμός των ατόμων,αυτοοργάνωση,προσαρμοστικότητα(adaptation) και αυτοεκμάθηση (self-learning).Τα άτομα εργάζονται συλλογικά(όχι ατομικά) για να επιτύγχουν ένα κοινό αλλά γενικό(global) στόχο.Στις περισσότερες των περιπτώσεων το περιβάλλον δίνεται και εκφράζεται από πολύπλοκες συναρτήσεις και συνθήκες,ενώ επίσης υπάρχουν σε κάποια μορφή και σχέσεις ανταγωνιστικότητας μεταξύ τους(πχ. η επιλογή του πληθυσμού των ατόμων για διασταύρωση μεσω των διαδικασιών κατάταξης(ranking) και επιλογής), Ruhul,Sarker & Ray(2010).Παίρνοντας ως παράδειγμα την γενετική διαδικασία κατα την διεργασία της διασταύρωσης τα επιλεγμένα άτομα συνεργάζονται και ανταλλάσουν πληροφορία μεταξύ τους(γενετικό υλικό).Αυτή η ανταλλαγή πληροφορίας μπορεί να θεωρηθεί ως μια μορφή επικοινωνίας μεταξύ των ατόμων-τα άτομα μπορούν να επικοινωνήσουν περαιτέρω για να ελέχξουν σχετικές τοποθεσίες στο τοπίο καταλληλότητας.Η διαδικασία κατάταξης μπορεί να θεωρηθεί ως ένας τύπος έμμεσης επικοινωνίας βασιζόμενη στη πληροφορία.Οι memetic αλγόριθμοι τέλος κάνουν χρήση αλγορίθμων τοπικής αναζήτησης η οποία μπορεί να θεωρηθεί ως τοπική εκμάθηση.

Όπως συζητήθηκε στην προηγούμενη ενότητα τα συστήματα τύπου 1 αποτελούν ουσιαστικά πολυπρακτορικά συστήματα και τα τύπου 3 βασίζονται ταυτόχρονα στα σχήματα πολυπρακτορικών συστημάτων και φυσικών διαδικασιών.Ωστόσο τα συστήματα τύπου 2 ενσωματώνουν κάποια στοιχεία των πρακτόρων και πολυπρακτορικών συστημάτων υπο το αλγοριθμικό πλαίσιο των φυσικών διαδικασιών.Ως απόδειξη παρατηρείται αύξηση στην διεθνή βιβλιογραφία χρήσης τέτοιων σύστημάτων σε διαφορετικά προβλήματα, όπως προβλήματα ικανοποίησης περιορισμών(Zhong et al,2005) ,συνδιαστικής(Liu et al,2010) και αριθμητικής βελτιστοποίησης(Barkat et al,2009).

Όπως προείπαμε στα τύπου 2 συστήματα τα άτομα μιας φυσικής διαδικασίας ορίζονται ως πράκτορες με ή χωρίς περαιτέρω χαρακτηριστικά και υποθέσεις.Σαν παράδειγμα περαιτέρω υποθέσεων ο Barkat Ullah et al. δήλωσε ότι :"Τα άτομα του πληθυσμού μιας φυσικής διαδικασίας δεν αποτελούν πράκτορες,ανταυτού βασιζόμενος στις πεποιθήσεις του ατόμου και την εμπειρία εκμάθησής τους ,κάθε πράκτορας αντιπροσωπεύει(ή υποστηρίζει) ένα άτομο του πληθυσμού''.Μια τέτοια υπόθεση θα ενίσχυε την ενσωμάτωση ορισμένων χαρακτηριστικών στην αναπαράσταση του πράκτορα/ατόμου εντός της φυσικής μεθόδου.Επιπλέον τα άτομα δύναται σε αυτή την περίπτωση να εφαρμόσουν ιδιότητες των πρακτόρων στην φυσική διαδικασία όπως επικοινωνία,συνεργασία ,ανταγωνισμό και εκμάθηση για την βελτίωση της απόδοσης της.Ο Davidsson et al.(2009) πρότεινε την συγχώνευση των δυό προσεγγίσεων σε μια υβριδική καθώς αλληλοσυμπληρώνουν η μια την άλλη.Ο De Jon υπέδειξε επίσης ότι η εξέλιξη και επίτευξη των στόχων συμβαίνει σε επίπεδο πληθυσμού ,ενώ η ''δια βίου μάθηση'' συμβαίνει σε επίπεδο ατόμου.Οι πράκτορες μαθαίνουν κατα την διάρκεια της ζωής τους βελτιώνοντας την ποιότητα τους (τιμή καταλληλότητας).Αυτή η διαδικασία εκμάθησης μπορεί να επιλεγεί από τον κάθε πράκτορα αυτόνομα.Για παράδειγμα σε ένα πρόβλημα βελτιστοποίησης τεχνικές τοπικής αναζήτησης θα μπορούσαν να αποτελούν διαδικασίες εκμάθησης για ένα πράκτορα.Πρός αυτή την κατεύθυνση ο Barkat Ullah et al(2007) πρότεινε πολλαπλούς αλγόριθμους τοπικής αναζήτησης από τους οποίους ο πράκτορας θα επιλέξει μια μέθοδο εκμάθησης βασιζόμενος στην εμπειρία εκμάθησής του.Σημειώνεται ότι ο συνδυασμός της τοπικής οπτικής των πρακτόρων και της ικανότητας ολικής εξερεύνησης μιας φυσικής μεθόδου εξισσοροπεί της λειτουργίες εκμμετάλευσης(exploitation) της συσωρευμένης γνώσης και της εξερεύνησης(exploration) νέων λύσεων.

Σε κάθε κινησή του ο πράκτορας εφαρμόζει ποιοτική ή ποσοτική κρίση ή λογισμό,βασιζόμενος στην δική του πεποίθηση,κοινωνική αληλεπίδραση ,γνώση και νοημοσύνη.Επιπλεόν η πληροφορία που έχει κερδίσει στο παρελθόν μπορεί να αρχειοθετηθεί και να ανακτηθεί συστηματικά όποτε αυτό είναι αναγκαίο ,κάτι που θα έκανε τις βασιζόμενες σε πράκτορες φυσικές μεθόδους πιο ελκυστικές σε πρακτικά συνδυαστικά προβλήματα.

2.3 Χρονοπρογραμματισμός Παραγωγής

Σκοπός της παρούσας ενότητας είναι μια σύντομη ανασκόπηση των βασικών συστημάτων χρονοπρογραμματισμόυ παραγωγής και η παρουσίαση των κυριοτέρων κριτηρίων που χρησιμοποιούνται στη βιομηχανία.Ολοκληρώνουμε με μια εκτενή αναφορά στη εφαρμογή των συστημάτων πρακτόρων με μεθόδους εμπνευσμένες από την φύση σε προβλήματα χρονοπρογραμματισμού παραγωγής,όπως το Job Shop Scheduling Problem(JSP).

2.3.1 Χρονοπρογραμματισμός στο σύστημα παραγωγής

Τα προβλήματα χρονοπρογραμματισμού παραγωγής έχουν εκλάβει τόσο μεγάλη ποσότητα προσοχής για το λόγο ότι συνδυάζουν επιστημονικές περιοχές τόσο διαφορετικές όπως είναι αυτές του σχεδιασμού παραγωγής (production planning), του computer design και της παραγωγής πλάνων (timetabling) και επιπρόσθετα έχουν τη δυνατότητα δραματικά να μειώσουν το κόστος και να αυξήσουν την παραγωγή και επακολούθως τα κέρδη. Ένα σημαντικό πεδίο της έρευνας, τόσο από την πλευρά της επιχειρησιακής έρευνας όσο και από την πλευρά της τεχνητής νοημοσύνης, έχει αφιερωθεί στην θεωρία και τις εφαρμογές του χρονοπρογραμματισμού παραγωγής.

Τα Συστήματα παραγωγής κατά παραγγελία (job-shop), 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. Εκτός όμως από αυτό το ευρέως διαδεδομένο σύστημα παραγωγής, υπάρχουν και άλλα συστήματα όπως είναι αυτό της συνεχούς ροής και της παραγωγής παρτίδων.Τα συστήματα παραγωγής τα οποία χρησιμοποιούνται για το χρονοπρογραμματισμό διαφόρων εργασιών μέσα σε μια βιομηχανία είναι δυνατόν να ταξινομηθούν σε κατηγορίες με διαφορετικό κριτήριο ταξινόμησης. Το σημαντικότερο κριτήριο είναι η πολυπλοκότητα επεξεργασίας στη δομή παραγωγής. Ένας συνηθισμένος διαχωρισμός σύμφωνα με το παραπάνω κριτήριο είναι(Pinedo,2005):

- **Συστήματα μοναδικής μηχανή (single machine):** *Οι εργασίες εκτελούνται σε μια μηχανή. Αυτή η περίπτωση είναι και η απλούστερη όλων των δυνατών βιομηχανικών προβλημάτων και αποτελεί ειδική περίπτωση πιο πολύπλοκων περιπτώσεων.*
- **Συστήματα με παράλληλες μηχανές (parallel machine):***Κάθε εργασία μπορεί να εκτελεστεί εναλλακτικά σε πολλές μηχανές ίδιες ή και διαφορετικές.*
- **Συστήματα κατασκευής έργων**:*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*
- **Συστήματα συνεχούς ροής (flow-shop):***Κάθε εργασία χωρίζεται σε διεργασίες καθεμία από τις οποίες εκτελείται σε ορισμένη μηχανή. Η σειρά εκτέλεσης των διεργασιών είναι προκαθορισμένη και ίδια για όλες τις εργασίες.*
- **Συστήματα παραγωγής(Open-shop)**:*Κάθε εργασία χωρίζεται σε διεργασίες καθεμία από τις οποίες εκτελείται σε ορισμένη μηχανή. Η σειρά εκτέλεσης των διεργασιών είναι διαφορετική για κάθε εργασία και μια εργασία μπορεί να έχει και παραπάνω από έναν εναλλακτικούς δρόμους για τη σειρά των διεργασιών κάθε εργασίας.*

- **Συστήματα παραγωγής σε παρτίδες (batch-shop)**:*Τα συστήματα αυτά έχουν χαρακτηριστικά συστημάτων τόσο συνεχούς ροής όσο και παραγωγής κατά παραγγελία. Συναντιώνται συχνά στην πράξη και αφορούν παραγωγή αποθηκεύσιμων προϊόντων, που παράγονται με κοινό παραγωγικό εξοπλισμό.Για τη μείωση των αποθεμάτων, η ετήσια ποσότητα που πρέπει να παραχθεί από κάθε προϊόν διαιρείται σε παρτίδες, οι οποίες διαδέχονται η μία την άλλη, σε έναν κύκλο που επαναλαμβάνεται αρκετές φορές στη διάρκεια του έτους.*

2.3.2 Αντικειμενικά κριτήρια

Ο χρονοπρογραμματισμός παραγωγής επιδιώκει να καθορίσει χρόνους έναρξης για τις διάφορες εργασίες χωρίς να παραβιάζει τους διάφορους περιορισμούς που διέπουν το πρόβλημα ,όπως για παράδειγμα οι χρόνοι επεξεργασίας διαφορετικών εργασιών στην ίδια μηχανή να μην επικαλύπτονται χρονικά.Το παραγόμενο διάγραμμα χρόνου (Gantt Chart) ονομάζεται χρονοπρόγραμμα(Siang Lyn,2005).Επίσης στον χρονοπρογραμματισμό επιδιώκουμε την βελτιστοποίηση τουλάχιστον ενός αντικειμενικού κριτηρίου.Τυπικά κριτήρια ή αντικειμενικές συναρτήσεις σε προβλήματα χρονοπρογραμματισμού παραγωγής είναι τα ακόλουθα(Pinedo,2005):

1. *Χρόνος ολοκλήρωσης(**completion time**):ο χρόνος στον οποίο μια συγκεκριμένη εργασία ολοκληρώνει την επεξεργασία της.*
2. *Χρόνος ροής(**flow time**):το χρονικό διάστημα κατα το οποίο μια συγκεκριμένη εργασία βρίσκεται στο σύστημα κατασκευής.Ο χρόνος ροής είναι το χρονικό διάστημα μεταξύ μιας άφιξης μιας εργασίας στο σύστημα και της αναχώρησης από αυτό.*
3. *Καθυστέρηση(**lateness**):ο χρόνος(θετικός ή αρνητικός) κατα τον οποίο ο χρόνος ολοκλήρωσης διαφέρει απο την ημερομηνία παράδοσης(due date).Η καθυστέρηση συνηθως 'ανταμείβει' τις εργασίες που γίνονται νωρίτερα και 'τιμωρεί' αυτές που αργοπορούν.*
4. *Χρόνος βραδύτερης περάτωσης(**tardiness**):η καθυστέρηση μιας εργασίας εαν αυτή είναι θετική.Αν η καθυστέρηση δεν είναι θετική,ο χρόνος βραδύτερης περάτωσης είναι μηδέν.Ο χρόνος βραδύτερης περάτωσης έχει να κάνει με το γεγονός ότι σε πολλές περιπτώσεις συγκεκριμένα κόστη μπορεί να προέρχονται από μια θετική καθυστέρηση ,ενώ μια αρνητική καθυστέρηση δεν αποφέρει πραγματικά οφέλη.Αυτό είναι ένα καλό μέτρο της χρησιμότητας(utility) ενος προγράμματος σε πραγματικούς όρους.*
5. *Χρόνος νωρίτερης περάτωσης (**earliness**):το αντίστροφο της καθυστέρησης μιας εργασίας,εαν αυτή είναι αρνητική.Εάν η καθυστέρηση είναι θετική ο χρόνος νωρίτερης περάτωσης είναι μηδέν.Ο χρόνος νωρίτερης περάτωσης αντιστοιχεί σε κάποιες περιπτώσεις όπου μπορεί να υπάρξει κόστος όταν μια δραστηριότητα ολοκληρώνεται πρίν την ημέρα παράδοσης(το κόστος αποθεμάτων ενδέχεται να είναι αρκετά υψηλό).*

Συνολικός χρόνος ολοκλήρωσης(makespan,Cmax,):*Με τον όρο makespan καλούμε την διάρκεια στην οποία όλες οι λειτουργίες όλων εργασιών έχουν ολοκληρωθεί.Το πρόγραμμα με το μικρότερο makespan λαμβάνεται συνήθως ως το πρόγραμμα με τη μεγαλύτερη χρησιμότητα.Η βασική ιδέα πίσω απο αυτό είναι πως ολοκληρώνοντας το δοσμένο σύνολο δραστηριοτήτων νωρίτερα θα επιτρέψει νέες δραστηριότητες να ξεκινήσουν νωρίτερα.Αυτή η ιδέα είναι μια πολύ καλή προσέγγιση στην ειδική περίπτωση όπου διαθέτουμε μια μηχανή και αποτελεί μια χρήσιμη σκέψη για πιο γενικά προβλήματα.Το makespan δεν λαμβάνει υπόψη τις ημερομηνίες παράδοσης ,κάτι που στη βιομηχανία σε πραγματικές συνθήκες είναι παράγοντας*

υψηλής προτεραιότητας.

Σταθμισμένη καθυστέρηση(weighted lateness,Lwt):*Η σταθμισμένη καθυστέρηση είναι ένα μέτρο της διαφοράς μεταξύ της αυστηρής ημερομηνίας παράδοσης και της πραγματικής ημερομηνίας ολοκλήρωσης των εργασιών. Στην περίπτωση της καθυστερημένης παράδοσης η τιμή της είναι θετική και στην περίπτωση της παράδοσης σε νωρίτερη χρονική στιγμή από αυτή της ημερομηνίας παράδοσης το μέτρο αυτό έχει τιμή αρνητική. Στη συνέχεια η τιμή αυτή πολλαπλασιάζεται με ένα συντελεστή βάρους. Είναι ένα δημοφιλές μέτρο της ποιότητας του χρονοπρογραμματισμού. Είναι πολύ ισχυρό μέτρο με την έννοια ότι οι βέλτιστες λύσεις που βρίσκουμε στηριζόμενοι σε αυτό συχνά παράγουν καλές λύσεις για προβλήματα με άλλες αντικειμενικές συναρτήσεις.*

Σταθμισμένος χρόνος βραδύτερης περάτωσης(weighted tardiness,Twt): *Σε πολλές περιπτώσεις ο χρόνος βραδύτερης περάτωσης αποτελεί μία καλή αντικειμενική συνάρτηση. Αυτό το μέτρο είναι το ίδιο με την σταθμισμένη καθυστέρηση εκτός από το ότι στις νωρίτερα ολοκληρωμένες εργασίες θέτει την τιμή μηδέν. Με αυτό τον τρόπο ανταποκρίνεται καλύτερα στις ανάγκες του πραγματικού πελάτη. Σε πολλές πρακτικές καταστάσεις είναι το πιο κατάλληλο μέτρο της απόδοσης ενός χρονοπρογράμματος μιας και λαμβάνει υπόψη του το κόστος που συνεπάγεται μία καθυστερημένη παράδοση αλλά δεν αντισταθμίζει με κέρδος τις νωρίτερες παραδόσεις εργασιών από τη στιγμή που αυτό δεν είναι αναγκαίο. Παρά το ότι είναι ένα μέτρο αρκετά χρήσιμο, στη βιβλιογραφία θα βρούμε λίγες περιπτώσεις που να χρησιμοποιείται ως μέτρο απόδοσης.*

Μέγιστος χρόνος βραδύτερης περάτωσης(maximum tardiness,Tmax): *Η ελαχιστοποίηση του μέγιστου χρόνου βραδύτερης περάτωσης είναι σημαντικό μέτρο όταν οι πελάτες ανέχονται τις μικρές καθυστερήσεις αλλά γίνονται σταδιακά όλο και πιο ανήσυχοι και πιεστικοί όσο αυξάνονται αυτές. Η ελαχιστοποίηση της μέγιστης καθυστέρησης είναι αντίστοιχο πρόβλημα και μπορεί να χρησιμοποιηθεί ως βοηθητικό για τη λύση άλλων προβλημάτων.*
Παρακάτω, το C_i δηλώνει το χρόνο ολοκλήρωσης της εργασίας J_i. Τα περισσότερα κλασικά κριτήρια χρονοπρογραμματισμού λαμβάνουν υπόψη τους μία ημερομηνία παράδοσης δ_i για κάθε εργασία. Η βραδύτερη περάτωση T_i της εργασίας J_i ορίζεται ως $T_i = \max(0, C_i - \delta_i)$. Η μεταβλητή U_i χρησιμοποιείται για να δηλώσει μία μονάδα «ποινής» (penalty) για κάθε καθυστερημένη εργασία. Η U_i ισούται με 0 όταν $C_i \leq \delta_i$ και με 1 σε άλλη περίπτωση.
Το κριτήριο χρονοπρογραμματισμού διατυπώνεται είτε ως άθροισμα είτε ως μέγιστο. Ένας συντελεστής βάρους w_i μπορεί να δοθεί σε κάθε εργασία δίνοντας ιδιαίτερη βαρύτητα σε κάποιες από αυτές. Σημειώνουμε τα παρακάτω γνωστά κριτήρια βελτιστοποίησης:

1) *Makespan : $F = C_{max} = \max C_i$*
2) *Σταθμισμένος συνολικός χρόνος ροής (total weighted flow time) : $F = \sum w_i C_i$*
3) *Μέγιστος χρόνος βραδύτερης περάτωσης (maximum tardiness) : $F = T_{max} = \max T_i$*
4) *Σταθμισμένος συνολικός χρόνος βραδύτερης περάτωσης (total weighted tardiness) : $F = \sum w_i T_i$*
5) *Σταθμισμένος συνολικός αριθμός αργοπορημένων εργασιών (total weighted number of late jobs) : $F = \sum w_i U_i$*

*Σε αυτή την διπλωματική επιλέξαμε το **Job Shop Problem**(JSP) ως αντιπροσωπευτικό πρόβλημα του πεδίου του χρονοπρογραμματισμού ,καθώς όχι μόνο ανήκει στην κατηγορία των NP-hard προβλημάτων συνδυαστικής βελτιστοποίησης (Garey et al., 1976) αλλά αποτελεί και ένα από τα πλέον δύσκολα διαχειρίσιμα προβλήματα λόγω των πολλών περιορισμών που το διέπουν(Nakano and Yamada, 1991; Lawler et al., 1993).Αυτό γίνεται σαφές από το γεγονός οτι αλγόριθμοι μπορούν να λύσουν βέλτιστα άλλα NP-hard προβλήματα,όπως το γνωστό Πρόβλημα του Πλανώδιου Πωλητή(TSP) με περισσότερες από 4000 πόλεις ,ωστόσο δεν έχουν σχεδιαστεί ακόμα στρατηγικές που να εγγυούνται βέλτιστες λύσεις για στιγμιότυπα του JSP μεγαλύτερα των 20 εργασίες(n) x 10 μηχανές(m) .Ένα JSP μεγέθους n x m έχει ως άνω φράγμα ((n!) ^{m} λύσεις ,άρα ένα 20x 10 πρόβλημα μπορεί να έχει το πολύ 7.2651 x 10^{183} πιθανές λύσεις.Είναι ευνόητο ότι μια ολοκληρωμένη απαρίθμιση όλων των πιθανοτήτων για την εξαγωγή των εφικτών χρονοπρογραμμάτων και της βέλτιστης λύσης δεν είναι πρακτική,γεγόνος που ευνοεί την χρήση πολυπρακτορικών μεθόδων εμπνευσμένων από την φύση ,οι οποίες αποτελούν ουσιαστικά προσεγγιστικές μεθόδους, ως εργαλείο επίλυσης του JSP για την εξαγωγή ποιοτικών λύσεων σε λογικά χρονικά διαστήματα.*

2.4 Προηγούμενη Κατάσταση

Σκοπός μας σε αυτή τη διπλωματική είναι η δημιουργία συστήματος πρακτόρων με χρήση μεθόδων εμπνευσμένων από την φύση και συγκεκριμένα από αποικίες μυρμηγκιών για την επίλυση μιας υποκατηγορίας συστημάτων παραγωγής αυτή του JSP.Θεωρώντας δυό κριτήρια θα επιχειρήσουμε στα επόμενα κεφάλαια την επίλυση του μονοκριτηριακού JSP για κάθε κριτήριο ξεχωριστά και στη συνέχεια θα επεκτείνουμε το σύστημά μας για την πολυκριτηριακή περίπτωση στοχέυοντας στην ταυτόχρονη ελαχιστοποίηση των δυό κριτηρίων και την εξαγωγή ενός βέλτιστου Pareto συνόλου λύσεων.Προς αυτή την κατεύθυνση παρουσιάζουμε την δουλειά που έχει γίνει τόσο στη μονοκριτηριακή όσο και στην πολυκριτηριακή περίπτωση των συστημάτων χρονοπρογραμματισμού παραγωγής ,όπου πρακτορικές μεθόδοι εμπνευσμένες από την φύση σχεδιάζονται για την επίλυση τους.

2.4.1 Μονοκριτηριακός χρονοπρογραμματισμός παραγωγής

Η πλειοψηφία των περιπτώσεων στον μονοκριτηριακό χρονοπρογραμματισμό παραγωγής θεωρεί ως αντικειμενικό κριτήριο το makespan.Η βιβλιογραφία είναι πλούσια προς αυτήν την κατεύθυνση με χρήση μεθόδων εμπνευσμένων από την

θεωρία σμήνους και την εξελικτική διαδικασία:

Όπως έχουμε αναφέρει οι Γενετικοί Αλγόριθμοι (Genetic Algorithms, GAs) βασίζονται σε ένα αφηρημένο πρότυπο της φυσικής εξέλιξης, έτσι ώστε η ποιότητα των πρακτόρων να χτίζεται στο πιο υψηλό επίπεδο που είναι συμβατό με το περιβάλλον (περιορισμοί του προβλήματος) (Holland 1975, Goldberg 1989). Η ανάλυση που παρουσιάζεται εδώ είναι βασισμένη στην ταξινόμηση που δόθηκε από τον Cheng το 1996, ο οποίος σύνταξε σε ένα ευρετήριο όλα τα σχέδια απεικόνισης που ίσχυσαν τα τελευταία χρόνια και τα κατέταξε σε 9 κατηγορίες:

1) Στηριζόμενα στην διαδικασία *2) Στηριζόμενα στις εργασίες*

3) Στηριζόμενα στην δυαδικότητα των εργασιών *4) Στηριζόμενα στους χρόνους ολοκλήρωσης*

5) Στηριζόμενα σε τυχαία ταξινόμηση *6) Στηριζόμενα στην λίστα προτιμήσεων*

7) Στηριζόμενα στους κανόνες προτεραιοτήτων *8) Στηριζόμενα στη διασπαστική γραφική αναπαράσταση* *9) Στηριζόμενα στις μηχανές*

Αυτά τα σχέδια απεικόνισης μπορούν να ομαδοποιηθούν σε δύο βασικές προσεγγίσεις κωδικοποίησης, στις άμεσες και στις έμμεσες. Η άμεση προσέγγιση κωδικοποιεί ένα πρόγραμμα παραγωγής ως χρωμόσωμα και οι γενετικοί χειριστές χρησιμοποιούνται για να εξελίξουν αυτά τα χρωμοσώματα σε καλύτερα προγράμματα. Οι κατηγορίες 1 έως 5 είναι παραδείγματα αυτής της στρατηγικής. Στην έμμεση προσέγγιση, μια ακολουθία προτιμήσεων απόφασης κωδικοποιείται στο χρωμόσωμα, παραδείγματος χάριν οι κανόνες αποστολής για τις αναθέσεις εργασίας, και οι γενετικοί χειριστές εφαρμόζονται για να βελτιώσουν τη διάταξη των διάφορων προτιμήσεων. Ένα πρόγραμμα παραγωγής δημιουργείται έτσι έπειτα από την ακολουθία προτιμήσεων. Οι κατηγορίες 6 έως 9 είναι παραδείγματα αυτής της μεθόδου.

Η πιο πρόωρη μέθοδος GA που εφαρμόστηκε στο JSP ήταν αυτή του Davis (1985) και αποτελεί παράδειγμα έμμεσης προσέγγισης. Η τεχνική του κατασκευάζει μια προτιμημένη διάταξη των διεργασιών για κάθε μηχανή. Η προσέγγιση επεκτάθηκε περισσότερο από τους Falkenauer και Bouffouix (1991) οι οποίοι κωδικοποίησαν τις διεργασίες που υποβάλλονται σε επεξεργασία σε μια μηχανή σαν κατάλογο προτίμησης, με μια σειρά συμβόλων. Ο Della Croce (1995) υιοθέτησε και αυτός αυτή τη μέθοδο κωδικοποίησης αλλά με ένα πιο διορατικό βλέμμα και στόχο την παραγωγή ενεργών προγραμμάτων.

Μια από τις πιο πρόσφατες αναφορές βασισμένες στην ταξινόμηση με βάση τις λίστες προτιμήσεων είναι αυτή του Kobayashi (1995) όπου ένα χρωμόσωμα είναι μια συμβολοσειρά μήκους n και κάθε στοιχείο της αντιστοιχεί στην διεργασία που πρόκειται να επεξεργαστεί σε μια μηχανή.Μια από τις πιο πρόωρες άμεσες προσεγγίσεις είναι και αυτή των Nakano και Yamada (1991) οι οποίοι δημιούργησαν μια δυαδική κωδικοποίηση βασισμένη στη σχέση προτεραιότητας των διεργασιών στην ίδια μηχανή. Υιοθετείται επίσης μια στρατηγική αποκαλούμενη καταναγκασμός, που τροποποιεί ένα χρωμόσωμα εάν μια λειτουργία μπορεί να μετατοπιστεί αριστερά. Οι Yamada και Nakano (1992) επιπλέον ήταν σε θέση να βελτιώσουν αυτήν την εργασία με τον καθορισμό ενός χειριστή διασταυρώσεων αποκαλούμενου GA/GT που βασίζεται στον αλγόριθμο των Giffler και Thompson (1960). Εδώ το χρωμόσωμα είναι ένας διαταγμένος κατάλογος χρόνων ολοκλήρωσης των διαδικασιών. Τέλος, πιο

πρόσφατα, ο Shi (1997) εφάρμοσε μια τεχνική διασταυρώσεων που διαιρεί τυχαία έναν αυθαίρετα επιλεγμένο σύντροφο σε δύο υποσύνολα, από τα οποία ο παράγεται ο απόγονος.

Θα πρέπει να σημειωθεί ωστόσο ότι η πλειοψηφία των προσεγγίσεων των GA εμφανίζεται να δίνει φτωχά αποτελέσματα λόγω των δυσκολιών που υπάρχουν με τους χειριστές και το πρόγραμμα διασταυρώσεως της κωδικοποίησης.

Άλλες μεθόδοι όπως η PSO έχουν εφαρμοστεί στο JSP . Οι YAN Ping et al ,2011 παρουσιάσαν μια standar μέθοδο PSO και μιας βελτιωμένης IPSO(Improved PSO) διαδικασίας .Το IMPSO σύστημα χρησιμοποιεί νέες μεθόδους κωδικοποίησης και αποδικωποίησης των λύσεων , ενώ έχει ενσωματωθεί και μια ευρετική μέθοδος τοπικής αναζήτησης ώστε να βελτιώθεί περαιτέρω η διαδικασία αναζήτησης του ολικού βέλτιστου.

Οι Xiaoyu Song et al ,2009 πρότειναν μια standar PSO και μια υβριδική PSO διαδικασία(Hybridised PSP,HPSO) για το JSP.Ο HPSO αλγόριθμος αναθέτει την εξερέυνηση του χώρου αναζήτησης στον πράκτορα με την βέλτιστη στρατηγική διατήρησης και χρησιμοποιεί τον αλγόριθμο Taboo Search για να ενισχύσει την τοπική αναζήτηση .

Οι Zhong et al(2005) σχεδίασαν ένα πολυπρακτορικό σύστημα βασισμένο στην εξελικτική διαδικασία για το JSP.Ο κάθε πράκτορας κατοικεί σε περιβάλλον πλέγματος όπου διατηρεί συγκεκριμένη ενέργεια ίση με τη τιμή του makespan.Σε κάθε γενιά του αλγόριθμου ο κάθε πράκτορας επιδεικνύει ανταγωνιστική συμπεριφορά διαγράφοντας τους πράκτορες με χαμηλή ενέργεια στην γειτονιά του και επιπλεόν εκτελεί μια λειτουργία αυτοεκμάθησης εφαρμόζοντας τεχνικές τοπικής αναζήτησης στη λύση(χρονοπρόγραμμα) που αντιπροσωπεύει.

Πολυπρακτορικές μέθοδοι βασισμένες σε αποικίες μυρμηγκιών έχουν εφαρμοστεί στο JSP.Οι Colorni et al(1993) ήταν οι πρώτοι που εφάρμοσαν ένα πολυπρακτορικό σύστημα εμπνευσμένο από αποικίες μυρμηγκιών.Η διαδικασία χρησιμοποιεί έναν πληθυσμό πρακτόρων οι οποίοι κατοικούν σε περιβάλλον γράφου και μοιράζονται μια κοινή μνήμη με την μορφή μιας δομής φερομόνης στην οποία έχουν ταυτόχρονη πρόσβαση.Οι πράκτορες επικοινωνούν έμμεσα μέσω αυτής της δομής(στιγμεργία) για την κατασκευή χρονοπρογραμματών και η συνολική ικανότητα βελτιστοποίησης του συστήματος οφείλεται στην ασύχγρονη ανανέωση της μνήμης από κάθε πράκτορα ατομικά.Η διατήρηση αυτής της δομής ενισχύει την προσαρμοστικότητα των πρακτόρων στην γενική πεποίθηση του πληθυσμού για το βέλτιστο πρόγραμμα , καθώς κατασκευάζουν τις λύσεις τους επαναληπτικά.Οι συγγραφείς κάνουν χρήση των κανόνων μετάβασης και ανανέωσης της φερομόνης του Ant System αλγόριθμου.

Πιο πρόσφατα οι Mandahawi et al(2011) πρότειναν ένα σύστημα ευφυών πρακτόρων βασιζόμενο στις αποικίες μυρμηγκιών για την ελάχιστοποίηση της συνολικής βραδύτερης περάτωσης(Td) για την επίλυση του προβλήματος χρονοπρογραμματισμού μιας μηχανής με εξαρτώμενους χρόνους περάτωσης.Οι πράκτορες του συστήματος κινούνται σε ένα περιβάλλον γράφου για την κατασκευή προγραμμάτων.Ορίζονται πάνω και κάτω όρια για τον περιορισμό της εναπόθεσης φερομόνης κατα την διαδικασία ανανέωσης ώστε να μήν εμφανιστεί στασιμότητα(stagnation) κατα την επίλυση του προβλήματος.Τέλος ορίζουν κατάλληλα την ευρετική πληροφορία για κάθε μετάβαση στον γράφο με σκοπό την βέλτιστη επιλογή του επόμενου συστατικού λύσης από τον πράκτορα.Οι αλγοριθμική διαδικασία του συστήματος χρησιμοποιεί τους κανόνες ανανέωσης και μετάβασης του MAX-MIN Ant System αλγόριθμου(T. Stützle, H. Hoos,2000).

Είναι εμφανές από τα τελευταία δύο παραδείγματα ότι τα συστήματα βελτιστοποίησης με αποικίες μυρμηκιών βασίζουν την σχεδιασή τους στις έννοιες τις στιγμεργίας και αυτοοργάνωσης ,ώστόσο οι τεχνητοί πράκτορες που αντιπροσωπεύουν τον πληθυσμό των μυρμηγκιών στο τεχνητό περιβάλλον(γράφο) επιλέγουν να κινηθούν και να ανανέωσουν την φερομόνη στο περιβάλλον ,ανάλογα με τις διάφορες παραλλαγές των ACO(AS,MAX-MIN) αλγόριθμων που χρησιμοποιούν ως βάση για τον ορισμό των κανόνων ανανέωσης και μετάβασης τους.Στην παρούσα εργασία θα ακολουθήσουμε το αλγοριθμικό πλαίσιο του Ant Colony System(ACS) κατα την σχεδίαση του συστήματος για την επίλυση του JSP.

2.4.2 Πολυκριτηριακός χρονοπρογραμματισμός

Η πλειοψηφία της έρευνας στον χρονοπρογραμματισμο παραγωγής ασχολείται με την ελαχιστοποίηση ενος κριτηρίου ,του makespan.Ωστόσο πρακτικά σε πολλες βιομηχανίες όπως ηλεκτρονικών ,αεροσκαφών κλπ. παρατηρείται ανταπόδωση(trade off) τιμών στα διάφορα προβλήματα χρονοπρογραμματισμού που αντιμετωπίζουν , όπου πολλαπλές αντικειμενικές συναρτήσεις πρέπει να λήφθουν υπόψη με σκοπό την βελτιστοποίηση της συνολικής απόδοσης του συστήματος. Φυσικά τα πολυκριτηριακά προβλήματα χρονοπρογραμματισμού είναι πιο πολύπλοκα από τα μονοκριτηριακά και η έυρεση μιας συμβιβαστικής λύσης γίνεται δύσκολη υπόθεση αφού οι αντικειμενικές είναι συχνά ασυνεπής(incosistent), συγκρουόμενες(conflicting) ή ακόμα κα αντιφατικές(contradictory). Η βελτιστοποίηση δηλαδή ενός κριτηρίου γενικά οδηγεί στην επιδείνωση ενός άλλου.Για παράδειγμα η αύξηση του ρυθμού εισόδου ενός προιόντος οδηγεί σε υψηλότερη απόδοση, αλλά ταυτόχρονα σε αυξανόμενη work-in process(WIP).Λόγω λοιπόν της αυξανόμενης πολυπλοκότητας έχει δοθεί αρκετή προσοχή στα πολυκριτηριακά προβλήματα χρονοπρογραμματσιμού στην παραγωγή.

Ο πολυκριτηριακός χρονοπρογραμματισμός αποτέλεσε αντικείμενο έρευνας από τις αρχές της δεκαετίας του '70 σε διάφορα επιστημονικά πεδία.Εάν θεωρήσουμε περισσότερα του ενός κριτήρια σέ ένα πρόβλημα συνδυαστικής βελτιστοποίησης ,τότε έχουμε να κάνουμε με ένα πρόβλημα πολυκριτηριακής βελτιστοποίησης.Η πλειοψηφία των προβλημάτων αποφάσεων του πραγματικού κόσμου περιλαμβάνουν πολλαπλές και συγκρουόμενες αντικειμενικές συναρτήσεις οι οποίες οποίες πρέπει να βελτιστοποιηθούν ικανοποιώντας παράλληλα τους διάφορους περιορισμούς του προβλήματος.Στα πολυκριτηριακά προβλήματα λοιπόν δεν υπάρχει μια μοναδική λύση οι οποία να είναι βέλτιστη προς όλα τα κριτήρια.Υπάρχει ανταυτού ένα σύνολο λύσεων το οποίο είναι καλύτερο από τις υπόλοιπες λύσεις στον χώρο αναζήτησης όταν λαμβάνουμε υπόψη όλες τις αντικειμενικές ,ωστόσο κάθε λύση του συνόλου είναι κατώτερη μιας άλλης ως προς την τιμή ενός ή περισσοτέρων αντικειμενικών και αυτές οι λύσεις καλούνται μη-κυριαρχούμενες.

Οι πολυκριτηριακή βελτιστοποίηση με συγκρουόμενα κριτήρια παράγει ένα σύνολο βέλτιστων λύσεων ,παρα μια βέλτιστη λύση, στο οποίο ανήκουν λύσεις οι οποίες δεν είναι καλύτερες από κάποια άλλη σε όλα τα κριτήρια.Σε αυτή την περίπτωση η λύση καλείται Pareto βέλτιστη λύση.Τις ιδιότητες της Pareto βελτιστότητας θα αναλύσουμε στο επόμενο κεφάλαιο.

Η κλασσική προσέγγιση σε προβλήματα χρονοπρογραμματισμού ορίζει ένα λήπτη αποφάσεων ο οποίος αποφασίζει για το σύνολο των εργασιών και ενδιαφέρεται για την ανακατανομή τους με τον πιο επικερδή τρόπο,την βελτιστοποίηση δηλαδή ενός συγκεκριμένου κριτηρίο(Agnetis , Pacciarelli & Pacifici,2006). Στην περίπτωση πολλαπλών κριτηρίων το πρόβλημα ανάγεται σε πολυκριτηριακό,αλλά και πάλι τα

προβλήματα αποφάσεων και οι αντίστοιχοι αλγόριθμοι βελτιστοποίησης χειρίζονται βάση της ίδιας κεντροποιημένης αντίληψης.Η παραπάνω προσέγγιση διαφοροποιείται στη περίπτωση του πολυπρακτορικού χρονορπρογραμματισμού,όπου η διαδικασία κατανομής πόρων περιλαμβάνει διαφορετικούς πράκτορες καθένας από τους οποίους είναι υπεύθυνος για ένα υποσύνολο εργασιών και έχει την δική του αντικειμενική συνάρτηση.Οι πράκτορες θα πρέπει να μοιραστούν τις μηχανές μεταξύ τους ,ενώ δεν υπάρχει ένας ανώτερος επιβλέπων, υπέυθυνος για την λύση των συγκρούσεων σε περίπτωση ζήτησης κοινών μηχανών. Ένα τέτοιο πρόβλημα συνιστά ένα πολυπρακτορικό πρόβλημα χρονοπρογραμματισμού(multiagent scheduling problem) και ανάγεται στο πρόβλημα ανακατανομής πόρων ανάμεσα στους πράκτορες.Διαφέρει από ένα πρόβλημα πολυκριτηριακού χρονοπρογραματισμού καθώς στο τελευταίο κάθε εργασία συνεισφέρει το ίδιο σε όλα τα κριτήρια,ενώ σε ένα πολυπρακτορικό πρόβλημα η κάθε εργασία συνεισφέρει σε συγκεκριμένο κριτήριο(πχ το κριτήριο του πράκτορά της, Pinedo,2005).Οι μεθόδοι σχεδιασμού πολυπρακτορικών συστημάτων για την τελευταία περίπτωση προέρχονται κυρίως απο την θεωρία παιγνίων και δεν θα μας απασχολήσουν σε αυτή την διπλωματική.

Η σχεδίαση και ανάλυση του συστήματος πρακτόρων σε αυτή την διπλωματική προσεγγίζει και επιλύει το πολυκριτηριακό πρόβλημα χρονοπρογραμματισμού JSP με χρήση μεθόδων εμπνευσμένες από τη φύση βασιζόμενες σε ευφυείς πράκτορες ,θα επιχειρήσουμε λοιπόν μια επισκόπηση της δουλειάς που έχει γίνει πρός αυτή την κατεύθυνση.

Έχουμε ήδη περιγράψει στις Ενότητες 2.2.3 και 2.2.4 πως πράκτορες οι οποίοι αναπαριστούν λύσεις ενός προβλήματος συνδιαστικής βελτιστοποίησεις μπορούν να συνεργαστούν και να συντονιστούν χρησιμοποιώντας διαδικασίες εμπνευσμένες από την φύση.Οι γενετικοί αλγόριθμοι αποτελούν παράδειγμα αυτών των αλγόριθμων και οι Arroyo et al(2005) παρουσίασαν ένα γενετικό αλγόριθμο τοπικής αναζήτησης με στοιχεία όπως διατήρηση της διασποράς στον πληθυσμό και elitism για ένα flow shop πρόβλημα χρονοπρογραμματισμού,έτσι ώστε να παρέχεται στον ληπτη αποφάσεων μια προσέγγιση του pareto βέλτιστου μετώπου.

Ο Kazi(2006) παρουσίασε μια εξελικτική διαδικασία για την λύση του πολυκριτηριακού JSP χρησιμοποιώντας έναν γενετικό αλγόριθμο αναπηδώντων γονιδίων(Jumping genes Genetic Algorithm).Ο αλγόριθμος αναζητά ευριστικά σχεδόν βέλτιστα χρονοπρογράμματα που βελτιστοποιούν πολλάπλα κριτήρια ταυτόχρονα.

Ο Griimm et al (2011) πρότεινε ένα σύστημα χρονοπρογραματισμου βασιζόμενο σε πράκτορες εμπνευσμένο από την φυσική αλλήλεπίδραση του κυνηγού και του θηράματος για την λύση πολυκριτιριακών προβλημάτων χρονοπρογραματισμού.Στο προτεινόμενο σχήμα η οντότητα του κυνηγού αντιπροσωπεύεται από ένα πράκτορα και αναφέρεται σε ένα συγκεκριμένο κριτήριο.Ο κυνηγός διαβαίνει τυχαία σε ένα γράφο του οποίου οι κόμβοι αποτελούν τα θηράματα και επιλέγει σε κάθε βήμα του μια γειτονία θηραμάτων από την οποία αντικαθιστά το χειρότερο θήραμα (βάση του κριτηρίου που βελτιστοποιεί)με το καλύτερο χρησιμοποιώντας ένα μηχανισμό μεταλλάξης.Ο αλγόριθμος κάνει χρήση χειριστών ποικιλότητας(mutation operators) οι οποίοι χρησιμεύουν ως ευρετικοί κανόνες τοπικής αναζήτησης ώστε να επιταχυνθεί η σύγκλιση σε κάποιο φράγμα του pareto μετώπου.

Οι Duan et al(2007) παρουσίασαν ένα πολυπρακτορικό εξελικτικό αλγόριθμο για το πολυκριτιριακο πρόβλημα του JSP(multiagent evolutionary algorithm,MAEA).Στον MAEA ο κάθε πράκτορας αναπαριστά μια υποψήφια λύση και όλοι οι πράκτορες

κατοικούν σε ένα δικτυωτό περιβάλλον(lattice enviroment).Οι πράκτορες επικοινωνούν και αλληλεπιδρούν με τους γειτονές τους ακολουθώντας τρεις σχεδιασμένες συμπεριφορές:Εφόσον οι πόροι(μηχανές) είναι περιορισμένοι και οι συμπεριφορές των πρακτόρων οδηγούνται απο τον σκοπό τους ο πράκτορας ανταγωνίζεται με τους υπόλοιπους για να εξασφαλίσει όσο το δυνατόν περισσότερους πόρους.Για τον λόγο αυτό τρείς συμπεριφορές εμπνευσμένες από την εξελικτική διαδικασία σχεδιάστηκαν ώστε να ενισχύσουν τους πράκτορες στην επίτευξη του στόχου τους.Οι πράκορες επιδεικνύουν ανταγωνιστική συμπεριφορά στοχεύοντας έτσι στην εξάλειψη των πρακτόρων που αντιπροσωπευόυν χειρότερες λύσεις(πχ. κυριαρχούμενες).Επιδεικνύουν συμπεριφορά αυτοεκμάθησης(self learning) χρησιμοποιώντας μεθόδους τοπικής αναζήτησης για την βελτιώση των λύσεων τους και τέλος οδηγούνται από μια συμπεριφορά μετάλλαξης των μη κυριαρχόυμενων λύσεων ώστε να διατηρηθεί η διαφοροποίηση του Pareto συνόλου.

Οι Cardon et al(1998) πρότειναν ένα πολυπρακτορικό σύστημα για το πολυκριτηριακό JSP συγχωνεύοντας την γενετική διαδικασία για την παραγωγή πρακτόρων διαφόρων επιπέδων.Το πολυπρακτορικό σύστημα προσομειώνει την συμπεριφορά των συνεργαζόμενων πρακτόρων με σκοπό την ανάληψη ενεργειών στο διάγραμμα Gannt για την βελτιστοποίηση συγκεκριμένων κριτηρίων.Η επικοινωνία ανάμεσα σε τοπικούς και ολικούς πράκτορες ,στοιχεία του πολυπρακτορικού συστήματος, αναδεικνύει την εμφάνιση πρακτόρων ενδιάμεσης λειτουργικότητας (intermediate granulity), ονομαστικά micro- και meta-πρακτόρων οι οποίοι βελτιστοποιούν τα διάφορα κριτήρια βάση ενός πολυκριτηριακού γενετικού αλγόριθμου.

Όπως βλέπουμε η έρευνα στο πεδίο του χρονοπρογραμματισμού παραγωγής με χρήση πρακτορικών μεθόδων εμπνευσμένων από την φύση είναι περιορισμένη.Από τα συστήματα που παρουσιάσαμε τα περισσότερα είναι τύπου 2(Kazi, Griimm, Duan) εκτός του πολυπρακτορικού συστήματος του Cardon όπου η χρήση πολυπρακτορικού συστήματος και του γενετικού αλγόριθμου γίνεται ακολουθιακά είναι δηλαδή ένα σύστημα τύπου 3.

2.5 Στόχοι της διπλωματικής

Ο σχεδιασμός του δικού μας συστήματος ευφυών πρακτόρων αποτελεί εκ της φύσης του ένα σύστημα τύπου 2.Οι αποικίες μυρμηγκιών μοντελοποιούνται ως τεχνητοί πράκτορες και αντιπροσωπέυουν λύσεις του προβλήματος. Όπως προείπαμε τα συστήματα τύπου 2 ενσωματώνουν κάποια στοιχεία των πρακτόρων και πολυπρακτορικών συστημάτων υπο το αλγοριθμικό πλαίσιο της φυσικής διαδικασίας. Βρίσκουμε ότι αλγόριθμοι συνδιαστικής βελτιστοποίησης ACO(Ant Colony Optimization) προσφέρουν τα κατάλληλο αλγόριθμικό πλαίσιο για την σχεδίαση τέτοιων συστημάτων.Η φυσική διαδικασία μοντελοποιείται από την διαδικασία εύρεσης τρόφής των μυρμηγκιών στη φύση,ενώ παράλληλα τα μυρμήγκια κληρονομούν κάποιες ιδιότητες των MAS όπως ο πληθυσμός των μυρμηγκιών, αυτοοργάνωση, προσαρμοστικότητα(adaptation) και αυτοεκμάθηση (self-learning).

Στο 4^{ο} κεφάλαιο θα αναλύσουμε λεπτομερώς την προτεινόμενη μεθοδολογία μοντελοποίησης του συστήματος ευφυών πρακτόρων αρχίζοντας από την σχεδίαση του μοντέλου του ACO-πράκτορα μας καταλήγωντας στην μοντελοποίηση του συστήματος των ACO-πρακτόρων βασιζόμενοι πάντα στην γενική ACO μεθευρετική διαδικασία που παρουσιάσαμε στην ενότητα 2.2.2.Το σύστημα μας χρησιμοποιεί μια νέα για την επίλυση του JSP ,αλγοριθμική παραλλάγη της ACO οικογένειας ,τον Ant

Colony System,ACS(εισάγει νέους κανόνες μετάβασης και ανανέωσης φερομόνης),άρα πρωταρχικός σκοπός μας είναι να δοκιμάσουμε την απόδοση του προτεινόμενου συστήματος στο μονοκριτηριακό JSP για δυό διαφορετικά κριτήρια εξίσου σημαντικά στην βιομηχανία,το makespan και τον συνολικό χρόνο βραδύτερης περάτωσης.Βάση των αποτελεσμάτων να το προτείνουμε σαν μια εναλλακτική μέθοδο επίλυσης του JSP και να σημειώσουμε πιθανές προτάσεις βελτίωσεις του.

Σε 2η φάση προεκτείναμε το μονοκριτηριακό σύστημα για το πολυκριτηριακό JSP στοχεύοντας στην ταυτόχρονη ελαχιστοποίηση των δυό κριτηρίων με στόχο την σχεδίαση ενός συστήματος ευφυών πρακτόρων για την παραγωγή ενός βέλτιστου Pareto μετώπου.

Κεφάλαιο 2 Θεωρητικό Υπόβαθρο

3.1 Μοντελοποίηση του Προβλήματος

Το ντετερμινιστικό πρόβλημα του χρονοπρογραμματισμού εργασιών παραγωγης αποτελείται από ένα καθορισμένο σύνολο J από n εργασίες $\{J_i\}_{i=1}^{n}$ που πρέπει να δρομολογηθούν σε ένα καθορισμένο σύνολο M από m μηχανές $\{M_k\}_{k=1}^{m}$. Η κάθε εργασία J_i πρέπει να δρομολογηθεί σε όλες τις μηχανές και έτσι προκύπτει ένα σύνολο από m_i διεργασίες $O_{i1}, O_{i2}\ldots O_{imi}$, που πρέπει να προγραμματιστούν με μια προκαθορισμένη σειρά. Συνολικά υπάρχουν N διεργασίες , $N=\sum_{i=1}^{n} m_i$. O_{ik} είναι η διεργασία της εργασίας J_i που υλοποιείται στην μηχανή M_k για μια μη διακοπτόμενη περίοδο χρόνου τ_{ik}. Κάθε εργασία είναι ανεξάρτητη από τις υπόλοιπες εργασίες. Έτσι,το πρόβλημα πλαισιώνουν διάφοροι ποσοτικοί περιορισμοί που εξασφαλίζουν πως κάθε μηχανή μπορεί να αναλάβει μόνο μια διεργασία και κάθε διεργασία μπορεί να υλοποιηθεί μόνο από μια μηχανή την φορά. Αν ο χρόνος ολοκλήρωσης της εργασίας J_i στην μηχανή M_k είναι C_{ik} τότε η συνολική διάρκεια όλων των διεργασιών θα αναφέρεται σαν εύρος C_{max} , "makespan". Ο ουσιαστικός στόχος του προγραμματιστή αυτής της γραμμής παραγωγής είναι να καθορίσει χρόνους έναρξης για την κάθε διεργασία, $t_{ik} \geq 0$, με σκοπό την ελαχιστοποίηση του εύρους του συνολικού χρόνου ικανοποιώντας όμως παράλληλα όλους τους περιορισμούς. Το πρόβλημά μας λοιπόν μπορεί να μοντελοποιηθεί ως πρόβλημα ακέραιου προγραμματισμού και θα έχει την ακόλουθη μορφή :

$$C_{max} = minimize\ (C_{max}) = min\ (max\ (t_{ik} + \tau_{ik})):\ \ \forall\ J_i \in J,\ M_k \in M \qquad (3.1)$$

Subject to :

$$t_{ik} - t_{ih} \geq \tau_{ih} \quad , 1 \leq k,h \leq m,\ 1 \leq i \leq n\ . \qquad (3.2)$$

$$t_{ik} - t_{pk} \geq \tau_{pk} + \gamma * (1 - Y_{pik}) \quad , 1 \leq k \leq m\ ,\ 1 \leq i,p \leq n\ . \qquad (3.3)$$

$$t_{pk} - t_{ik} \geq \tau_{ik} + \gamma * (1 - Y_{pik}) \quad , 1 \leq k \leq m\ ,\ 1 \leq i,p \leq n\ . \qquad (3.4)$$

$$t_{ik} \geq 0\ , \qquad (3.5)$$

$$Y_{pik} = \{0,1\}\ , \qquad (3.6)$$

Ο περιορισμός (2.2) διασφαλίζει ότι διατηρούνται οι περιορισμοί προτεραιοτήτων ανάμεσα στις διεργασίες μιας εργασίας και επιπλέον οτι δεν επικαλύπτονται χρονικά κατα την εκτέλεσή τους ή εναλλακτικά ότι μια εργασία υπόκειται σε επεξεργασία μόνο από μια μηχανή κάθε φορά.Ο δυαδικοί περιορισμοί (2.3) και (2.4) εξασφαλίζουν ότι κάθε μηχανή αναλαμβάνει προς επεξεργασία μόνο μια εργασία τη φορά.Εφόσον οι δύο περιορισμοί ειναι περιοριστικοί ο ένας για τον άλλον, εισάγουμε την δυαδική μεταβλητή Y_{pik} , όπου Y_{pik} = 1 ,εάν η εργασία p προηγείται της i στη σειρά επεξεργασίας (2.3) και Y_{pik} = 0 διαφορετικά (2.4).Τέλος ο περιορισμός (2.5)

εξασφαλίζει ότι όλες οι εργασίες ολοκληρώνουν την επεξεργασία τους.
*Οι παραπάνω περιορισμοί ισχύουν και για τον συνολικό χρόνο βραδύτερης περάτωσης(**total tardiness,T_d**) ως αντικειμενικό κριτήριο - κατα αντιστοιχία με τον σταθμισμένο συνολικό χρόνο βραδύτερης περάτωσης χωρίς να θέσουμε συγκεκριμένα βάρη στις εργασίες - ,όπου εάν θεωρήσουμε ως d_j τον χρόνο παράδοσης της εργασιας j τότε η ο χρόνος βραδύτερης περάτωσης της εργασίας j ορίζεται ως $T_j = max(C_j - d_j , 0)$ και η αντικειμενική συνάρτηση προς ελαχιστοποίηση θα είναι:*

Η σημασία της ελαχιστοποίησης αυτού του μέτρου στον χρονοπρογραμματισμό είναι ιδιαίτερα σημαντική.Ο χρόνος παράδοσης μιας εργασίας δείχνει τον χρόνο κατα τον οποίο μια εργασία πρέπει να ολοκληρωθεί ,ώστε να ικανοποιηθεί ο προβλεπόμενος χρόνος παράδοσης ενος έργου.Χρησιμοποιείται κυρίως για να βελτιώσει το επιπεδο εξυπηρέτησης του πελάτη (Kuluc and Khraman, 2006; Cicirello and Smith, 2001) ,καθώς η ικανοποιησή του στον ανταγωνιστικό χώρο της βιομηχανίας ειναι μείζονος σημασίας.Επίσης χρόνοι παράδοσης χρησιμοποιούνται σε περιπτώσεις οπου τίθονται προτεραιότητες για την ολοκλήρωση μιας εργασίας βάση των χρονικών απαιτήσεων ενός πελάτη.Σε αυτές τις περιπτώσεις το makespan δεν αποτελεί το πρωτεύον κριτήριο για την επίτευξη του στόχου(Baykasoglu et al., 2008).Συνοψίζοντας μια εργασία που καθυστερεί την περάτωση της μπορεί να βλάψει την φήμη μιας εταιρίας και να περιορίσει την ικανοποίηση του πελάτη.Για το λόγο αυτό ο έλεγχος των χρόνων παράδοσης των εργασιών είναι μια απο τις μεγαλύτερες προκλήσεις που αντιμετωπίζουν οι εταιρείες σήμερα(Jayamohan, M.S., Rajendran, C.,2004).
Θα πρέπει επίσης να σημειωθεί πως οι διαστάσεις του προβλήματος JSP,είναι $n \times m$ και ότι το N είναι $n \times m$ αρκεί το $m_i = m$ για κάθε εργασία $J_i \in J$, δηλαδή ο αριθμός των διεργασιών m_i είναι ίσος με τον αριθμό των μηχανών m και ότι κάθε εργασία θα πρέπει να υπόκειται σε επεξεργασία ακριβώς μία φορά σε κάθε μηχανή. Σε μια γενικότερη μορφή του προβλήματος οι επαναλήψεις των μηχανών (ή η απουσία τους) επιτρέπονται στη δεδομένη σειρά των εργασιών $J_i \in J$ και έτσι το m_i μπορεί να είναι μεγαλύτερο (ή μικρότερο) από το m(Jain & Meeran,1998).

Το πλήρες σύνολο των περιορισμών που θα πρέπει να ικανοποιούνται στο βασικό JSP δίνεται παρακάτω(French, 1982):

1. *Δεν γίνεται δυο διεργασίες της ίδιας εργασίας να επεξεργάζονται ταυτόχρονα.*
2. *Δεν επιτρέπεται διακοπή(pre-emption) της επεξεργασίας των διεργασιών.*
3. *Όλες οι εργασίες επεξεργάζονται το πολύ μια φορά σε κάθε μηχανή.*
4. *Όλες οι εργασίες επεξεργάζονται μέχρι την ολοκλήρωσή τους.*
5. *Οι εργασίες μπορούν να εκκινήσουν την επεξεργασία τους οποιαδήποτε χρονική στιγμή,δεν υπάρχουν χρόνοι εκκίνησης(release dates) για τις εργασίες.*
6. *Οι εργασίες μπορούν να ολοκληρώσουν οποιαδήποτε χρονική στιγμή, δεν υπάρχουν χρόνοι παράδοσης (due dates) για τις εργασίες.*

7. *Μια εργασία θα πρέπει να αναμένει την επόμενη διαθέσιμη μηχανή.*
8. *Καμία μηχανή δεν μπορεί να επεξεργαστεί παραπάνω από μια διεργασία τη φορά.*
9. *Οι χρόνοι εξάρμωσης(setup times)των μηχανών είναι αμελητέοι.*
10. *Κάθε μηχανή είναι μοναδική.*
11. *Οι μηχανές μπορεί να παραμένουν αδρανείς κατα την διάρκεια του χρονοπρογραμματισμού.*
12. *Οι μηχανές είναι διαθέσιμες οποιαδήποτε χρονική στιγμή.*
13. *Οι περιορισμοί προτεραιοτήτων είναι γνωστοί εκ των προτέρων και είναι αμετάβλητοι.*

Ένα πολυκριτηριακό πρόβλημα χρονοπρογραμματισμού job-shop με δύο κριτήρια μπορεί να διατυπωθεί μαθηματικά ως εξής:

Minimize $F= [F^1 \; F^2]^T$

Subject to

$$t_{ik} - t_{ih} \geq \tau_{ih} \quad , 1 \leq k,h \leq m, \; 1 \leq i \leq n \; .$$

$$t_{ik} - t_{pk} \geq \tau_{pk} + \gamma * (1 - Y_{pik}) \quad , 1 \leq k \leq m \; , \; 1 \leq i,p \leq n \; .$$

$$t_{pk} - t_{ik} \geq \tau_{ik} + \gamma * (1 - Y_{pik}) \quad , 1 \leq k \leq m \; , \; 1 \leq i,p \leq n \; .$$

$$t_{ik} \geq 0 \; ,$$

$$Y_{pik} = \{0,1\} \; ,$$

,*όπου οι περιορισμοί του πολυκριτηριακού προβλήματος είναι οι περιορισμοί που διέπουν το βασικό JSP και F είναι το δισδιάστατο διάνυσμα με τις αντικειμενικές .****Στη συγκεκριμένη εργασία στοχεύουμε στην ταυτόχρονη ελαχιστοποίηση των***

- *F^1=makespan(C_{max}) και*
- *F^2=T_d= ,του συνολικού χρόνου βραδύτερης περάτωσης(T_d).*

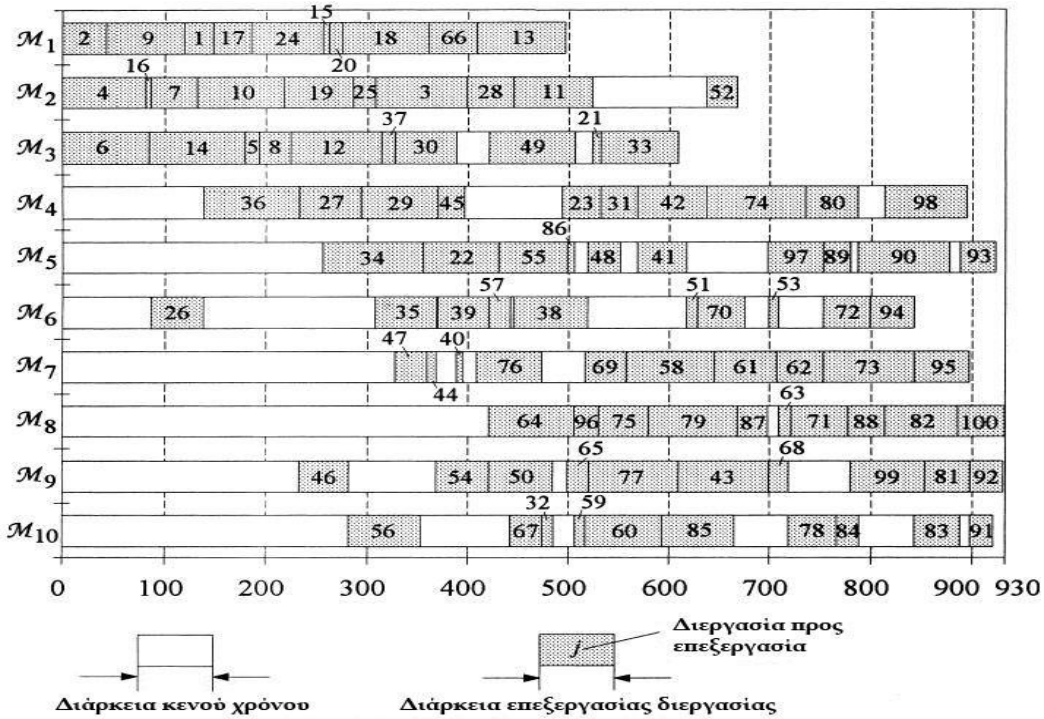
Για τα παραπάνω δύο κριτηρία δεν χρειάζεται να προσθέσουμε επιπλέον περιορισμούς στο μαθηματικό μοντέλο.

*Το σύνολο ωστόσο των περιορισμών σε εφαρμογές του πραγματικού κόσμου είναι αρκετά πιο περίπλοκο .Πρακτικά μόνο ορισμένοι από τους παραπάνω περιορισμούς συνεχίζουν να ισχύουν,ενώ τυπικές προεκτάσεις του βασικού JSP περιλαμβάνουν παράλληλες μηχανές,μηχανές πολλαπλών τρόπων (multipurpose) , βλάβες μηχανών και χρονικά παράθυρα (**time windows**) ,τα οποία εισάγονται στο σύστημα ορίζοντας χρόνους εκκίνησης(release dates) και παράδοσης(due dates) για τις εργασίες.Επίσης ορίζουμε και προβλήματα δυναμικού χρονοπρογραμματισμού ,όπου οι εργασίες εισέρχονται στοχαστικά στο σύστημα της παραγωγικής διαδικασίας.Τέλος στον μη-ντετερμινιστικό χρονοπρογραμματισμό οι χρόνοι/περιορισμοί επεξεργασίας των διεργασιών είναι δυνατόν να ανανεώνονται κατα την διάρκεια της διαδικασίας(πχ εξαρτώμενοι χρόνοι εξάρμωσης των παραγγελιών).Ωστόσο ,παρα τις υποθέσεις περιορισμών που αναφέραμε παραπάνω, το βασικό JSP έχει καταφέρει να κερδίσει*

αρκετή φήμη λογω της δυσκολίας του να λυθεί και ειναι πιθανώς το πιο μελετημένο και καλά θεμελιωμένο μοντέλο στη ντετερμινιστική θεωρία του χρονοπρογραμματισμού ,το οποίο χρησιμεύει σαν ενα συγκριτικό μοντέλο για τις διάφορες τεχνικές επίλυσης ,καινούριες αλλά και παλαιότερες(Siang Lyn,2005).

3.2 Αναπαράσταση του Job Shop Problem

Μολονότι το διάγραμμα Gantt (Εικόνα 4.1) που περιγράφτηκε από τους Gantt (1919), Clark (1922) και Porter (1968) παραδοσιακά αποτέλεσε την πιο δημοφιλή μέθοδο αναπαράστασης της λύσης ενός προβλήματος χρονοπρογραμματισμού εργασιών, ο Blazewicz (1996) έδειξε ότι το διασπαστικό γραφικό μοντέλο (disjunctive graph model) G = {N,A,E} των Roy και Sussmann (1964) είναι πλέον αυτό που επικρατεί. Λαμβάνοντας υπόψη ότι στο μοντέλο αυτό στηρίζεται η πλειοψηφία των μεθόδων επίλυσης του JSP, κρίνουμε απαραίτητη μια σύντομη αναφορά στο μοντέλο αυτό.

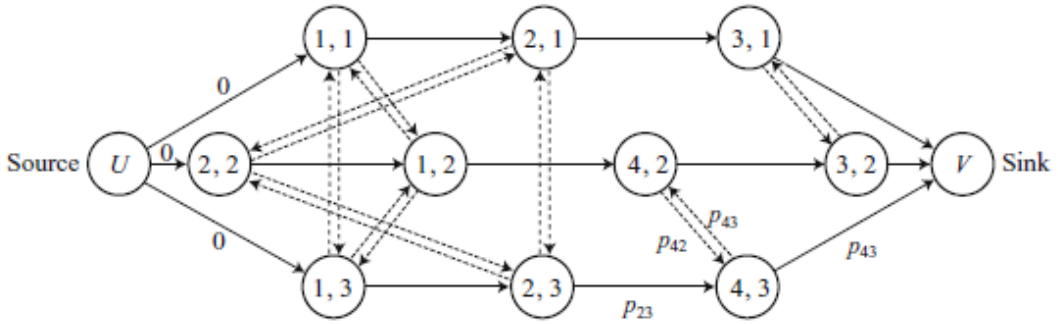


Εικόνα 3.1:Διάγραμμα Gantt για το βέλτιστο χρονοπρόγραμμα του FT10 benchmark (makespan 930).

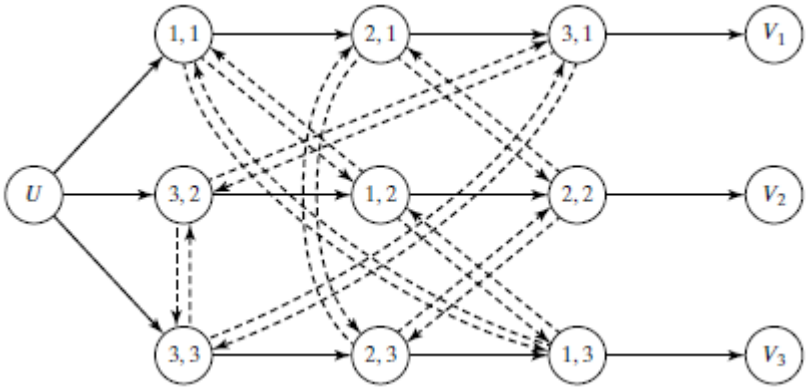
3.2.1 Γραφική Αναπαράσταση του Job Shop Problem (disjunctive graph model)

Το πρόβλημα του JSP για την ελαχιστοποίηση του C_{max} *μπορεί να αναπαρασταθεί με ένα διαζευκτικό γράφο* G*, ο οποίος περιέχει ενα σύνολο κόμβων* N *και δύο σύνολα ακμών* A *και* E *.Οι κόμβοι στο* N *αντιστοιχούν σε όλες τις διεργασίες* (i,j) *που πρέπει να περαιώσουν οι* n *εργασίες ,ενώ οι κατευθυνόμενες ακμές στο* A *αντιπροσωπεύουν τις διαδρομές των διεργασιών μιας εργασίας.Εάν για παράδειγμα η ακμή* $(i,j) \rightarrow (h,j)$ *ανήκει στο* A *,η εργασία* j *πρέπει να δρομολογηθεί στη μηχανή* i *,προτού δρομολογηθεί στη μηχανή* h*.Επιπλέον κάθε ζεύγος διεργασιών που ανήκουν σε διαφορετικές εργασίες και πρέπει να επεξεργαστούν στην ίδια μηχανή, συνδέονται με δυο διαζευκτικές(διακκεκομένες) ακμές που έχουν και τις δύο κατευθύνσεις.Οι*

διαζευκτικές λοιπόν ακμές που ανήκουν στο E σχηματίζουν m 'κλίκες' διπλών ακμών, όπου κάθε μια αντιστοιχεί σε μια μηχανή, ενώ οι διεργασίες (κόμβοι) που ανήκουν στην ίδια 'κλίκα' πρέπει να δρομολογηθούν στην ίδια μηχανή.Όλες οι ακμές που 'φεύγουν' από ένα κόμβο,συζευκτικές ή διαζευκτικές, έχουν ως μήκος τον χρόνο επεξεργασίας της διεργασίας που αντιπροσωπεύει ο κόμβος αυτός, ενώ τέλος ενσωματωμένοι μέσα στο N είναι δύο πλασματικοί κόμβοι U και V οι οποίοι αντιπροσωπεύουν την αρχική και τελική διεργασία του συστήματος και είναι γνωστοί ως πηγή(source) και καταβόθρα(sink) αντίστοιχα .Από την πηγή U 'φεύγουν' n κατευθυνόμενες ακμές προς όλες τις πρώτες διεργασίες των n εργασιών ,ενώ στην καταβόθρα V καταλήγουν αντίστοιχα n κατευθυνόμενες ακμές από όλες τις τελευταίες διεργασίες των εργασιών.Οι ακμές που 'φεύγουν' από την πηγή έχουν μήκος μηδέν,όπως φαίνεται στον γράφο της Εικόνας 3.2 .Ορίζουμε λοιπόν αυτόν τον γράφο ως $G = (N,A,E)$.



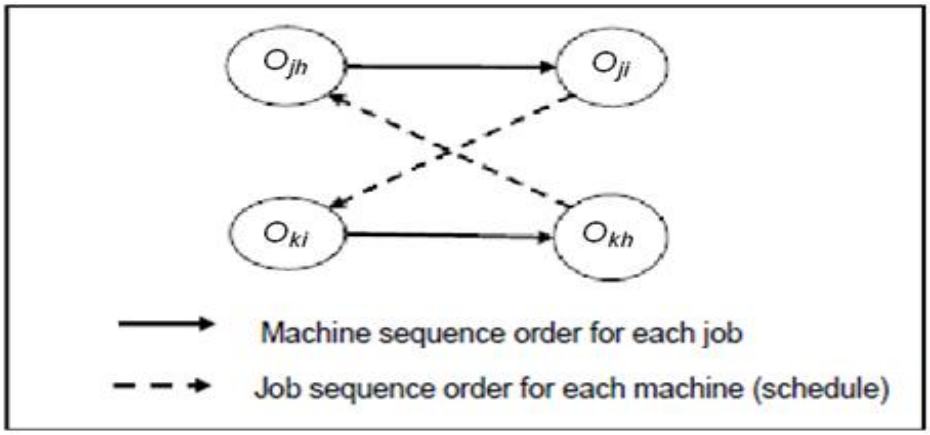
Εικόνα 3.2 :Κατευθυνόμενος γράφος ενός προβλήματος χρονοπρογραμματισμού JSP για την ελαχιστοποίηση του C_{max} ,(Pinedo,2005).



Εικόνα 3.3:Κατευθυνόμενος γράφος ενός προβλήματος χρονοπρογραμματισμού JSP για την ελαχιστοποίηση του ,(Pinedo,2005).

*Η έννοια του **εφικτού**(feasible) χρονοπρογράμματος ορίζει όλα τα έγκυρα χρονοπρογράμματα ,τα οποία πληρούν όλους τους χρονικούς περιορισμούς των αντίστοιχων εργασιών τους(Pinedo,2005).Με βάση λοιπόν τον παραπάνω ορισμό ένα εφικτό χρονοπρόγραμμα αντιστοιχεί στην επιλογή μιας ακμής από κάθε ζεύγος διαζευκτικών ακμών, έτσι ώστε ο τελικός κατευθυνόμενος γράφος να μη περιέχει κύκλο.Η επιλογή των ακμών αυτών υποννοεί ουσιαστικά ότι δεν πρέπει να υπάρχει κύκλος μέσα σε μια 'κλίκα' και καθορίζει παράλληλα την ακολουθία επεξεργασίας των διεργασιών σε μια συγκεκριμένη μηχανή.Η απαίτηση μη ύπαρξης κύκλου σε μια 'κλίκα' είναι προφανής ,καθώς στην αντίθετη περίπτωση η ακολουθία των διεργασιών*

σε μια συγκεκριμένη μηχανή δεν θα ήταν εφικτή.Ωστόσο η απαίτηση να μην υπάρχει κύκλος μεταξύ κατευθυνόμενων ακμών και διαζευκτικών ακμών από διαφορετικές 'κλίκες' δεν είναι τόσο εμφανής.Ένα πιθανό σενάριο που μπορεί να οδηγήσει σε κύκλο είναι το εξής: Έστω (h,j) , (i,j) δύο διεργασίες της εργασίας j και (i,k) , (h,k) δύο διεργασίες που ανήκουν στην εργασία k.Έστω ακόμα ότι σε ένα πιθανό χρονοπρόγραμμα η διεργασία (i,j) προηγείται της (i,k) στην μηχανή i (πχ. υπάρχει μια διαζευκτική ακμή (i,j) → (i,k)) και η διεργασία (h,k) προηγείται της (h,j) στη μηχανή h (πχ υπάρχει μια διαζευκτική ακμή (h,k) →(h,j)) ,τότε ο γράφος περιέχει κύκλο σχηματιζόμενο από τέσσερις ακμές,δυο κατευθυνόμενες και δυο διαζευκτικές ακμές από διαφορετικές 'κλίκες'(Εικ. 3.4). Το παραπάνω χρονοπρόγραμμα λοιπόν είναι μη εφικτό.Συνοψίζοντας εάν D είναι το υποσύνολο των επιλεγμένων διαζευκτικών ακμών στο E και ο γράφος G(D) ορίζεται ως G(D) =(A D), τότε το σύνολο D αντιστοιχεί σε ένα εφικτό χρονοπρόγραμμα ,αν, το G(D) δεν περιέχει κατευθυνόμενους κύκλους.



Εικόνα 3.4:Κύκλος σε ένα διαζευκτικό γράφο από διαφορετικές κλίκες (Mahanim Omar ,2008).

*Μπορούμε να καθορίσουμε ως C_{max} ενός εφικτού χρονοπρογράμματος το μακρύτερο μονοπάτι(**Crical Path**,κρίσιμο μονοπάτι) στο G(D) από την πηγή U μέχρι την καταβόθρα V,το οποίο περιέχει ένα σύνολο από διεργασίες, όπου η πρώτη ξεκινά την επεξεργασίας της την χρονική στιγμή μηδέν και η τελευταία ολοκληρώνει την χρονική στιγμή C_{max}.*

Κάθε στιγμή, η κάθε διεργασία j (εκτός της πηγής και της καταβόθρας) έχει ακριβώς δυο άμεσους προκατόχους και διαδόχους, τους προκατόχους της δουλειάς και της μηχανής (JP(j) και MP(j)) και τους διαδόχους της δουλειάς και της μηχανής (JS(j) και MS(j)). Άρα ο διάδοχος κάθε διεργασίας στο μονοπάτι αυτό ανήκει ειτε στην ίδια μηχανή με την διεργασία ή στην ίδια εργασία της διεργασίας αλλά δρομολογείται σε διαφορετική μηχανή.

*Σημαντικές παράμετροι ακολουθώντας την διασπαστική γραφική μέθοδο είναι η κεφαλή και η ουρά της κάθε διεργασίας. Η κεφαλή, ή αλλιώς η εναρκτήρια ημερομηνία(**ενωρίτερος χρόνος έναρξης**) , r_j , είναι το μήκος του μακρύτερου μονοπατιού, l, από την πηγή μέχρι την έναρξη O_j και ορίζεται από την σχέση r_j = l(U,j). Παρεμφερώς, το μήκος της μακρύτερης διαδρομής ή **ενωρίτερος χρόνος περάτωσης**. από την ολοκλήρωση της Oj μέχρι την καταβόθρα είναι η ουρά, q_j, και ορίζεται από την σχέση q_j = l(j,V) – τ_j. Έτσι, αν ένα από τα μακρύτερα(**κρίσιμα**) μονοπάτια του G περνάει από τις διεργασίες i και j και i<j τότε C_{max} = l(i,j) = r_i + τ_i + τ_j + q_j και οι διεργασίες i ,j ονομάζονται **κρίσιμες**.*

Συνοψίζοντας το πρόβλημα ελαχιστοποίησης του C_{max} ανάγεται στην επιλογή ενός

κατάλληλου συνόλου διαζευκτικών ακμών απο το E το οποίο ελαχιστοποιεί το μήκος του μακρύτερου μονοπατιού.Σύμφωνα λοιπόν με τα παραπάνω το πρόβλημα του job shop μπορεί εναλλακτικά να μοντελοποιηθεί ως πρόβλημα μαθηματικού προγραμματισμού χρησιμοποιώντας την μοντελοποίηση διαζευκτικού προγραμματισμού(disjunctive programming formulation,Pinedo 2005) και το μαθηματικό πρόγραμμα που ελαχιστοποιεί το C_{max} *είναι το εξής* :

$$\text{minimize} \quad C_{max}$$

subject to

$$t_{ik} - t_{ih} \geq \tau_{ih} \quad , \text{(conjunctive constraint)} \ \forall\ (k,i) \rightarrow (h,i) \in A$$

$$C_{max} - t_{ih} \geq \tau_{ih} \quad , \forall\ (h,i) \in N$$

$$t_{ik} - t_{pk} \geq \tau_{pk} \ \text{or}\ t_{pk} - t_{ik} \geq \tau_{ik} \quad , \text{(disjunctive constraint)} \ \forall (k,i) \text{ and } (k,p),\ k = 1, .\ .\ m.$$

$$t_{ih} \geq 0 \quad , \text{(earliest starting time constraint)} \ \forall\ (h,i) \in N$$

,όπου η μεταβλητή t_{ih} *ορίζει τον χρόνο έναρξης της διεργασίας* (h,i), N *είναι το σύνολο όλων των διεργασιών* (h,i), *και* A *το σύνολο όλων των περιορισμών προτεραιοτήτων* $(i,j) \rightarrow (h,i)$ *,οι οποίοι απαιτούν η εργασία* j *να αρχίσει την επεξεργασία της στη μηχανή* i *,προτού ξεκινήσει να επεξεργάζεται στη μηχανή* h.

Κατά αντιστοιχία για το πρόβλημα βραδύτερης περάτωσης μας ενδιαφέρουν οι χρόνοι περάτωσης όλων των εργασιών,άρα αντι μίας θα υπάρχουν n $(V_1,\ldots, V_n)$ *καταβόθρες, Εικόνα 3.3.Πλέον το μακρύτερο μονοπάτι από την πηγή U στην καταβόθρα* V_k *αντιπροσωπεύει τον χρόνο περάτωσης της εργασίας* k.

3.3 Κανόνες Προτεραιότητας (Dispathcing rules).

Σε αυτό το σημείο θα κάνουμε μια μικρή εισαγωγή στους κανόνες ανάθεσης και στην λειτουργικότητά τους σε προβλήματα χρονοπρογραμματισμού παραγωγής.

Ένας κανόνας ανάθεσης ή κανόνας διεκπεραίωσης προτεραιότητας είναι ένας κανόνας ο οποίος παρέχει προτεραιότητες στις εργασίες που περιμένουν να εκτελεστούν σε μια μηχανή(Pinedo,2005).Οι κανόνες ανάθεσης μπορούν να λάβουν υπόψιν τους τα χαρακτηριστικά της εργασίας ,της μηχανής όπως επίσης και τον τρέχων χρόνο άφιξης μιας εργασίας στο σύστημα.Κατηγοριοποιούνται με βάση διάφορα κριτήρια:Για παράδειγμα μπορούμε να τους διακρίνουμε σε ***στατικούς*** *και* ***δυναμικούς*** *κανόνες.Οι στατικοί δεν εξαρτώνται απο τον χρόνο,αλλά απο τα χαρακτηριστικά της εκάστοτε εργασίας και/ή μηχανής,όπως πχ. ο SPT(Shortest Processing Time) κανόνας,ο οποίος αναθέτει υψηλότερες προτεραιότητες στις εργασίες με το μικρότερο χρόνο εκτέλεσης σε μια συγκεκριμένη μηχανη.Από την άλλη οι δυναμικοί κανόνες ανάθεσης είναι εξαρτημένοι από τον χρόνο.Χαρακτηριστικό παράδειγμα είναι ο MST (Minimum Slack Time) κανόνας ,ο οποιος επιλέγει την επόμενη εργασία προς εκτέλεση σε μια μηχανή ,βάση του χρόνου ύφεσης της(slack time).Ο χρόνος ύφεσης μιας εργασίας ορίζεται ως η διαφορά μεταξύ του χρόνου παράδοσης μιας εργασίας(due date) και του υπολοιπόμενου χρόνου επεξεργασίας της απο το τρέχον χρονικό σημείο εκτέλεσης.Για παράδειγμα ο υπολοιπόμενος χρόνος ύφεσης μπορεί να υπολογιστεί ως* $\max(dj - pj - t, 0)$,*οπου* dj *ο χρόνος*

παράδοσης,p_j ο χρόνος επεξεργασίας και t το τρέχον χρονικό σημείο κατα το οποίο η μηχανή είναι διαθέσιμη.Σύμφωνα με τον παραπάνω τύπο είναι ευνόητο οτι μέχρι ένα σημείο μια εργασία j μπορεί να έχει μεγαλύτερη προτεραιότητα απο μια εργασία k,ενώ σε μεταγενέστερο χρόνο μπορεί να έχουν ίσες προτεραιότητες.
Ένας δεύτερος τρόπος κατάταξης των κανόνων ανάθεσης λαμβάνει υπόψη τη πληροφορία , βάση της οποίας ορίζονται οι διάφορες εργασίες. Ένας ***τοπικός*** *(local) κανόνας χρησιμοποιεί πληροφορία που σχετίζεται είτε με την ουρά αναμονής στην οποια βρίσκεται η εργασία,είτε με την μηχανή απο την οποία περιμένει να εκτελεστεί.Χαρακτηριστικό παράδειγμα ειναι ο FIFO (First In First Out) .Ένας* ***γενικός*** *(global) κανόνας χρησιμοποιεί πληροφορίες που σχετίζονται με άλλες μηχανές,οπως πχ τον χρόνο εκτέλεσης μιας εργασιάς στην επόμενη μηχανή που θα επισκεφτεί.Παράδειγμα τέτοιου κανόνα είναι ο SQNO (Shortest Queue at the Next Operation),όπου μια μηχανή επιλέγει την εργασία ,της οποίας στο τρέχον χρονικό σημείο , η διεργασία προς εκτέλεση στην επόμενη μηχανή βρίσκεται πιό μπροστα στην ουρά αναμονής.Τέλος παραθέτουμε ενα επιπρόσθετο δείγμα κανόνων ανάθεσης(Pinedo,2005):*

1. *Ο Longest Processing Time first (LPT) κανόνας :Εκτελούνται πρώτα οι εργασίες με το μεγαλύτερο χρόνο επεξεργασίας.Έχει το πλεονέκτημα οι εργασίες με το μικρότερο χρόνο εκτέλεσης να εκτελούνται σε μεταγενέστερο σημείο ,καθώς ειναι χρήσιμες στο να εξισορροπούν προς το τέλος τον φόρτο εργασίας της επεξεργασίας.*
2. *Ο Shortest Setup Time first (SST) κανόνας: Όταν μια μηχανή αποδεσμεύται ,ο κανόνας επιλέγει προς εκτέλεση την εργασία με τον μικρότερο χρόνο προετοιμασίας(setup time).*
3. *Ο Earliest Release Date first (ERD) κανόνας:Είναι ισοδύναμος του γνωστού First-Come-First-Served κανόνα και ελαχιστοποιεί κατα μια έννοια τις διαφοροποιήσεις στους χρόνους αναμονής των εργασιών σε μια μηχανή.*
4. *Ο The Earliest Due Date first (EDD) κανόνας:Η εργασία με το μικρότερο χρόνο παράδοσης επιλέγεται προς εκτέλεση.Ο κανόνας στοχεύει στην ελαχιστοποίηση της μέγιστης καθυστέρησης των εργασιών που περιμένουν να εκτελεστούν, κλπ.*

Ένας κανόνας ανάθεσης λοιπόν χρησιμοποιείται , για να επιλεχθεί και να προωθηθεί η επόμενη εργασία με την μεγαλύτερη προτεραιότητα ,από ένα σύνολο εργασιών που περιμένουν να ικανοποιηθούν από μια μηχανή που είναι αποδεσμευμένη.Η προτεραιότητα της κάθε εργασίας καθορίζεται βάση των χαρακτηριστικών της εργασίας και της μηχανής .Οι κανόνες ανάθεσης είναι συνήθως γρήγοροι ,διαισθητικοί και εύκολοι στην υλοποίηση(Queldhadj & Petrovic,2008).

3.4 Job Shop Benchmark Problems

Προκειμένου να ορίσουμε συγκριτικά μέτρα των διαφόρων αλγορίθμων και τεχνικών για την επίλυση του JSP ,θα πρέπει να τους εφαρμόσουμε σε κοινά στιγμιότυπα του προβλήματος(Siang Lyn,2005). Τα benchmark προβλήματα παρέχουν μια κοινή πλατφόρμα για τη δοκιμή και βελτίωση των αλγορίθμων και καθώς διαφέρουν ως προς τις διαστάσεις και τα επίπεδα δυσκολίας τους ,μας βοηθούν να καθορίσουμε την απόδοση και τα μειονεκτήματα του αλγόριθμου.Επιπλέον οδηγούν στη λήψη ασφαλών συμπερασμάτων για τη βελτίωση του αλγόριθμου και σε ποιά σημεία του θα πρέπει να εφαρμοστούν.Στην βιβλιογραφία αρκετοί ερευνητές έχουν

μοντελοποιήσει διάφορα benchmark προβλήματα (Fisher and Thompson, 1963; Lawrence 1984; Adam et al., 1988, Applegate and Cook, 1991; Storer et al., 1992; Yamada and Nakano, 1992; Taillard, 1993; Demirkol et al., 1998).
Αξίζει να σημειωθεί οτι η πλειοψηφία των benchmarks στη βιβλιογραφία χρησιμοποιεί ακέραιες τιμές των χρόνων επεξεργασίας των εργασιων ,από ένα σχετικά μικρό εύρος ,ενώ σε περιβάλλοντα παραγωγής του πραγματικού κόσμου οι τιμές αυτές δεν είναι ακέραιοι αριθμοί ,ούτε επιλέγονται από καθορισμένα intervals.Αν και αυτό το γεγονός δημιούργησε αμφιβολίες για την πρακτική χρησιμότητά τους, η ανάλυση των Amar and Gupta (1986),οι οποίοι υπολόγισαν τον χρόνο της CPU και των αριθμό των επαναλήψεων που καταναλώθηκαν , κατα την εφαρμογή προσεγγιστικών και αλγόριθμων βελτιστοποίησης, σε δεδομένα τόσο του πραγματικού κόσμου , όσο και σε προσομειωμένα δεδομένα ,έδειξε οτι τα προσομειωμένα προβλήματα χρονοπρογραμματισμού είναι πιο δύσκολο να λυθούν από αυτά του πραγματικού κόσμου ,ανεξάρτητα απο τον αλγόριθμο που χρησιμοποιήθηκε.
Οι Matsuo et al. (1988) and Taillard (1989, 1994) έδειξαν ότι η δυσκολία των JSP's έχει την τάση να μειώνεται καθώς η αναλογία των εργασιών προς τον αριθμό των μηχανών αυξάνεται (τουλάχιστον 4 φορές μεγαλύτερη).Οι Ramudhin and Marier (1996) παρατήρησαν επίσης οτι όταν $n > m$ η πιθανότητα να παγιδευτεί ο αλγόριθμος σε τοπικό ελάχιστο είναι μειωμένη ,ενώ το πρόβλημα απλοποιείται ακόμα περισσότερο για μικρό αριθμό μηχανών (Taillard, 1994; Adam et al., 1988). Ο Taillard, (1994) κατάφερε να παράξει βέλτιστες λύσεις σε πολυωνυμικό χρόνο για στιγμιότυπα με 1,000,000 διεργασίες αρκεί να χρησιμοποιούνται το πολύ 10 μηχανές ,δηλαδή ο λόγος εργασιών/μηχανών να είναι το πολύ 100,000:1, ενώ τέλος είναι αξιοσημείωτο ότι σε αρκετά 'ευκολα ' στιγμιότυπα ,τα διάφορα τοπικά ελάχιστα τους ισοδυναμούν με το ολικό βέλτιστο.
Ωστόσο με την αύξηση του μεγέθους του προβλήματος και καθώς οι διαστάσεις του τείνουν να γίνουν τετραγωνικές ($n \rightarrow m$) ,γίνεται ιδιαίτερα δύσκολο να βρεθεί και η βέλτιστη λύση.Για παράδειγμα είναι εμφανής η διαφοροποίηση σε εύκολα και δύσκολα προβλήματα στα στιγμιότυπα του Lawrence , (LA) (Lawrence, 1984).Ο Adam et al. (1988) έλυσε βέλτιστα τα LA11-15 (20×5) και LA31-35 (30×10) χρησιμοποιώντας μια από τις πρώιμες ευρετικές μεθόδους – επιπλέον οι Caseau and Laburthe (1995) έδειξαν ότι η βελτιστότητα για τα παραπάνω στιγμιότυπα μπορεί να επιτευχθεί εύκολα ,ενώ για τα LA21 (15×10) και LA36-40 (15×15), απαιτείται μεγαλύτερο υπολογιστικό κόστος.Συνοψίζοντας ένα στιγμιότυπο του JSP θεωρείται 'δύσκολο' εάν έχει την ακόλουθη δομή: $l \geq 200$ όπου $n \geq 15$, $m \geq 10$, $n < 2.5m$, όπου l ο συνολικός αριθμός των διεργασιών, n και m το πλήθος των εργασιών και μηχανών αντίστοιχα.

3.4.1 Στιγμιότυπα των Fisher και Thompson

Είναι από τα πιο γνωστά και μελετημένα benchmarks, έχουν δημιουργηθεί από τους Fisher and Thompson (Fisher and Thompson, 1963) και είναι τα εξής : FT06 (6×6);FT10 (10×10); FT20 (20×5).Ενώ τα FT06 και FT20 λύθηκαν βέλτιστα το 1975 ,η βέλτιστη λύση για το FT10 παρέμεινε ανοιχτή εώς το 1987. Ο Florian et al. (1971) έδειξε ότι η δική του υλοποίηση του αλγόριθμου του Balas (1969) μπορούσε να επιτύχει τη βέλτιστη λύση (makespan 55) για το FT06.Για το FT20 από την άλλη (με βέλτιστο makespan 1165) χρειάστηκαν 12 χρονια για να λυθεί βέλτιστα (McMahon and Florian, 1975).Όσον αφορά το περιβόητο FT10 στιγμιότυπο(Πίνακας 3.1) ,το οποίο αναδεικνύει και την δυσκολία στη λύση του JSP,πάρα τον τεράστιο υπολογιστικό κόπο και την σταθερή πρόοδο των ερευνητών στο πεδίο ανα τον χρόνο ,η βέλτιστη τιμή για το makespan του (930) αποδείχτηκε ύστερα απο 26 χρόνια (Carlier and Pinson, 1989).Ένας από τους βασικούς λόγους της δυσκολίας του είναι

το μεγάλο χάσμα ανάμεσα στο κάτω φράγμα (LB ,808) και τη βέλτιστη τιμή του makespan.Οι Pesch and Tetzlaff (1996) επεσήμαναν οτι υπάρχει μόνο μια κρίσιμη ακμή που να συνδέει τις διεργασίες 13 και 66 ,η οποία αν οριοθετηθεί με λάθος τρόπο αποκλείει την εύρεση του βέλτιστου .Το καλύτερο makespan που μπορεί να επιτευχθεί σε αυτή την περίπτωση,όταν δηλαδή η διεργασάα 13 προηγείται της 66 στη σειρά επεξεργασίας ,ακόμα και εάν οι υπόλοιπες διεργασίες έχουν χρονοπρογραμματιστεί σωστά είναι 937.Επιπλέον οι Pesch and Tetzlaff ανέδειξαν την σημασία αυτής της ακμής δείχνοντας ότι εάν η συγκεκριμένη διάζευξη είναι στη σωστή κατεύθυνση εκ των προτέρων μια εφαρμογή του αλγόριθμου του Brucker et al. (1994) στο FT10 δίνει το ολικό βέλτιστο σε 448 secs σε έναν PC386 , διαφορετικά χρειάζονται 1138 secs.Τέλος ο Lawler et al. (1993) αναφέρει ότι η εφαρμογή ντετερμινιστικής τοπικής αναζήτησης στο FT10 σε διάρκεια 6000 secs παράγει περισσότερα από 9000 τοπικά βέλτιστα με το καλύτερο makespan να είναι 1006,αναδεικνύοντας έτσι ακόμα περισσότερο την δυσκολία του συγκεκριμένου προβλήματος.

Operation number	JOB																			
	J_1		J_2		J_3		J_4		J_5		J_6		J_7		J_8		J_9		J_{10}	
	$\mathcal{M}_i$	τ_i	$\mathcal{M}_i$	τ_i	$\mathcal{M}_i$	τ_i	$\mathcal{M}_i$	τ_i	$\mathcal{M}_i$	τ_i	$\mathcal{M}_i$	τ_i	$\mathcal{M}_i$	τ_i	$\mathcal{M}_i$	τ_i	$\mathcal{M}_i$	τ_i	$\mathcal{M}_i$	τ_i
1–10	1	29	1	43	2	91	2	81	3	14	3	84	2	46	3	31	1	76	2	85
11–20	2	78	3	90	1	85	3	95	1	6	2	2	1	37	1	86	2	69	1	13
21–30	3	9	5	75	4	39	1	71	2	22	6	52	4	61	2	46	4	76	3	61
31–40	4	36	10	11	3	74	5	99	6	61	4	95	3	13	6	74	6	51	7	7
41–50	5	49	4	69	9	90	7	9	4	26	9	48	7	32	5	32	3	85	9	64
51–60	6	11	2	28	6	10	9	52	5	69	10	72	6	21	7	88	10	11	10	76
61–70	7	62	7	46	8	12	8	85	9	21	1	47	10	32	9	19	7	40	6	47
71–80	8	56	6	46	7	89	4	98	8	49	7	65	9	89	10	48	8	89	4	52
81–90	9	44	8	72	10	45	10	22	10	72	5	6	8	30	8	36	5	26	5	90
91–100	10	21	9	30	5	33	6	43	7	53	8	25	5	55	4	79	9	74	8	45

Πίνακας 3.1: Το 10x10 πρόβλημα των Fisher και Thompson(FT10).Ένα πρόβλημα που απασχόλησε τους ερευνητές για περισσότερο απο 25 χρόνια.

3.4.2 Στιγμιότυπα του Lawrence

Τα benchmarks (LA) που πρότεινε ο Lawrence(1984) περιλαμβάνουν 40 στιγμιότυπα 8 διαφορετικών διαστάσεων : 10×5, 15×5, 20×5, 10×10, 15×10, 20×10, 30×10, 15×15. Λόγω της μεγάλης ποικιλίας διαφορετικών διαστάσεων και του επαρκούς συνδυασμού ευκόλων και δύσκολων στιγμιοτύπων ,το σύνολο αυτών των προβλημάτων έχει χρησιμοποιηθεί κατεξακολούθηση από μια μεγάλη μερίδα ερευνητών.Οι Applegate and Cook (1991) κατονόμασαν τέσσερα LA στιγμιότυπα τα οποία δεν κατάφεραν να λύσουν- τα LA (21, 27, 29, 38) - και τα οποία είναι αρκετά δυσκολότερα από το FT10 ,ενώ μέχρι πρόσφατα ήταν άγνωστες και οι βέλτιστες λύσεις τους ,παρότι είχε εφαρμοστεί πάνω τους κάθε αλγόριθμος.Επιπλέον ο Vaessen et al. (1996) υπέδειξε επίσης ως δύσκολα στιγμιότυπα τα LA (24, 25, 40), τα 15×15 (36, 37, 39), δύο μικρότερα LA (2,19) και το FT10 τα οποία χρησιμοποιεί ως μέτρα σύγκρισης της απόδοσης διαφόρων αλγόριθμων.Αυτά λοιπόν τα 13 στιγμιότυπα που αναφέραμε προηγουμένως αποτελούν ένα κατάλληλο και αποδοτικό συγκριτικό μοντέλο για την μελέτη νέων προτεινόμενων αλγορίθμων από τους διάφορους ερευνητές.Παραθέτουμε τους αλγοριθμους που έχουν εφαρμοστεί από αυτούς τους ερευνητές στον Πίνακα 3.2 .

Πίνακας 3.2:Περίληψη των αλγορίθμων που έχουν εφαρμοστεί στα LA & FT στιγμιότυπα(Siang Lyn,2005).

Algorithms	Researchers
Shifting Bottleneck Heuristics	Adams et al., 1988 Balas and Vazacopoulos, 1998 Balas et al., 1995 Applegate and Cook, 1991
Threshold Algorithms (Threshold Accepting & Simulated Annealing)	Matsuo et al., 1988 Applegate and Cook, 1991 Van Laarhoven et al., 1992 Aarts et al., 1994
Tabu Search	Dell Amico and Trubian, 1993 Barnes and Chambers, 1995 Nowicki and Smutnicki, 1996
Genetic Algorithms	Aarts et al., 1994 Della Croce et al., 1995 Dorndorf and Pesch, 1995
Greedy Randomised Adaptive Search Procedures	Binato et al., 2002
Priority Rules Heuristics	Jain et al., 1997

3.5 Τοπική Αναζήτηση

Η τοπική αναζήτηση υιοθετεί την ιδέα ότι βάση μιας δοσμένης λύσης ,αυτή μπορεί να βελτιωθεί επαναληπτικά κάνοντας μικρές αλλαγές ,παράγοντας νέες λύσεις ξανα και ξανά.Άρα θα μπορούσαμε να εφαρμόσουμε μια μέθοδο τοπικής αναζήτησης (daemon action) προκειμένου να ενισχύσουμε τα επίπεδα της φερομόνης στις ακμές του διαζευκτικού γράφου που αντιστοιχούν σε καλύτερες λύσεις και να βελτιώσουμε το αποτέλεσμα της αναζήτησής μας. Αν και δεν αποτελεί σημείο έρευνας της συγκεκριμένης διπλωματικής θα παραθέσουμε τις βασικες αρχές των μεθόδων αυτών κάνοντας μια σύντομη αναδρομή στην πρόοδο που έχει επιτευχθεί στον τομέα . Σημειώνουμε επίσης ότι οι πλειοψηφία των μελετών στη βιβλιογραφία χρησιμοποιεί ως αντικειμενική συνάρτηση το makespan στο οποίο θα αναφερόμαστε ως αντικειμενικό κριτηρίο για το υπόλοιπο της ενότητας.

Έστω ένα συνολο F εφικτών λύσεων.Μπορούμε να ορίσουμε δύο συναρτήσεις πάνω στο F.Την αντικειμενική συνάρτηση $c{:}F\rightarrow R$ όπως την έχουμε ορίσει παραπάνω και την γειτονική συνάρτηση $N{:}\ F\rightarrow 2^F$ η οποία ορίζει για κάθε λύση x ,$x \in F$ μια γειτονιά λύσεων $N(x)\subseteq F$. Κάθε λύση στο $N(x)$ καλείται γείτονας του x.Μια λύση $x \in F$ αποτελεί ένα τοπικό ελάχιστο ως προς την $N(x)$ εάν $c(x)\leq c(y)$,$\forall\ y \in N(x)$. Ο βασικός αλγόριθμος για την εύρεση ενός τοπικού ελάχιστου ,εκκινεί την αναζήτησή του στην γειτονιά μιας αρχικής εφικτής λύσης με σκοπό την εύρεση μιας λύσης μικρότερου κόστους.Εάν βρεθεί ο αλγόριθμος συνεχίζει με βάση αυτή τη λύση, διαφορετικά ένα τοπικό ελάχιστο έχει βρεθεί .Η ποιότητα του τοπικού ελάχιστου εξαρτάται απο δυο παραγοντες , δηλαδη την ποιοτητα της αρχικης λυσης και τον ορισμο της γειτονικης συναρτησης σε συνδυασμο με τον τροπο που θα γινεται η

αναζήτηση στον χώρο των γειτονικών λύσεων.Προκειμένου να ορίσουμε μια γειτονική συνάρτηση παραθέτουμε καποιους βασικές έννοιες(Vaessens & Lenstra ,1994). :

*Ακολουθώντας τον συμβολισμό της Ενότητα 3.1 δύο διεργασίες i και j καλούνται διαδοχικές εάν ισχύει $t_i+\tau_i=t_j$. Ένα **block** είναι μια μέγιστη ακολουθία από διαδοχικές διεργασίες μεγέθους τουλάχιστον 1 οι οποίες δρομολογούνται στην ίδια μηχανή και ανήκουν στο κρίσιμο μονοπάτι.Η διεργασία ενός block καλείται **εσωτερική**(internal), εάν δεν είναι ούτε η πρώτη ούτε η τελευταία διεργασία στο block.Ανακαλώντας λοιπόν την έννοια ενός πλήρως κατευθυνόμενου άκυκλου γράφου $G(D)$ που αντιστοιχεί σε ένα εφικτό χρονοπρόγραμμα- θυμίζουμε οτι το D περιέχει το σύνολο των ακμών που συνδέουν πλήρως τους κόμβους του G εκτός από τους κόμβους που ανήκουν στην ίδια εργασία - οι περισσότερες γειτονικές συναρτήσεις που έχουν προταθεί στην βιβλιογραφία βασίζονται στην κατασκευή ενός νέου G(D") το οποίο θα αντιστοιχεί στην γείτονα λύση.Οι παρακάτω ιδιότητες είναι ιδιαίτερα χρήσιμες στην εξαγωγή αποδοτικών γειτονικών συναρτήσεων*

1. *Δεδομένου ενός εφικτού κατευθυνόμενου γράφου,η αντιστροφή μιας διαζευκτικής ακμής δεν αλλάζει την εφικτότητα του χρονοπρογράμματος.*
2. *Δεδομένου ενός εφικτού κατευθυνόμενου γράφου $G(D)$ η αντιστροφή μιας διαζευκτικής ακμής (i,j) ανάμεσα σε δύο εσωτερικές διεργασίες ενός block έχει ως αποτέλεσμα ένα εφικτό χρονοπρόγραμμα με makespan τουλάχιστον όσο αυτό του $G(D)$.*
3. *Δεδομένου ενός εφικτού κατευθυνόμενου γράφου $G(D)$, έστω i,j οι πρώτες δύο διεργασίες του πρώτου block του κρίσιμου μονοπατιου και η j διεργασία είναι εσωτερική διεργασία.Αντιστρέφοντας την ακμή (i,j) το παραγόμενο χρονοπρόγραμμα G(D") θα έχει τουλάχιστον το ίδιο makespan με το $G(D)$.Το ίδιο ίσχυει εάν οι i,j είναι οι τελευταίες διεργασίες στο τελευταίο block του κρίσιμου μονοπατιού και η i είναι εσωτερική.*

*Με βάση τις παραπάνω ιδιότητες οι πιό απλές γειτονικές συναρτήσεις βασίζονται στην αντιστροφή ακριβώς μιας ακμής από το σύνολο D. Ο **P.J.M van Laarhoven** πρότεινε μια γειτονική συνάρτηση N_1 η οποία παράγει μια γείτονα λύση αντιμεταθέτοντας δύο οποιεσδήποτε διεργασίες μέσα σε ένα block. Ωστόσο το μέγεθος της N_1 γειτονιάς παραμένει ιδιαίτερα μεγάλο και περιλαμβάνει ένα μεγάλο αριθμό μη αποδοτικών αντιμεταθέσεων(Vaessens & Lenstra ,1994).Παραυτά η μελέτη αυτή έθεσε τα θεμέλια για την δημιουργία πιό αποδοτικών μεθόδων τοπικής αναζήτησης , όσον αφορά τον χρονοπρογραμματισμό εργασιών, εφόσον περιόρισε το σύνολο των πιθανών μεταθέσεων μόνο ανάμεσα σε κρίσιμες διεργασίες.Τα πλεονεκτήματα είναι αρκετά καθώς οι παραγόμενες λύσεις 1) θα είναι πάντα εφικτές , 2) μόνο οι μεταθέσεις ανάμεσα σε κρίσιμες διεργασίες μπορούν να οδηγήσουν σε μικρότερο κρίσιμο μονοπάτι και κατεπέκταση σε μικρότερο C_{max} .Αντιμεταθέτοντας δηλαδή μη-κρίσιμες διαδοχικές διεργασίες το κρίσιμο μονοπάτι δεν γίνεται να μικρύνει ,εφόσον δεν επηρεάζεται και επιπλέον θα υπάρχει ήδη στην παραγόμενη λύση.Τέλος για οποιαδήποτε αρχική εφικτή λύση θα υπάρχει πάντα πεπερασμένος αριθμός κινήσεων που να μπορεί να οδηγήσει σε μια ολικά βέλτιστη λύση. Η ιδιότητα αυτή αναφέρεται και ως συνδεκτικότητα (**connectivit**y),Zapfel et al(2010)).*

*Ο **Matsuo et al. (1988)** κατάφερε να βελτιώσει τον αριθμό των κινήσεων δείχνοντας οτι εάν ο προκατόχος της εργασίας της i , JP(i) ή ο διαδόχος της εργασίας της j ,JS(j), δεν ανήκουν στο κρίσιμο μονοπάτι τότε η αντιμετάθεση των i,j δεν μπορεί να ελλατώσει το makespan-πιο απλά αντιμεταθέτοντας δυο εσωτερικές*

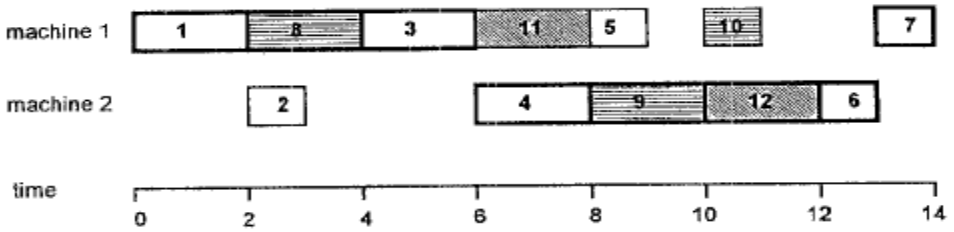
διεργασίες σε ένα block δεν οδηγεί σε κάποια βελτίωση.Ένας ακόμα πιο ενδιαφέρον ορισμός και ίσως ο πιο δημοφιλής στο πεδίο ,είναι αυτός των ***Nowicki and Smutnicki(1996)*** *οι οποίοι περιόρισαν ακόμα περισσότερο τον αριθμό των κινήσεων δείχνοντας ότι η αντιμετάθεση των πρώτων(τελευταιών) δύο διεργασιών στο πρώτο(τελευταίο) block δεν οδηγεί σε κάποια βελτίωση.Πιο συγκεκριμένα οι συγγραφείς ορίζουν ως π_k μια συγκεκριμένη σειρά επεξεργασίας των m_k διεργασιών που δρομολογούνται στη μηχανή $k \in M$ από ένα σύνολο M μηχανών.Έχουμε δηλαδή $\pi_k=(\pi_k(1),\ldots,\pi_k(m_k))$ $,k \in M$, $\pi_k(i) \in O$.Εάν θεωρήσουμε λοιπόν ως Π_k όλες τις πιθανές αντιμεταθέσεις ανάμεσα στις διεργασίες στην μηχανή $k \in M$,μια πλήρης σειρά επεξεργασίας π όλων των διεργασιών σε όλες τις μηχανές θα είναι στοιχείο του καρτεσιανού γινομένου Π όλων των πιθανών αντιμεταθέσεων σε όλες τις μηχανές: $\pi \in \Pi = \Pi_1 \times \Pi_2 \times \ldots \times \Pi_m$. Πλέον μπορούμε να συμβολίσουμε τον κατευθυνόμενο γράφο που περιγράφει μια συγκεκριμένη σειρά επεξεργασίας π ως $G(\pi)=(O,R \cup E(\pi))$,όπου R και $E(\pi)$ το σύνολο των κατευθυνόμενων και διαζευκτικών ακμών αντίστοιχα.Επιστρέφοντας στον ορισμό μιας συναρτήσης γειτονιάς οι συγγραφείς ορίζουν ένα κρίσιμο μονοπάτι u μήκους w διεργασιών, $u=(u_1, u_2,\ldots, u_w)$, $u_i \in O$. Κατόπιν αποσυνθέτουν το κρίσιμο μονοπάτι σε ένα ή περισσότερα blocks –τυπικά σε ένα κρίσιμο μονοπάτι αποτελούμενο από r blocks ορίζουν ένα block B_j ως μια ακολουθία απο διεργασίες : $B_j =(u_{aj}, u_{aj+1},\ldots, u_{bj})$, $j=1,\ldots r$.Για να διασφαλίσουμε ότι τα blocks αποτελούν από κοινού το κρίσιμο μονοπάτι απαιτούμε οι δείκτες των διεργασιών να αυξάνονται σε ένα block, όπως επίσης και ανάμεσα σε διαφορετικά blocks :*

$$1 = \alpha_1 \le b_1 < b_1+1= \alpha_2 \le b_2 \ldots \le \alpha_r \le b_r = w. \qquad (3.11)$$

Ακόμα οι διεργασίες ενός block θα πρέπει να δρομολογούνται στην ίδια μηχανή: $\mu(u_{aj})= \mu(u_{aj+1}) = \ldots = \mu(u_{bj})$, $j=1,\ldots,r$,ενώ δυο διαδοχικά blocks επεξεργάζονται σε διαφορετικες μηχανές : $\mu(\alpha_j) \neq \mu(\alpha_{j+1})$, $j=1,\ldots,r-1$. Για την καλύτερη κατανόηση των παραπάνω παραθέτουμε το παρακάτω παράδειγμα.

Παράδειγμα : Έστω 3 εργασιες των 12 διεργασιών συνολικά που πρέπει να επεξεργαστούν σε 2 μηχανές ,δηλαδή θα έχουμε n=3,m=2,o=12 .
Η εργασια J1 αποτελείται από την ακολουθία διεργασιών (1,2,3,4,5,6,7) η J2 απο την (8,9,10) και η J3 απο την (11,12).Οι διεργασίες δρομολογούνται στη μηχανή $\mu_1= \mu_3 = \mu_5= \mu_7= \mu_8= \mu_{10}= \mu_{11}=1$ και $\mu_2= \mu_4 = \mu_6= \mu_9= \mu_{12}=2$ αντίστοιχα με χρόνους επεξεργασίας $\tau_1= \tau_3= \tau_4= \tau_8= \tau_9= \tau_{11}= \tau_{12}=2$ και $\tau_2= \tau_5= \tau_6= \tau_7= \tau_{10}=1$.
Έστω επίσης μια εφικτή σειρά επεξεργασίας όλων διεργασιών $\pi =(\pi_1, \pi_2)$ στις μηχανές 1 και 2,όπου $\pi_1= (1,8,3,11,5,10,7)$ και $\pi_2= (2,4,9,12,6)$.Τότε ο γράφος $G(\pi)$ περιέχει ένα πιθανό κρίσιμο μονοπάτι u=(1,8,3,4,9,12,6,7) , μήκους w=8 διεργασιών, το οποίο μπορούμε να διακρίνουμε σε r =3 blocks ,όπως φαίνεται στο διάγραμμα Gantt της Εικόνας 3.8 ,με $B_1=(1,8,3)$, $B_2=(4,9,12,6)$, $B_3=(7)$ και $\mu(B_1)= \mu(B_3)=1$, $\mu(B_2)=2$, $\alpha_1=1$, $b_1=3$, $\alpha_2=4$, $b_2=7$, $\alpha_3=8$, $b_3=8$.

Εικόνα 3.8: Διάγραμμα Gantt.Διακρίνουμε σε παχύ περίγραμμα τα blocks $B_1=(1,8,3)$ στην μηχανή 1, $B_2=(4,9,12,6)$ στη μηχανή 2 και $B_3=(7)$ στη μηχανή 1(Nowiki &Smutnicki,1996).



Όπως προαναφέραμε η επιτυχία της γειτονιάς των (Nowicki & Smutnicki) βασίζεται στον δραματικό περιορισμό του αριθμού των κινήσεων από το σχήμα που πρότεινε ο Van Laarhoven et al.(1992) για το οποίο είναι γνωστό a-priori ότι δεν οδηγεί σε άμμεση βελτίωση του $C_{max}(\pi)$.Πιό συγκεριμένα αντιμεταθέτουν τις πρώτες δυο(τελευταίες δυο) διεργασίες σε κάθε block $B_2,\ldots, B_{r-1}$ που περιέχει τουλάχιστον δυο διεργασίες (Εξ. 3.17).Για το πρώτο block B_1 αντιμεταθέτουν μόνο τις τελευταίες δυο διεργασίες (Εξ. 3.16) και κατα αντιστοιχία για το τελευταίο block B_r μόνο τις πρώτες δυο (Εξ. 3.18).Μπορούμε να ορίσουμε δηλαδή το σύνολο των κινήσεων αρχίζοντας από το π ως $V(\pi)=$,όπου

$V_1(\pi) =$ - *(3.12)*

$V_j(\pi) =$ - *(3.13)*

$V_r(\pi) =$ *(3.14)*

Το σύνολο των κινήσεων είναι μη κενό ,εάν υπάρχουν τουλάχιστον δύο block(r>1) και εάν υπάρχει τουλάχιστον ένα block με αριθμό διεργασιών μεγαλύτερο του δύο.Έστω τώρα $Q(\pi,\upsilon) \in \Pi$ η σειρά επεξεργασίας όλων των διεργασιών σε όλες τις μηχανές μετά από εφαρμογή της κίνησης υ στο π .Τότε την γειτονιά $H(\pi)$ του π ορίζουν το σύνολο των πιθανών χρονοπρογραμμάτων που προκύπτουν από την εφαρμογή όλων των κινήσεων που ανήκουν στο $V(\pi)$, δηλαδή $H(\pi)=\{ Q(\pi,\upsilon) : \upsilon \in V(\pi) \}$. Σύμφωνα με το παράδειγμά μας θα έχουμε μια κίνηση (8,3) στο block B_1 ,δυο κινήσεις (4,9) , (12,6) στο block B_2 ,και καμία κίνηση στο τελευταίο block,δηλαδή θα είναι $V_1(\pi)=\{ (8,3) \}$, $V_2(\pi)=\{ (4,9) ,(12,6) \}$, $V_3(\pi)=$ και $V(\pi)= \{ (8,3) ,(4,9) ,(12,6) \}$. Τέλος η παραπάνω κινήσεις θα μας δώσουν τα νέα χρονοπρογράμματα $Q(\pi, (8,3))=((1,\mathbf{3,8},11,5,10,7), (2,4,9,12,6))$, $Q(\pi, (4,9))=((1,8,3,11,5,10,7), (2,\mathbf{9,4},12,6))$ και $Q(\pi, (12,6))= ((1,8,3,11,5,10,7), (2,4,9,\mathbf{6,12}))$ που θα ανήκουν στο $H(\pi)$.

Συνοψίζοντας, αντίθετα με τους υπόλοιπους ορισμούς γειτονικων συναρτησεων οι Nowicki & Smutnicki επιλέγουν αυθαίρετα μόνο ένα κρίσιμο μονοπάτι και αγνοούν όλα τα υπόλοιπα με αποτέλεσμα να μειώνεται σημαντικά το υπολογιστικό κόστος,ενώ τέλος αποδεικνύουν ότι η γειτονιά τους περιέχει εφικτές λύσεις που είναι υποσύνολο

της N_1 γειτονιάς και ακόμα οτι οποιοδήποτε χρονοπρόγραμμα στο N_1 που δεν ανήκει στην γειτονιά τους δεν βελτιώνει το makespan(Pedersen,2007).Τέλος αξίζει να σημειώσουμε ότι η γειτονιά των Nowicki & Smutnicki δεν διατηρεί την ιδιότητα της συνδεκτικότητας.

3.6 Πολυκριτηριακή βελτιστοποίηση

Στον παρόν κεφάλαιο γίνεται μια προσπάθεια προσέγγισης του πολυκριτηριακού JSP.Αρχικά γίνεται μια σύντομη ανασκόπηση των εννοιών και του σκοπού της ολικής βελτιστοποίησης ,ώστε να εισάγουμε τον αναγνώστη στις αρχές της πολυκριτηριακής βελτιστοποίησης.Στη συνέχεια παρουσιάζονται οι βασικές έννοιες της πολυκριτηριακής βελτιστοποίησης,οι διάφορες τεχνικές βελτιστοποίησεις, βελτιστότητα κατα Pareto κλπ.Παρουσιάζουμε μια ποικιλία ποιοτικών μέτρων αξιολόγησης της λύσης ενός πολυκριτηριακού προβλήματος και ακολούθως αναλύουμε όλες τις πτυχές ενός ACO αλγορίθμου για την βελτιστοποίηση του πολυκριτηριακού προβλήματος.

3.6.1ΟΛΙΚΗ ΒΕΛΤΙΣΤΟΠΟΙΗΣΗ

Η ολική βελτιστοποίηση είναι ο κλάδος των εφαρμοσμένων μαθηματικών και της αριθμητικής ανάλυσης που ασχολείται κυρίως με την βελτιστοποίηση.Ο στόχος της ολικής βελτιστοποίησης είναι να βρεί τα καλύτερα πιθανά στοιχεία x^* από ένα σύνολο X ,σύμφωνα με ένα σύνολο κριτηρίων $F=\{ f_1, f_2, \ldots, f_n \}$. Αυτά τα κριτήρια εκφράζονται συνήθως ως μαθηματικές συναρτήσεις που τις ονομάζουμε αντικειμενικές συναρτήσεις(Weisse,2009).

***Ορισμός** :Μία αντικειμενική συνάρτηση $f : X \rightarrow Y$ με $Y \subseteq \mathrm{R}$ είναι μια μαθηματική μαθηματική συνάρτηση που υπόκειται σε βελτιστοποίηση.Η περιοχή Y μιας αντικειμενικής συνάρτησης όπως επίσης και το εύρος της πρέπει να είναι υποσύνολο των πραγματικών αριθμών.Η περιοχή X της f καλείται χώρος αναζήτησης του προβλήματος και μπορεί να αντιπροσωπεύει οποιοδήποτε τύπου στοιχεία όπως αριθμούς ,σχέδια κατασκευής κλπ.Οι αντικειμενικές συναρτήσεις δεν είναι απαραίτητα μαθηματικές εκφράσεις ,αλλά μπορεί να είναι πολύπλοκοι αριθμοί που περιλαμβάνουν πολλλαπλές προσομειώσεις.Η ολική βελτιστοποίηση λοιπόν περιλαμβάνει όλες εκείνες τις τεχνικές που μπορούν να χρησιμοποιηθούν για να βρούν τα καλύτερα στοιχεία x^* που ανήκουν στο X και παράλληλα ικανοποιούν όλα τα κριτήρια της $f \in F$.*

3.6.1.1 Αντικειμενικές συναρτήσεις ενός κριτηρίου(Single Objective Functions)

Στην περίπτωση βελτιστοποίησης ενός κριτηρίου f (πχ. το makespan ή η συνολική βραδύτερη περάτωση όπως εξετάσαμε στο προηγούμενο κεφάλαιο) το βέλτιστο είναι είτε μέγιστο είτε ελάχιστο αναλόγως με το τι ψάχνουμε να ελαχιστοποιήσουμε,αν και συνήθως τα προβλήματα βελτιστοποίησης ορίζονται ως προβλήματα ελαχιστοποίησης και στην περίπτωση που θέλουμε να μεγιστοποιήσουμε απλά ελαχιστοποιούμε το αντίστροφο του κριτηρίου $f \rightarrow (-f)$.Στην Εικόνα 3.9 φαίνεται μια συνάρτηση f ορισμένη πάνω σε ένα δισδιάστατο χώρο $X =(X_1, X_2)$.Είναι εμφανής η διάκριση ανάμεσα στα τοπικά και ολικά βέλτιστα.Το ολικό βέλτιστο είναι βελτιστο συνολικά στον χώρο X ,ενώ το τοπικό βέλτιστο μόνο σε ένα υποσύνολο του X.

***Ορισμός :Τοπικό ελάχιστο** $\in X$ μιας αντικειμενικής συνάρτησης $f : X \rightarrow R$*

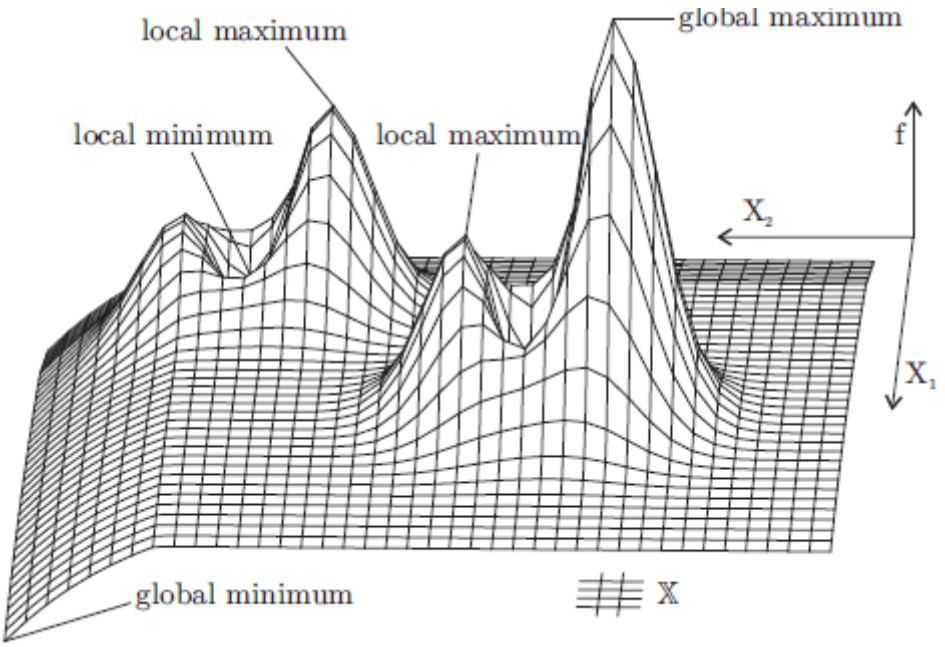
είναι μια μεταβλητή εισόδου με $f(\quad) \leq f(x)$ *για όλα τα* x *που βρίσκονται στην γειτονιά του , για την οποία δηλαδή ισχύει:*

$$\forall \quad >0 : f(\quad) \leq f(x)\ \forall\ x \in X\ ,\ |\ x - \quad| < \quad , \qquad (3.15)$$

Αντίστοιχα ορίζουμε και το τοπικό μέγιστο .

Ορισμός :Τοπικό βέλτιστο $x_l^* \in X$ *μιας αντικειμενικής συνάρτησης* $f : X \longrightarrow R$ *είναι είτε ένα τοπικό μέγιστο είτε ένα τοπικό ελάχιστο.*

Ορισμός :Ολικό ελάχιστο $\in X$ *μιας αντικειμενικής συνάρτησης* $f : X \longrightarrow R$ *είναι μια μεταβλητή εισόδου για την οποία* $f(\quad) \leq f(x)\ \forall\ x \in X$ *.*



Εικόνα 3.9 :Ολικά και τοπικά βέλτιστα σε μια δισδιάστατη συνάρτηση(Weisse,2009)*.*

Αυτό που πρέπει να παρατηρήσουμε είναι ότι ακόμα και μια συνάρτηση μιας διάστασης $f : X = R \longrightarrow R$ *μπορεί να περιέχει περισσότερα από ένα ολικά μέγιστα, πολλαπλά ολικά ελάχιστα ή και συνδυασμό των παραπάνω.Για παράδειγμα μια συνάρτηση συνημιτόνου έχει ολικά μέγιστα* $= 2i\pi$ *και ολικά ελάχιστα* $= (2i+1)\pi$ *για όλα τα* $i \in Z$*. Τότε μια σωστή λύση ενός τέτοιου προβλήματος βελτιστοποίησης θα ήταν ένα σύνολο σημείων* $\boldsymbol{X}^*$ *που θα περιέχει όλες τις βέλτιστες λύσεις στο* X*,παρα ένα μοναδικό ελάχιστο ή μέγιστο, μια κοινή λογική που επικρατεί σήμερα στην πολυκριτηριακή βελτιστοποίηση.*

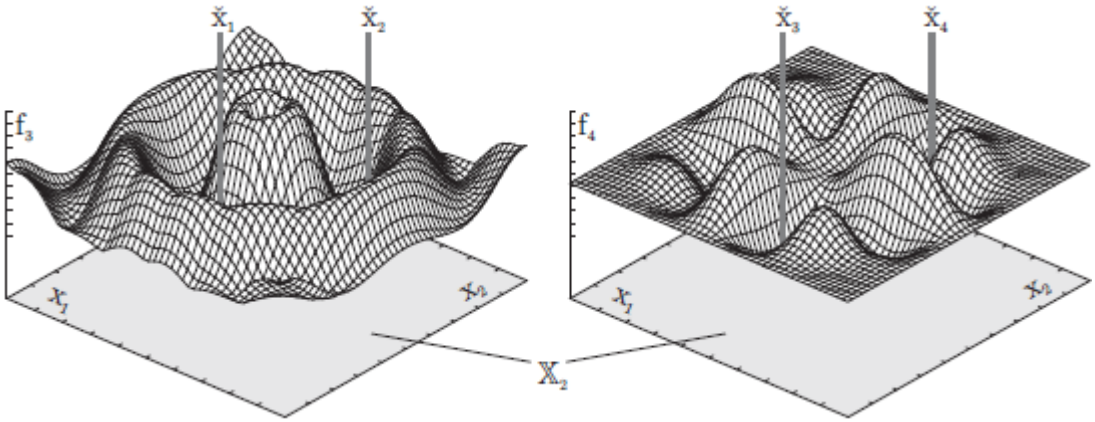
3.6.1.2 Αντικειμενικές Συναρτήσεις πολλαπλών κριτηρίων

Οι τεχνικές ολικής βελτιστοποίησης δεν χρησιμοποιούνται μόνο για την εύρεση του μέγιστου ή ελάχιστου μιας αντικειμενικής συνάρτησης f *αλλά εφαρμόζονται και σε*

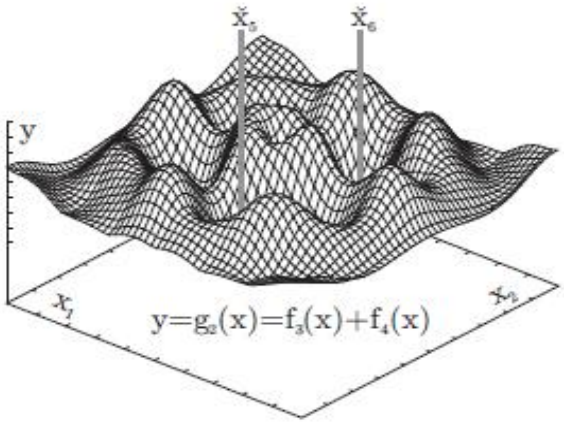
πολλά προβλήματα του πραγματικού κόσμου ή σε προβλήματα λήψης αποφάσεων για την βελτιστοποίηση ενός συνόλου F, που περιέχει $n=|F|$ αντικειμενικές συναρτήσεις f_i ,όπου η κάθε μια αναπαριστά ένα κριτήριο προς βελτιστοποίηση.

$$F=\{ f_i : X \rightarrow Y_i : 0 < i \leq n , Y_i \subseteq R\} \qquad (3.16)$$

Οι αλγόριθμοι που σχεδιάζονται για την βελτιστοποίηση τέτοιων συνόλων αντικειμενικών συναρτήσεων -συχνά ανταγωνιστικών μεταξύ τους- ονομάζονται ***πολυκριτηριακοί (Multi-objective)****. Σε αυτούς τους αλγόριθμους η εύρεση του ολικού βέλτιστου μπορεί να σημαίνει τη μεγιστοποίηση μιας συνάρτησης $f_i \in F$ και ελαχιστοποίηση μιας άλλης $f_j \in F$, $(i \neq j)$. Αρα σε αυτές τις περιπτώσεις δεν έχει νόημα η εύρεση ενός ολικού βέλτιστου(ελάχιστου ή μέγιστου) ,αλλά η εύρεση ενός συνόλου βέλτιστων στοιχείων $\boldsymbol{x}^* \in \boldsymbol{X}^* \subseteq X$. Ενα παράδειγμα φαίνεται στην Εικόνα 3.10 όπου οι αντικειμενικές συναρτήσεις f_3 και f_4 ελαχιστοποιούνται πάνω σε ένα χώρο δύο διαστάσεων $X_2 \subseteq R^2$ των πραγματικών αριθμών R . Από την Εικόνα 3.10 φαίνεται ότι κάθε συνάρτηση έχει δύο ολικά ελάχιστα ,πχ. η f_3 τα και ,ενώ η f_4 τα και με $\neq$ $\neq$ $\neq$.*



Εικόνα 3.10:Δύο συναρτήσεις f_3 και f_4 με διαφορετικά ελάχιστα , και , (Weisse,2009)..



Εικόνα 3.11: Βελτιστοποίηση με την μέθοδο του σταθμισμένου αθροίσματος(Weisse,2009).

Μέθοδος σταθμισμένων αθροισμάτων(Weighted Sums-Linear Aggregation)

Η πιό απλή μέθοδος για να ορίσουμε τι είναι βελτιστο είναι να υπολογίσουμε ένα σταθμισμένο άθροισμα $g(x)$ όλων των συναρτήσεων $f_i(x) \in F$.Κάθε αντικειμενική πολλαπλασιάζεται με ένα βάρος w_i το οποίο αντιπροσωπεύει την βαρύτητα της.Επίσης με τη χρήση προσημασμένων βαρών μπορούμε να ελαχιστοποιούμε μια αντικειμενική και να μεγιστοποιούμε μια άλλη. Για παράδειγμα μπορούμε να αναθέσουμε ένα βάρος $w_a=1$ σε μια αντικειμενική f_a και ένα βάρος $w_b=-1$ σε ένα άλλο κριτήριο f_b. Ελαχιστοποιώντας την $g(x)$ καταφέρνουμε να ελαχιστοποιούμε την πρώτη και να μεγιστοποιούμε την δεύτερη αντικειμενική συνάρτηση. Κατά αυτόν τον τρόπο τα πολυκριτηριακά προβλήματα μετατρέπονται σε προβλήματα ενός κριτηρίου.

$$\boldsymbol{x}^* \in \boldsymbol{X}^* \qquad g(\boldsymbol{x}^*) \geq g(x) \ \forall \ x \in X \qquad (3.18)$$

Ακολουθώντας το προηγούμενο παράδειγμα αναθέτουμε τα βάρη $w_3=1$ και $w_4=1$ στις αντικειμενικές f_3 και f_4 αντίστοιχα, οπότε και προκύπτει το άθροισμα g_2 το οποίο θέλουμε να ελαχιστοποιήσουμε.Τα δύο ολικά ελάχιστα που προκύπτουν είναι τα και όπως φαίνεται στην Εικόνα 3.11.

Μέθοδος προγραμματισμού στόχων (goal programming)

Αποτελεί επέκταση της μεθόδου των βαρών κατα την οποία ορίζονται τιμές-στόχοι T_i για κάθε κριτήριο i ,που εισάγονται στο πρόβλημα υπο μορφή συναρτήσεων ποινής.Η βαθμωτή συνάρτηση που διαμορφώνεται συνίσταται στην ελαχιστοποίηση της απόκλισης των κριτηρίων από τους αντίστοιχους στόχους,που στην γενική περίπτωση διαμορφώνεται ως:

Ως τιμή-στόχος κάθε κριτηρίου μπορεί να θεωρηθεί η ολικά βέλτιστη τιμή f_i^* κάθε επιμέρους κριτηρίου ,ενώ μεταβάλλοντας τις τιμές των στοχών T_i και των βαρών w_i προκύπτουν διαφορετικές λύσεις του προβλήματος που είναι Pareto βέλτιστες(Ευστρατιάδης & Κουτσογιάννης).

Βελτιστοποιήση Pareto

*Οι μαθηματικές αρχές για την πολυκριτηριακή βελτιστοποίηση με συγκρουόμενα κριτήρια δόθηκαν από τον οικονομολόγο Vilfredo Pareto.Η βελτιστότητα κατα Pareto αποτελεί σημείο αναφοράς σε εφαρμογές που αφορούν την οικονομία,την θεωρία παιγνίων, την μηχανική και τις κοινωνικές επιστήμες.Ορίζεται λοιπόν ένα **«μέτωπο»(frontier)** λύσεων το οποίο μπορεί να προκύψει από την «**ανταπόδωση»(trading-off)** συγκρουόμενων/ανταγωνιστικών αντικειμενικών κατα ένα βέλτιστο τρόπο(Weisse,2009).Ο «λήπτης αποφάσεων»(decision maker,είτε άνθρωπος είτε αλγόριθμος) μπορεί τότε χρησιμοποιώντας αυτό το μέτωπο να επιλέξει τη ρύθμιση που ικανοποιεί καλύτερα τον σκοπό του.Η έννοια του βέλτιστου κατα Pareto είναι στενά συνδεδεμένη με την έννοια της κυριαρχίας:*

Ορισμός(Κυριαρχία):*Ένα στοιχείο x_1 κυριαρχεί (προτιμάται απο) ένος στοιχείου x_2 , ($x_1 \vdash x_2$) εάν το x_1 είναι «καλύτερο» του x_2 σε τουλάχιστον μια αντικειμενική συνάρτηση και όχι «χειρότερο» στις υπόλοιπες. Βασιζόμενοι στο σύνολο F των συναρτήσεων f μπορούμε να γράψουμε:*

$$x_1 \vdash x_2 \qquad \forall i : 0 < i \leq n \rightarrow \omega_i f_i(x_1) \leq \omega_i f_i(x_2)$$
$$j : 0 < j \leq n : \omega_i f_i(x_1) < \omega_i f_i(x_2) \qquad (3.20)$$

–

Με διαφορετικό τρόπο από ότι τα βάρη στη προσέγγιση του σταθμισμένου αθροίσματος ,οι παράγοντες ω_i περιέχουν πληροφορία προσήμου που μας επιτρέπει να μεγιστοποιούμε κάποιες αντικειμενικές και να ελαχιστοποιούμε κάποια άλλα κριτήρια.

Ορισμός(Pareto Optimal): *Ένα στοιχείο $\boldsymbol{x}^* \in X$ είναι βέλτιστο κατα Pareto εάν δεν κυριαρχείται από κανένα άλλο στοιχείο του χώρου αναζήτησης X. Σύμφωνα με τους όρους βελτιστοποίησης κατα Pareto,το $\boldsymbol{X}^*$ καλείται σύνολο Pareto ή* ***Pareto μέτωπο****.*

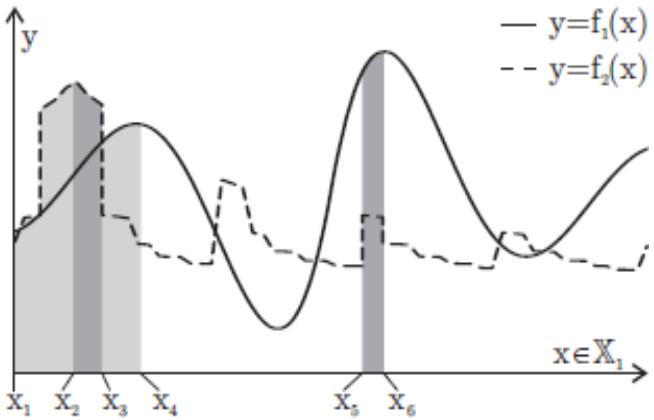
$$\boldsymbol{x}^* \in \boldsymbol{X}^* \qquad x \in X : x \vdash \boldsymbol{x}^* \qquad (3.22)$$

Ορισμος(Pareto Optimal Set): *Βέλτιστο σύνολο Pareto για ένα σύνολο αντικειμενικών συναρτήσεων πολυκριτηριακής βελτιστοποίησης ορίζεται ως:*

$$\boldsymbol{X}^* = \{ x^* \in X \sqcap x' \in X : f(x') \vdash f(x^*) \} \qquad (3.23)$$

Ένα απλό παράδειγμα παρουσιάζεται στην Εικόνα 3.12, όπου θεωρούμε δύο συναρτήσεις f_1 και f_2 τις οποίες θέλουμε να μεγιστοποιήσουμε ,άρα θέτουμε $\omega_1 = \omega_2 = -1$.Οι περιοχές με το σκούρο γκρι χρώμα αντιπροσωπεύουν το μέτωπο Pareto ,δηλαδή το βέλτιστο σύνολο $\boldsymbol{X}^* = [x_2, x_3]$ $[x_5, x_6]$ το οποίο στο συγκεκριμένο παράδειγμα περιέχει άπειρα σημεία. Όλα τα υπόλοιπα σημεία είναι κυριαρχούμενα,δηλαδή μη βέλτιστα.
Τα σημεία στην περιοχή ανάμεσα στα x_1 και x_2 (η σκιασμένη περιοχή με φωτεινό γκρι) κυριαρχούνται από άλλα σημεία στην ίδια περιοχή ή στην περιοχή $[x_2, x_3]$,εφόσον και οι δύο συναρτήσεις f_1 και f_2 μπορούν να βελτιωθούν αυξάνοντας το x. Ξεκινώντας από το πιό αριστερό σημείο του X το x_1 ,μπορούμε να κάνουμε ένα μικρό βήμα Δ προς τα δεξιά ,όπου συναντάμε το σημείο $x_1 + \Delta$ το οποίο κυριαρχεί επί του σημείου x_1 ,εφόσον $f_1(x_1 + \Delta) > f_1(x_1)$ και $f_2(x_2 + \Delta) < f_2 x_2)$. Επαναλαμβάνοντας αυτή τη διαδικασία θα βρίσκουμε πάντα νέα κυρίαρχα σημεία ωσότου φάασουμε στο σημείο x_2 ,το οποίο είναι και το ολικό μέγιστο της f_2 και κατεπέκταση δεν κυριαρχείται από κανένα άλλο σημείο X βάση της Σχέσης 5.8 . Στη συνέχεια η f_2 μειώνεται για λίγο ενώ η f_1 αυξάνεται. Εάν μετακινηθούμε κατά Δ δεξιά θα έχουμε $f_2(x_2 + \Delta) < f_2 x_2)$ και $f_1(x_2 + \Delta) > f_1(x_2)$. Η μια αντικειμενική θα μπορεί να αυξάνεται μόνο εάν μειώνεται η άλλη,άρα λοιπόν για να αυξήσουμε

την f_1 θα πρέπει να μειώνεται η f_2 και τα νέα σημεία θα είναι μη κυριαρχούμενα από το x_2. Κάποιες τιμές ωστόσο της f_2 στην περιοχή $x \in [x_1 , x_2)$ ίσως να είναι μεγαλύτερες από την $f_2 (x_2 + \Delta)$,όπου $f_1 (x_2 + \Delta) > f_1 (x)$. Αυτό σημαίνει ότι κανένα σημείο στην περιοχή $[x_1 , x_2)$ δεν κυριαρχεί κάποιου σημείου στο διάστημα $[x_2, x_4]$ καθώς η f_1 αυξάνεται ως το x_4. Στο σημείο x_4 ωστόσο η f_2 πέφτει απότομα πιο χαμηλά από το σημείο $f_2(x_5)$. Εφόσον οι τιμές της f_1 στην περιοχή $[x_5, x_6]$ είναι μεγαλύτερες από αυτές στη περιοχή $(x_3, x_4]$ όλα τα σημεία στην $[x_5, x_6]$ κυριαρχούν αυτών της $(x_3, x_4]$. Ανάλογα για τα σημεία στη λευκή περιοχή ανάμεσα στα x_4 και x_5 και μετά το x_6 κυριαρχούνται από τις μη κυριαρχούμενες περιοχές που συζητήσαμε παραπάνω.



Εικόνα 3.12 :Βελτιστοποίηση με την μέθοδο του Pareto μετώπου(Weisse,2009).

3.6.1.3 Χειρισμός περιορισμών

Σε αρκετά σενάρια βελτιστοποίησης , p περιορισμοί ανισότητας g και q περιορισμοί ισότητας h προστίθενται στις αντικειμενικές συναρτήσεις.Τότε μια υποψήφια λύση x είναι εφικτή , $\alpha\nu\nu$, οι $g_i (x) \geq 0 \;\; \forall i = 1,2,\ldots p$ και $h_i (x) = 0 \;\; \forall i = 1,2,\ldots q$ ισχύουν για το x. Όπως είναι φυσικό μια λύση μπορεί να βρεθεί μόνο στην εφικτή περιοχή λύσεων ,όπως για παράδειγμα το βέλτιστο (μέγιστο ή ελάχιστο) για ένα πρόβλημα βελτιστοποίησης(Weisse,2009).

Συναρτήσεις ποινών

Είναι πιθανώς από τις πιο διαδεδομένες προσεγγίσεις για τον χειρισμό περιορισμών ειδικά σε προβλήματα ενός κριτηρίου.Οι περιορισμοί συνδυάζονται με την αντικειμενική συνάρτηση f και προκύπτει μια νέα αντικειμενική προς ελαχιστοποίηση η f''.Η βασική ιδέα είναι ότι ο συνδυασμός των παραπάνω γίνεται κατα τέτοιο τρόπο ,ώστε να διασφαλιστεί ότι μια υποψήφια μη-εφικτή λύση έχει πάντα «χειρότερη » τιμή για την f'' από ότι μια εφικτή, δεδομένων των ίδιων αντικειμενικών τιμών,και κατεπέκταση ο αλγόριθμος θα την απορρίπτει.Η μέθοδος μπορεί να υλοποιηθεί με πολλούς τρόπους και συνήθως προστίθεται στην αρχική συνάρτηση μια συνάρτηση περιορισμού.

Περιορισμοί ως επιπλέον αντικειμενικές

Μια άλλη ιδέα για τον χειρισμό περιορισμών είναι να τις θεωρήσουμε ως νέες

αντικειμενικές συναρτήσεις .Εάν για παράδειγμα επιβάλλουμε τον περιορισμό $g(x) \geq 0$ μπορούμε να τον μετασχηματίσουμε σε μια νέα αντικειμενική $f^*(x) = \min\{-g(x), 0\}$ προς ελαχιστοποίηση.Άρα για την τελευταία συνάρτηση αρκεί να βρούμε το ελάχιστο ,εφόσον δεν έχει νόημα να μεγιστοποιήσουμε την g πέραν του 0.

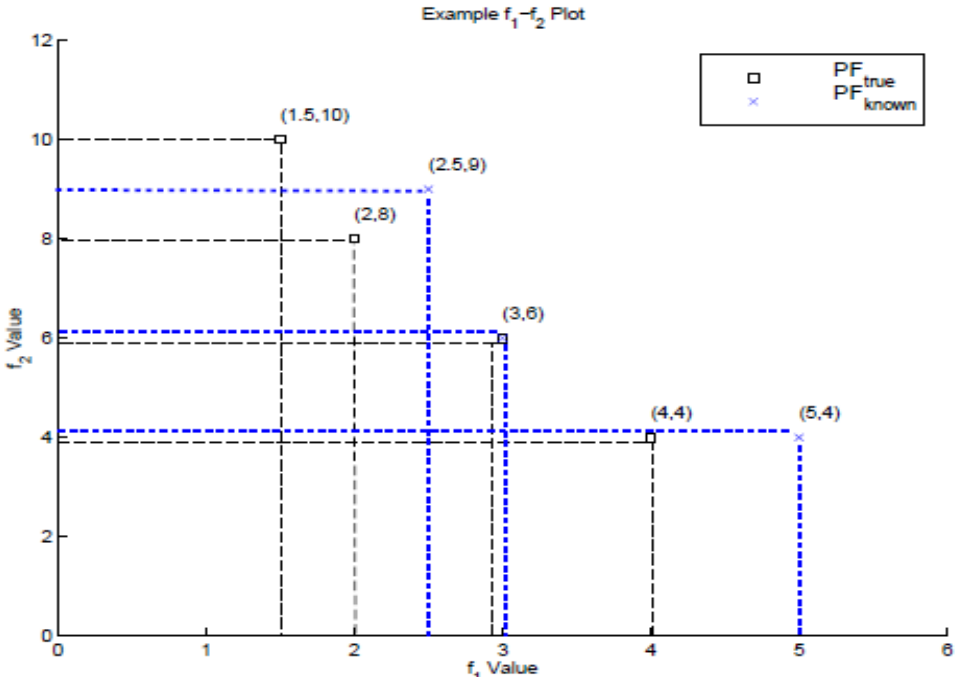
3.6.2 Μέτρα Απόδοσης(Performance metrics)

Στην πολυκριτηριακή βελτιστοποίηση η ανάλυση της ποιότητας του αποτελέσματος (πχ. το βέλτιστο Pareto σύνολο) είναι μια επίπονη εργασία εφόσον γίνεται προσπάθεια να καθορίσουμε ένα «καλό » σύνολο που να προσεγγίζει όσο το δυνατόν καλύτερα το βέλτιστο Pareto μέτωπο.Τα μέτρα απόδοσης λοιπόν είναι απαραίτητα ώστε να αξιολογήσουμε το καλύτερο μη-κυριαρχούμενο σύνολο λύσεων σε περιπτώσεις που είναι δύσκολο να ερμηνεύσουμε τα αποτελέσματα οπτικά,ενώ και η διαδικασία εκτίμησης του αποτελέσματος παραμένει αυτή καθαυτή ένα πολυκριτηριακό πρόβλημα , εφόσον διάφορα μέτρα θα πρέπει να ληφθούν υπόψη και κανένα μέτρο δεν υπερτερεί των άλλων , Angelo & Barbosa (2011).Ο Zitzler et al (Zitzler et al., 2000) πρότεινε τρία βασικά κριτήρια που θα πρέπει να λαμβάνουμε υπόψη κατα τη διαδικασία αξιολόγησης ενός Pareto συνόλου:

1. *Η απόσταση ανάμεσα στο υπο εξέταση Pareto σύνολο και στο « πραγματικό» Pareto μέτωπο θα πρέπει να ελαχιστοποιείται.*
2. *Είναι επιθυμητή μια καλή (στη πλειοψηφία των περιπτώσεων ομοιόμορφη) κατανομή των λύσεων του μη-κυριαρχούμενου συνόλου.*
3. *Το μέγεθος του Pareto μετώπου θα πρέπει να μεγιστοποιειίαι,πχ για κάθε αντικειμενική θα πρέπει να υπάρχει μια μεγάλη ποικιλια διακριτών λύσεων .*

Στη βιβλιογραφία έχουν προταθεί διάφορα μέτρα απόδοσης τα οποία μπορούν να ταξινομηθούν σε δύο κατηγορίες(Grossan,2003) :

1. ***Μέτρα σύγκλισης (Convergence metrics)**- αξιολογούν το πόσο μακριά απο το πραγματικό Pareto μέτωπο PF_{True} βρίσκονται οι λύσεις (διανύσματα) του Pareto μετώπου που προκύπτουν από τον αλγόριθμο βελτιστοποίησης.*
2. ***Μέτρα διαφοροποίησης (Diveristy metrics)**- αξιολογούν την **κατανομή** και το **εύρος** των λύσεων στο τελικό Pareto μέτωπο.*



Εικόνα 3.13 : Παράδειγμα συνόλων PF_{Known} / PF_{True}(Veldhuizen,1999).

3.6.2.1 Μέτρα σύγκλισης

Error ratio:*Ένας αλγόριθμος βελτιστοποίησης τερματίζει με ένα πεπερασμένο αριθμό διανυσμάτων που συνθέτουν το Pareto μέτωπο και συμβολίζουμε ως PF_{Known} ,ενώ αντίστοιχα το βέλτιστο/πραγματικό Pareto μέτωπο συμβολίζεται ως PF_{True} .Το συγκεκριμένο μέτρο έχει προταθεί από τον Veldhuizen(1999) και υπολογίζει την αναλογία των λύσεων που βρίσκονται στο PF_{True} ,αλλά όχι στο PF_{Known}. Η μαθηματική έκφραση του παραπάνω μέτρου είναι η εξής:*

$$E \qquad (3.24)$$

Όπου n είναι ο αριθμός των διανυσμάτων/λύσεων στο PF_{Known} και

Για παράδειγμα ,αν E=0 αυτό σημαίνει ότι κάθε λύση στο PF_{Known} ανήκει και στο PF_{True} – παρόμοια αν E =1 δεν ανήκει καμία.Στο παράδειγμα της Εικόνας 3.13 έχουμε E =2/3.

Generational Distance:*Το μέτρο αυτό υπολογίζει το πόσο «μακριά» βρίσκεται το PF_{Known} από το PF_{True} και ορίζεται ως:*

,όπου n ο αριθμός των λύσεων στο PF_{Known} ,p =2 ,ενώ το d_i είναι η ευκλείδια

απόσταση ανάμεσα σε κάθε λύση του προσεγγιστικού Pareto μετώπου (PF_{Known}) και της «κοντινότερης» λύσης του πραγματικού Pareto μετώπου (PF_{True}).Εάν $G=0$ τότε θα ισχύει $PF_{True}= PF_{Known}$ - οποιαδήποτε άλλη τιμή για το G δείχνει το μέγεθος της απόκκλισης του PF_{Known} από το PF_{True} . Σύμφωνα με την Εικόνα 3.13 έχουμε $d_1=$ - - , $d_2=0$, $d_3 =$ - - και $G=$ / $3=0.5$.

Maximum Pareto Front Error:*Υπολογίζει την μέγιστη απόσταση ανάμεσα σε κάθε λύση του PF_{Known} και στην αντίστοιχη κοντινότερη λύση που ανήκει στο PF_{True}.Μικρότερες τιμές του μέτρου υποδεικνύουν καλύτερα σύνολα.Ορίζεται ως :*

$$ME \quad max_j (min_i \ | f_1^i(\quad) - f_1^j(\quad)|^p + | f_2^i(\quad) - f_2^j(\quad)|^p)^{1/p}$$

,όπου $i =1,...,n_1$ και $j =1,...,n_2$ οι δείκτες των λύσεων στα PF_{Known} και PF_{True} αντίστοιχα και $p=2$. Μια τιμή 0 για το μέτρο υποδεικνύει ότι $PF_{Known} \subseteq PF_{True}$,ενώ οποιαδήποτε άλλη τιμή δείχνει ότι τουλάχιστον μια λύση στο PF_{Known} δεν ανήκει στο PF_{True}.Το παραπάνω μέτρο ωστόσο παραβιάζει την έννοια της ασθενούς μονοτονίας (weak monotony) , σύμφωνα με την οποία ένα μέτρο σύγκρισης R είναι συμβατό με μια σχέση σύγκρισης, πχ. " $\leq$ ", εάν για κάθε ζευγάρι μη-κυριαρχούμενων συνόλων A , B με $A \leq B$,το R θα αξιολογεί το A πάντα ως μη χειρότερο του B.Σύμφωνα ωστόσο με το συγκεκριμένο μέτρο ,είναι προτιμότερο να βρεθεί μια λύση κοντά στο PF_{True} ,παρα να βρεθούν δέκα λυσεις ,οι εννιά απο τις οποίες θα ανήκουν στο PF_{True} και μια η οποία θα απέχει σχετικά μακριά από το πραγματικό Pareto μέτωπο,κάτι φυσικά που δεν συμφωνεί με τη κοινή διαίσθηση για την ποιότητα ενός μη-κυριαρχούμενου συνόλου(Knowles & Come,2002).

Hyperarea and Ratio: *Οι Zitzler & Thiele(2001) πρότειναν ένα συγκριτικό μέτρο το οποίο ονομάζεται hyperarea και ορίζει την περιοχή του αντικειμενικού χώρου που καλύπτεται από το PF_{Known} (πχ. "η περιοχή κάτω από την καμπύλη του Pareto μετωπου").Για παράδειγμα μια λύση στο PF_{Known} για ένα Π.Π.Β. δύο αντικειμενικών ορίζει μια ορθογώνια περιοχή η οποία φράσσεται από ένα σημείο-αφετηρία(πχ (0,0)) και το σημείο/λύση $(f_1(\quad), f_2(\quad))$.Η ένωση όλων αυτών των περιοχών που ορίζουν όλες οι λύσεις στο PF_{Known} συνιστά το συγκριτικό μέτρο και ορίζεται ως:*

, (3.27)

,όπου υ_i είναι ένα μη-κυριαρχούμενο διάνυσμα στο PF_{Known} και α_i είναι η hyperarea που ορίζεται από κάθε υ_i και την αρχή των αξόνων.Λαμβάνοντας υπόψη το PF_{True} της Εικόνας 3.13 η ορθογώνια περιοχη που ορίζουν τα (0,0) και (4,4) έχει εμβαδό 16 μοναδων.Το ορθογώνιο αντίστοιχα που ορίζουν τα (0,0) και (3,6) συνεισφέρει (3(6-4))=6 μονάδες στη μέτρηση κ.ο.κ. Άρα θα είναι PF_{True}'s H = 16+6+4+3=29 μονάδες και αντίστοιχα θα έχουμε PF_{Known}'s H = 20+6+7.5=33.5 μονάδες . Ωστόσο όπως επισήμαναν οι Zitzler & Thiele το μέτρο αυτό μπορεί να είναι παραπλανητικό, εάν το PF_{Known} είναι μη κυρτό ,ενώ οι συντεταγμένες αφετηρίας δεν είναι πάντα (0,.....,0) όπως στο συγεκριμένο παράδειγμα.Ο Veldhuizen πρότεινε για τους παραπάνω λόγους ένα μέτρο αναλογίας (**ratio**) που ορίζεται ως εξής:*

$$\frac{\quad}{\quad}, \qquad (3.38)$$

,όπου H_1 είναι η hyperarea του PF_{Known} και H_2 εκείνη του PF_{True} .Σε ένα πρόβλημα ελαχιστοποίησης λοιπόν θα ισχύει $HR \geq 1$ και αντίστοιχα $HR \leq 1$ για προβλήματα μεγιστοποίησης.Σε κάθε περίπτωση θα έχουμε $PF_{Known} = PF_{True}$ εάν η αναλογία HR είναι ίση με 1.Στο παράδειγμά μας $HR = \frac{\quad}{\quad} = 1.155$.

Παρατηρούμε ότι τα παραπάνω μέτρα έχουν σχεδιαστεί ώστε να συγκρίνουν άμμεσα δύο σύνολα μη-κυριαρχούμενων λύσεων.Συνήθως αυτές οι μεθόδοι συγκρίνουν ένα σύνολο μη-κυριαρχούμενων λύσεων με ενα σύνολο βέλτιστων Pareto λύσεων και απαιτούν το «πραγματικό» Pareto μέτωπο, PF_{True} , να είναι γνωστό στον ερευνητή.

3.6.2.2 Μέτρα διαφοροποίησης

Schott's Spacing metric: *Το μέτρο αυτό εκτιμά την διαφοροποίηση των λύσεων στο PF_{Known} υπολογίζοντας πόσο ομοιόμορφα είναι κατανεμημένες στον αντικειμενικό χώρο(Schott,1997).Εφόσον η «αρχή» και το «τέλος» του PF_{Known} είναι γνωστά ένα κατάλληλο μέτρο θα μπορεί να αξιολογήσει πόσο καλά το σύνολο λύσεων στο PF_{Known} είναι διατεταγμένο στη περιοχή αυτή .Ο Schott λοιπόν προτείνει ένα μέτρο που υπολογίζει την στατιστική διακύμανση της απόστασης μεταξύ γειτονικών λύσεων σε ένα μη κυριαρχούμενο σύνολο λύσεων και το ορίζει ως εξής:*

$$\frac{\quad}{\quad}, \qquad (3.29)$$

,όπου $d_i = \min_i \left(| f_1^i(\quad) - f_1^j(\quad)| + | f_2^i(\quad) - f_2^j(\quad)| \right)$, i , $j = 1,\ldots,n$, (δηλαδη η ευκλίδεια απόσταση ανάμεσα στη λύση i του PF_{Known} και στη κοντινότερη λύση του PF_{Known} στην i), είναι ο μέσος όρος όλων των d_i και n ο αριθμός των λύσεων στο PF_{Known} .Μια τιμή 0 για το κριτήριο αυτό δείχνει ότι όλα τα μέλη του PF_{Known} είναι ισαπέχοντα διατεταγμένα. Όπως σημειώνεται από τον Veldhuizen(1999) το μέτρο θα πρέπει να τροποιηθεί κατάλληλα σε περιπτώσεις όπου, πχ. το PF_{Known} αποτελείται από δύο ή περισσότερες Pareto καμπύλες ,υπάρχει δηλαδη μια ασυνέχεια ανάμεσα στα διαφορετικά Pareto μέτωπα του PF_{Known} ,ενώ επίσης μπορεί να είναι παραπλανητικό εκτός και αν συνδυαστεί με ένα μέτρο που να δείχνει τη πληθικότητα των λύσεων στο βέλτιστο Pareto σύνολο που έχουμε υπολογίσει (πχ εάν παράγουμε μόνο δυο μη κυριαρχούμενες λύσεις το μέτρο θα δείξει μια ιδανική κατανομή) .Στο παράδειγμα μας είναι S=0.2887.

Spread: *Παρόμοια με το μέτρο του Schott , το μέτρο αυτό αξιολογεί την κατανομή των λύσεων του PF_{Known} στον αντικειμενικο χώρο.Συνοπτικά ,προκειμένου να αναλύσουμε την κατανομή των λύσεων, ο μη-κυριαρχούμενος χώρος χωρίζεται σε υποπεριοχές ίσων εμβαδών (ο αριθμός αυτός καθορίζεται απο τον λήπτη αποφάσεων,decision maker). Γνωρίζοντας τον αριθμό των λύσεων/διανυσμάτων που παράγονται από τον αλγόριθμο βελτιστοποίησης σε κάθε επανάληψη του ,μπορούμε να υπολογίσουμε τον αναμενόμενο αριθμό λύσεων σε κάθε υποπεριοχή και αρα να*

υπολογίσουμε το συγκεκριμένο μέτρο που ορίζεται ως εξής:

$$\frac{\quad}{\quad}$$

,όπου q είναι ο επιθυμητός αριθμός των λύσεων στο PF_{Known} ,n_i ο πραγματικός αριθμός των λύσεων στην i-οστη υποπεριοχή του μη-κυριαρχούμενου χώρου, ο αναμενόμενος αριθμός λύσεων στην i-οστη υποπεριοχή , $p=2$ και σ_i^2 η διακύμανση των λύσεων στην i-οστη υποπεριοχή.Τότε εάν $\iota=0$,η κατανομή των λύσεων στο PF_{Known} είναι ιδανική,άρα χαμηλές τιμές του μέτρου υποδεικνύουν και καλύτερη ικανότητα του αλγόριθμου βελτιστοποίησης να παράγει καλά κατανεμημένες λύσεις.
Τέλος σημειώνουμε ότι αν και τα παραπάνω δύο μέτρα δεν απαιτούν τη γνώση του PF_{True} ,θεωρούν οτι ο αλγόριθμος βελτιστοποίησης έχει συγκλίνει στο βέλτιστο Pareto μέτωπο («απέχει» δηλαδή σχετικά κοντά από το PF_{True}) ,διαφορετικά μια καλή κατανομή των μη κυριαρχούμενων λύσεων απο μόνη της δεν θα είχε ιδιαίτερη σημασία για την ποιότητα του αποτελέσματος.

3.6.2.3 Επιπλέον μέτρα

Overall Nondominated Vector Generation and Ratio(Veldhuizen,1999): *Το μέτρο αυτό υπολογίζει τον συνολικό αριθμό (πληθικότητα) των μη-κυριαρχούμενων λύσεων στο τέλος της εκτέλεσης του αλγόριθμου βελτιστοποίησης και ορίζεται ως:*

$$ONVG \quad |PF_{Known}| \quad , \qquad (3.31)$$

Αν και το μέτρο αυτό δίνει μια αίσθηση του πόσο αποτελεσματικός είναι ο αλγόριθμος στο να παράγει επιθυμητές λύσεις ,δεν αξιολογεί πόσο μακριά βρίσκονται οι λύσεις από το PF_{True} – επιπλέον μικρός αριθμός λύσεων θα αντιπροσωπεύει πάντα μια φτωχή απόδοση του αλγόριθμου,ενώ ένας μεγάλος αριθμός λύσεων μπορεί να προκαλέσει σύγχυση στον λήπτη αποφάσεων.Επειδή είναι δύσκολο να καθορίσουμε μια καλή τιμή για το $|ONVG|$ η οποία πιθανώς να αλλάζει ανάλογα με τον αλγόριθμο και την υπολογιστική ικανότητα , ορίζουμε σαν ένα επίπλεον μέτρο την αναλογία της πληθικότητας του PF_{Known} με αυτήν του PF_{True} :

$ONVGR \quad \frac{\quad}{\quad}$ *, που μας δίνει μια καλή εικόνα του αριθμού των μη κυριαρχούμενων λύσεων που έχουν βρεθεί ,σε σύγκριση με αυτές που υπάρχουν και δεν έχουν βρεθεί ακόμα.Μια τιμή 1 υποδεικνύει ότι έχει βρεθεί ο ίδιος αριθμός λύσεων που περιλαμβάνει και το PF_{True} . Στο παράδειγμα μας $ONVG=3$ και $ONVGR=0.75$.*

Generational Nondominated Vector Generation(Veldhuizen,1999)::*Το μέτρο αυτό εντοπίζει τον αριθμό των παραγόμενων μη κυριαρχούμενων λύσεων από τον αλγόριθμο βελτιστοποιήσης σε κάθε επανάληψη του t :*

$$GNVG \quad |PF_{current}(t)| \, , \qquad (3.32)$$

Nondominated Vector Addition(Veldhuizen,1999):: *Υπολογίζει τον αριθμό των νέων μη κυριαρχούμενων λύσεων που προκύπτουν από τον αλγόριθμο βελτιστοποίησης σε κάθε επανάληψη του t:*

$$NVA \quad |PF_{Known}(t)| - |PF_{Known}(t\text{-}1)| \ , \qquad (3.33)$$

Το παραπάνω μέτρο μπορεί να είναι παραπλανητικό ,καθώς εάν προστεθεί μια μη κυριαρχούμενη λύση στο $PF_{Known}(t)$ *μπορεί να κυριαρχεί κάποιων άλλων και κατεπέκταση να αφαιρεθούν από το* PF_{Known} *. Επιπλέον το μέγεθος του* $PF_{Known}(t)$ *μπορεί να παραμένει σταθερό για πολλές επαναλήψεις του αλγόριθμου ακόμα και εάν* $GNVG \neq 0$.

Σύμφωνα λοιπόν με τα παραπάνω ,δεν είναι εύκολο να ορίσουμε ένα μέτρο απόδοσης και πιθανώς δεν είναι εφικτό να σχεδιάσουμε ένα μόνο μέτρο το οποίο να ικανοποιεί όλες τις προτιμήσεις των χρηστών με αποδοτικό τρόπο, εφόσον κάθε ένα από αυτά αξιολογεί διαφορετικές πτυχές ενός μη-κυριαρχούμενου συνόλου.Στις περισσότερες περιπτώσεις το πραγματικό Pareto μέτωπο δεν είναι γνωστό και θα πρέπει να προσεγγιστεί πχ. από μια αριθμητική μέθοδο (enumerative approach). Επιπλέον η αξιολόγηση μόνο της κατανομής των λύσεων κατα μήκος του Pareto συνόλου μπορεί να είναι ανεπαρκής για την εκτίμηση της ποιότητάς του ,άρα είναι σημαντικό κατα την αξιολόγηση του αποτελέσματος να λαμβάνουμε υπόψιν όσο περισσότερα μέτρα και συνδυασμούς τους γίνεται.

3.6.3 Ant Colony Optimization για πολυκριτηριακή βελτιστοποίηση με περιορισμούς

Όπως αναφέραμε και παραπάνω κατα την διαδικασία στοχαστικής αναζήτησης για την βελτιστοποίηση πολυκριτηριακών προβλημάτων θα πρέπει να ληφθούν υπόψη δύο σημαντικά ζητήματα.Πρώτον η τακτική κατα την οποία θα αξιολογείται η καταλληλότητα των πιθανών λύσεων ώστε ο αλγόριθμος να συγκλίνει στο βέλτιστο Pareto σύνολο και δεύτερον θα πρέπει να διατηρείται η διασπορά των λύσεων στον αντικειμενικό χώρο (objective space) ,ώστε το τελικό Pareto μέτωπο να είναι καλά κατανεμημένο.Στην πολυκριτηριακή λήψη αποφάσεων δεν υπάρχει μια βέλτιστη λύση ,αλλά μια αποδοτική λύση θεωρείται αυτή όπου αύξηση σε μια από τις αντικειμενικές συναρτήσεις δεν μπορεί να γίνει χωρίς την μείωση σε τουλάχιστον μια από τις υπόλοιπες αντικειμενικές.Φυσικά η πλήρης ικανοποίηση μιας αντικειμενικής θα αποκλείει την πλήρη ικανοποιήση των υπολοίπων.

Ορίζουμε ένα πρόβλημα πολυκριτηριακής βελτιστοποίησης με περιορισμούς ως εξής(Barbosa et al,2008):

$$\begin{aligned} &\text{minimize} \quad f_{p,q}(x) \ , \qquad \mathbf{x}=[x_1, x_2, \ldots, x_n]^T \text{, όπου } \mathbf{x} \in R^n \\ &\text{subject to} \quad g_p(\mathbf{x}) \geq 0 \ , \qquad p = 1,2,\ldots, \\ &\qquad\qquad\quad h_q(\mathbf{x}) = 0 \ , \qquad q = 1,2,\ldots, \end{aligned}$$

, όπου το $\mathbf{x} \in R^n$ *είναι διάνυσμα που περιέχει τις μεταβλητές απόφασης* $\mathbf{x}=[x_1, x_2, \ldots, x_n]^T$, $f(\mathbf{x}) = [f_1(\mathbf{x}), f_2(\mathbf{x}), \ldots, f_n(\mathbf{x})]^T$ *είναι το διάνυσμα με τις αντικειμενικές , ενώ* $g_p(\mathbf{x})$ *και* $h_q(\mathbf{x}) = 0$ *οι περιορισμοί ανισότητας και ισότητας αντίστοιχα.*

Για την περίπτωση λοιπόν της πολυκριτηριακής βελτιστοποίησης ο Garcıa-Martınez et al (Garcıa-Martınez et al., 2007) προτείνει των διαχωρισμό των πολυκριτηριακών ACO (MOACO) αλγόριθμων βάση δυο κριτηρίων:

i. *τη χρήση μιας ή περισσοτέρων πληροφοριών φερομόνης(pheromone trails)*
ii. *και τη χρήση ενός ή περισσότερων πινάκων ευρετικής πληροφορίας (heuristic info)*

Επιπλέον με τα παραπάνω κριτήρια κάποιοι αλγόριθμοι μπορεί να ακολουθούν βελτιστοποίηση κατά Pareto,όπου επιστρέφουν ένα σύνολο μη-κυριαρχούμενων λύσεων(Pareto-based algorithms) ,ενώ κάποιοι άλλοι επιστρέφουν μια μοναδική λύση ως το τελικό αποτέλεσμα της εκτέλεσής τους (non Pareto-based algorithms).

Ο ψευδο-κώδικας για τον βασικό ACO αλγόριθμο στην πολυκριτηριακή βελτιστοποίηση παρουσιάζεται και αναλύεται παρακάτω(Angelo & Barbosa (2011)):

Algorithm 1: MultiobjectiveACO

```
Set parameters;
Initialize pheromone trails τ ;
Initialize heuristic matrix η ;
Initialize Pareto set P as empty;
while termination criteria not met do
    ConstructAntSolution();
    ApplyLocalSearch()    (optional);
    UpdateParetoSet();
    UpdateGlobalPheromone();
Return the Pareto set P
```

Αλγόριθμος 3.1: Η βασική διαδικασία ενός MOACO (Multi-Objective Ant Colony Optimisation) αλγορίθμου (Angelo & Barbosa (2011).

Αρχικά γίνεται αρχικοποίηση όλων των στοιχείων που χρησιμοποιούνται ,δηλαδή οι τιμές της αρχικής ποσότητας φερομόνης και της ευρετικής πληροφορίας που αναθέτουμε στις ακμές του γράφου ,πάνω στον οποίο κατασκευάζεται η λύση . Επίσης η διαδικασία απαιτεί την χρήση ενός συνόλου ή πίνακα P στον οποίο αποθηκεύονται οι μη κυριαρχούμενες λύσεις με βάση τα κριτήρια Pareto κατα την διαδικασία αναζήτησης στην εφικτή περιοχή και ανανεώνεται σε κάθε επανάληψη του αλγόριθμου ,ενώ στην αρχή της διαδικασίας ο πίνακας αρχικοποιείται ως κενός . Καλώντας την μέθοδο ***ConstructAntSolution()*** *τα μυρμήγκια εκκινούν την διαδρομή τους στον γράφο από ένα αρχικό σημείο, επιλέγοντας τον επόμενο κόμβο που θα επισκεφτούνε βάση μιας συγκεκριμένης πολιτικής.Η επιλογή αυτή κατευθύνεται από την ποσότητας της φερομόνης που έχει συγκεντρωθεί στα διάφορα μονοπάτια του γράφου και από την ευρετική πληροφορία που σχετίζεται με τον επόμενο κόμβο που θέλουμε να μεταβούμε.Εφόσον καθε μυρμήγκι ολοκληρώσει μια πλήρη διαδρομή (εφικτή λύση) στον γράφο,μπορούμε να εφαρμόσουμε μια διαδικασία τοπικής αναζήτησης (local search) μέσω της μεθόδου* ***ApplyLocalSearch()*** *,ώστε να βελτιωθεί η παραγόμενη λύση.Αφού όλα τα μυρμήγκια ολοκληρώσουν τις διαδρομές τους (λύσεις) ,λαμβάνει χώρα η ανανέωση του Pareto συνόλου μέσω της μεθόδου* ***UpdateParetoSet(),*** *η οποία διατηρεί όλες τις μη-κυριαρχούμενες λύσεις που έχουν παραχθεί μέχρι το σημείο αυτό της διαδικασίας. Κατόπιν ανανεώνεται η ποσότητα της φερομόνης στα διάφορα μονοπάτια του γράφου, βάση της ποιότητας της κάθε λύσης,ενώ επίσης ένα συγκεκριμένο ποσοστό φερομόνης «εξατμίζεται», μέσω της μεθόδου* ***UpdateGlobalPheromone()****. Επισημαίνουμε τέλος ότι πολιτική επιλογής των κόμβων κατα την κατασκευή μιας λύσης και ο τρόπος ανανέωσης της φερομόνης ,διαφέρουν ανάλογα με τις διάφορες παραλλαγές του αλγόριθμου και εξαρτώνται κυρίως από τον σχεδιαστή του και τη φύση του προβλήματος.*

Χειρισμός περιορισμών

Παίρνοντας ως παράδειγμα το πρόβλημα του Πλανώδιου Πωλητή ο μοναδικός περιορισμός που επιβάλλεται είναι να επισκέπτονται όλες οι πόλεις ακριβώς μια φορά.Ο χώρος αναζήτησης λοιπόν περιορίζεται σε λίστες πόλεων (1,....,N).Για την ικανοποίηση αυτού του περιορισμού ένας ACO αλγόριθμος εφοδιάζει με μνήμη τα μυρμήγκια για τις πόλεις που έχουν ήδη επισκεφτεί περιορίζοντας τις επιλογές τους στις πόλεις που δεν έχουν επισκεφτεί ακόμα. Σε άλλα όμως είδη προβλημάτων ,όπου οι διάφοροι περιορισμοί δεν μπορούν να ικανοποιηθούν κατα τη διάρκεια κατασκευής της λύσης ,χρειάζονται πιο προηγμένες τεχνικές που κατηγοριοποιούνται σε δυο γενικές κατηγορίες : direct,όπου λαμβάνονται υπόψη μόνο εφικτά στοιχεία κατα τη διαδικασία αναζήτησης και indirect ,όπου χρησιμοποιούνται εφικτά και μη εφικτά στοιχεία (Fonseca et al., 2007).

3.6.3.1 Αλγοριθμικά συστατικά ενός MOACO αλγορίθμου

Στην βιβλιογραφία υπάρχει μια μεγάλη ποικιλία υλοποιήσεων των ACO αλγορίθμων για πολυκριτηριακά προβλήματα ,τα οποία διαφέρουν συνήθως ως προς τον single objective αλγόριθμο στον οποίο έχει βασιστεί η ανάπτυξή τους (AS,ACS ή MMAS) και επιπλέον στις διάφορες παραλλαγές των αλγοριθμικών τους συστατικών. Παρακάτω παρουσιάζουμε τα πιό βασικά από αυτά, ενώ στον Πίνακα 3.4 δίνεται μια ταξινόμηση των πιο γνωστών υλοποιήσεων στη βιβλιογραφία :

3.6.3.1.1 Πολλαπλές αποικίες μυρμηγκιών:*Σύμφωνα με αυτή τη προσέγγιση ,ένας καθοριζόμενος από το χρήστη αριθμός μυρμηγκιών συνιστά μια αποικία.Κάθε αποικία κατασκευάζει λύσεις ανεξάρτητα η μια από την άλλη , λαμβάνοντας υπόψη την δική της πληροφορία φερομόνης και ευρετική πληροφορία και περιορίζοντας έτσι την αναζήτησή της σε συγκεκριμένες περιοχές του Pareto μετώπου.Οι αποικίες συνεργάζονται μεταξύ τους είτε i) ανταλλάσοντας λύσεις (πληροφορία) μέσω ενός κοινόχρηστου αρχείου μη κυριαρχούμενων λύσεων (archive), ούτως ώστε να αναγνωρίζουν τις κυριαρχούμενες λύσεις και το ανανεώνουν κατάλληλα κατα την διάρκεια της εκτέλεσης ή ii) διαμοιράζοντας λύσεις η μια με την άλλη για την κατάλληλη ανανέωση της πληροφορίας φερομόνης ,έτσι ώστε οι λύσεις που παράγονται από μια αποικία να επηρεάζουν την «γνώση» ,ως προς την βέλτιστη λύση, που έχει συσσωρευτεί από την εναπόθεση φερομόνης σε άλλες αποικίες.*

3.6.3.1.2 Πληροφορία φερομόνης και ευρετική πληροφορία:*Υπάρχουν δύο βασικά μοντέλα για να εισάγουμε την ευρετική/φερομόνης πληροφορία στον αλγόριθμό μας: χρησιμοποιώντας έναν ή πολλαπλούς πίνακες.Στην περίπτωση των πολλαπλών πινακών κάθε πίνακας περιέχει πληροφορία που αντιστοιχεί σε μια αντικειμενική συνάρτηση,ενώ οι αρχικοποίηση τους και οι τιμές τους ως προς την πληροφορία φερομόνης, μπορεί να διαφέρουν ανάλογα με την υλοποίηση που ακολουθείται.Στην περίπτωση ενός μόνο πίνακα φερομόνης, η κατασκευή των λύσεων γίνεται κατα παρόμοιο τρόπο με την περίπτωση των single objective ACO αλγόριθμων – η πληροφορία φερομόνης που σχετίζεται με κάθε αντικειμενική συνδυάζεται ,ανάγοντας το πρόβλημα σε μονοκριτηριακό.Η ίδια λογική εφαρμόζεται και για την ευρετική πληροφορία.*

3.6.3.1.3 Προσάθροιση φερομόνης και ευρετικής πληροφορίας: *Στην περίπτωση που χρησιμοποιούνται πολλαπλοί πίνακες η πληροφορία τους θα πρέπει να προσαθροιστεί μέσω κάποιας προσαθροιστικής διαδικασίας, ώστε να κατευθύνει κατάλληλα την διαδικασία κατασκευής της λύσης ,ενώ σε κάθε περίπτωση ορίζονται βάρη τα οποία ρυθμίζουν την επίδραση της κάθε αντικειμενικής συνάρτησης στο τελικό αποτέλεσμα. Για τους MOACO αλγόριθμους ,μπορούμε να επιλέξουμε ανάμεσα*

σε 3 μεθόδους για την προσάθροιση της πληροφορίας: i) η μέθοδος του σταθμισμένου αθροίσματος (weighed sum) ,όπου οι πίνακες συγχωνεύονται μέσω ενός αθροίσματος βαρών ,( ), K είναι ο αριθμός των αντικειμενικών προς βελτιστοποίηση , λ_k , $k=1,\ldots K$ τα βάρη που αντιστοιχούν σε κάθε αντικειμενική και τ_{ij}^{k} η πληροφορία φερομόνης της k-οστης αντικειμενικής για το μονοπάτι που περνάει από τους κόμβους i,j , ii) η μέθοδος του σταθμισμένου γινομένου (weighed product), όπου οι πίνακες συχγωνεύονται μέσω ενός γινομένου βαρών, ( ) , και τέλος iii) τυχαία, όπου σε κάθε βήμα κατασκευής της λύσης επιλέγεται τυχαία η πληροφορία φερομόνης από τις διάφορες αντικειμενικές προς βελτιστοποίηση.Τέλος ο ορισμός των βαρών μπορεί να γίνει είτε ***δυναμικά*** *,όπου αναθέτονται διαφορετικά βάρη στις αντικειμενικές συναρτήσεις κατα τη διάρκεια του χρόνου, δίνοντας την ανάλογη βαρύτητα σε κάθε αντικειμενική σε διαφορετικά χρονικα σημεια της εκτελεσης ,επιτρεπώντας έτσι στα μυρμήγκια να συνδυάζουν την αντικειμενική πληροφορία χρησιμοποιώντας συγκεκριμένα βάρη- ή* ***στατικά****, όπου τα βάρη ορίζονται a-priori και κάθε αντικειμενική συνάρτηση διατηρεί την ίδια βαρύτητα καθόλη τη διάρκεια εκτέλεσης του αλγόριθμου.*

Component	Values
Colony	Single, Multiple
Pheromone matrix	Single, Multiple
Heuristic matrix	Single, Multiple
Aggregation	Weighted product, Weighted sum, Random
Weight setting	Dynamic, Fixed
Pheromone update	Non-dominated, Best-of-objective, Elite, All
Pareto archive	Offline, Online, No-archive

Πίνακας 3.3 :Βασικά συστατικά και τιμές των MOACO αλγορίθμων(Angelo & Barbosa (2011)).

Algorithm	[τ]	[η]	Colony	Agg./Weight	τ update	P. archive
MOAQ (Mariano & Morales, 1999)	1	k	multiple	w.p-w.s/-	ND	off
COMPETants (Doerner et al., 2001)	k	k	multiple	w.s./fix.	BoO	none
BicriterionAnt (Iredi et al., 2001)	k	k	single	w.p./dyn.	ND	off
SACO (Vincent et al., 2002)	1	1	single	-/-	E	none
MONACO (Cardoso et al., 2003)	k	1	single	w.p./dyn.	all	off
P-ACO (Doerner et al., 2004)	k	k	single	w.s./dyn.	BoO	off
M3AS (Pinto & Barán, 2005)	1	k	single	w.p./fix.	ND	off
MOA (Gardel et al., 2005)	1	1	single	w.p./dyn.	ND	off
MAS (Paciello et al., 2006)	1	k	single	w.p. /dyn.	ND	off
MOACSA (Yagmahan & Yenisey, 2010)	1	1	single	w.p./dyn.	E	none

Πίνακας 3.4:Ταξινόμηση των MOACO αλγορίθμων βάση των αλγ. τους συστατικών(Angelo & Barbosa (2011)).

Πιό συγκεκριμένα εφόσον κάθε αντικειμενικό κριτήριο ορίζει διαφορετικά τον τρόπο που θα επιλέγει και θα προσθέτει το επόμενο συστατικό στη μερική λύση,ο

πράκτορας-μυρμήγκι λαμβάνει υπόψη τις τιμές καθενός από τα αντικειμενικά κριτήρια για να καθορίσει το πιθανό συστατικό που θα προσθέσει(Lopez-Ibanez,2004).Οι τιμές αυτές μπορουμε να θεωρήσουμε ότι ανήκουν σε ένα διάνυσμα κατάστασης S, στο οποίο πχ,δύο αντικειμενικές συναρτήσεις q και r με διαφορετικούς ορισμούς των συστατικών λύσης,θα έχουν και διαφορετικές τιμές στα αντίστοιχα στοιχεία τους στο διάνυσμα κατάστασης($S_q \# S_r$),ενώ εαν έχουν την ίδια ερμηνεία των συστατικών τότε και οι αντίστοιχες τιμές στο S, θα είναι ίδιες($S_q = S_r$).Ας θεωρήσουμε ένα Π.Π.Β με I αντικειμενικές.Τότε το k-οστο μυρμήγκι θα προσθέσει το συστατικο j στη μερική λύση με πιθανότητα :

,όπου $S= \{ S_1,...., S_I \}$ είναι το διάνυσμα κατάστασης της τρέχουσας μερικής λύσης και κάθε στοιχείο του S_i έχει τιμή καθοριζόμενη απο την αντίστοιχη αντικειμενική $i=1,....,I$ που ορίζει ένα συστατικό λύσης, είναι η εφικτή γειτονιά για το k-οστο μυρμήγκι βαση του τρέχοντος διανύσματος κατάστασης S , είναι η πληροφορία φερομόνης του j δεδομένου του S_i για την i-οστη αντικειμενική στη τρέχουσα επανάληψη,ομοίως ορίζεται και η ευρετική πληροφορία και τέλος λ_i είναι το βάρος για την i-οστη αντικειμενική ,όπως ορίστηκε στην εν. 5.1.2. .Βλέπουμε ότι όταν $\lambda_i=0$, η i-οστη αντικειμενική δεν λαμβάνετε υπόψη ,διαφορετικά είναι η μόνη που βελτιστοποιείται($\lambda_i=1$).

Ένα απλό παράδειγμα(Lopez-Ibanez,2004),όπως συναντάται συχνά σε προβλήματα μιας μηχανής,είναι ένα πρόβλημα χρονοπρογραματισμού δυο κριτηρίων, όπου η τιμή της μιας αντικειμενικής εξαρτάται από την θέση στην οποία θα τοποθετηθεί στο χρονοπρόγραμμα μια εργασία ,ενώ η δεύτερη από την θέση στην οποία θα τοποθετηθεί η εργασία βάση των προηγούμενων εργασιών που έχουν προγραμματιστεί (Single Machine Total Tardiness Problem with Changeover Costs,SMTTP). Στη περίπτωση αυτή μπορούμε να ορίσουμε τα συστατικά μιας λύσης για την 1η αντικειμενική, ως αναθέσεις εργασιών σε θέσεις του χρονοπρογράμματος και για την 2η κατα τον ίδιο τρόπο ,λαμβάνοντας υπόψη όμως κατα τον υπολογισμό του κριτηρίου τις σχέσεις διαδοχής/προκατοχής μεταξύ των εργασιών στο χρονοπρόγραμμα.Δεδομένης λοιπόν μιας λύσης φ (δηλ. μιας ακολουθίας από εργασίες στη μηχανή) ,η πληροφορια φερομόνης για την 1η αντικειμενικη θα αντιπροσωπευει την ελκυστικότητα του να τοποθετήσουμε την εργασια j στην θέση i του χρονοπρογράμματος και για την 2η αντικειμενική την ελκυστικότητα η εργασία j να αρχίσει την επεξεργασία της ,αμέσως μετά την περάτωση της εργασίας h στην i-οστή θέση του χρονοπρόγραμματος.Είναι ευνόητο για το παραπάνω σενάριο ότι χρειαζόμαστε δύο πίνακες φερομόνης, εφόσον η είσοδος τ_{ij} για κάθε αντικειμενική στον αντίστοιχο πίνακα θα έχει διαφορετική ερμηνεία.Για να αποφασίσει το μυρμήγκι την επόμενη εργασία j που θα προσθέσει στο πρόγραμμα χρειάζεται την γνώση της θέσης i και της τελευταίας εργασίας h που έχει προγραμματιστεί στη μηχανή.Το διάνυσμα κατάστασης της μερικής λύσης θα είναι $S= \{ i, h \}$,επίσης $\{ I=2\}$ και το k-οστο μυρμήγκι θα αποφασίσει με πιθανότητα :

$$h)_j = \frac{\quad}{\quad}$$

,όπου η εφικτή γειτονιά του k-οστου μυρμηγκιού,εκείνες δηλ. οι εργασίες που δεν έχουν ακόμα προγραμματιστεί, και η ποσότητα φερομόνης και η ευρετική πληροφορία αντίστοιχα για την 1η αντικειμενική ,ομοίως και η ποσότητα φερομόνης και η ευρετική πληροφορία για την 2η αντικειμενική και τέλος $\lambda \in [0,1]$ τιμή που καθορίζει την βαρύτητα κάθε αντικειμενικής κατα την διαδικασία της βελτιστοποίησης.

*Στην περίπτωση του **JSP** τα συστατικά μιας λύσης για τις αντικειμενικές που μελετάμε στην συγκεκριμένη διπλωματική(C_{max} ,T_d) ορίζονται κατά τον ίδιο τρόπο ,δηλαδή ως οι μεταβάσεις από μια εργασία i σε μια εργασία j. Για να επιλέξουμε την επόμενη εργασία j προς επεξεργασία το μυρμήγκι χρειάζεται να γνωρίζει μόνο την τελευταία εργασία i που έχει ανατεθεί στο χρονοπρόγραμμα,δηλαδή θα είναι S= { i }. Επιπρόσθετα εάν θεωρήσουμε δύο κριτήρια προς ελαχιστοποίηση ,{ I=2} ,σύμφωνα με την Εξ. 3.34 το μυρμήγκι k επιλέγει την διεργασία j ,ενώ έχει αναθέσει την εργασια i στο χρονοπρόγραμμα με πιθανότητα :*

$$j) = \frac{\quad}{\quad}$$

,όπου $\mathbf{N}_i^k$ το σύνολο των διαθέσιμων εφικτών κόμβων για το k-οστό μυρμήγκι,τ_{ij}^1και τ_{ij}^2 η πληροφορια φερομόνης για κάθε αντικειμενική, n_{ij}^1και n_{ij}^2 συγκεκριμένη ευρετική πληροφορία για κάθε αντικειμενική και τέλος $\lambda \in [0,1]$ η τιμή βαρύτητας για κάθε αντικειμενική

Από την Εξ. 5.22 προκύπτει και ένα πιθανό ζήτημα στην περίπτωση που οι αντικειμενικές ορίζουν τα συστατικά της λύσης κατα τον ίδιο τρόπο(Lopez-Ibanez,2004).Κατά την ανανέωση των πινάκων φερομόνης εάν τα συστατικά λάβουν την ιδια ποσότητα φερομόνης ανεξάρτητα από τις αντικειμενικες που τα ορίζουν ,ισχύει δηλαδή $\tau_{ij}^1 = \tau_{ij}^2 = \tau$, η προσάθροιση των πινάκων φερομόνης δεν θα έχει νόημα εφόσον $\tau^{(1-\lambda)} \cdot \tau^{\lambda} = \tau$, θα πρέπει λοιπόν να διασφαλίζουμε ότι τα διάφορα στοιχεία φερομόνης που ορίζονται στα πλαίσια ισοδύναμων(ίδιων) συστατικών λύσης είναι στην πραγματικότητα διαφορετικά.

Στην περίπτωση ενός μόνο πίνακα φερομόνης το κύριο πρόβλημα παραμένει ο ορισμός της πληροφορίας φερομόνης για τις αντικειμενικές που ορίζουν διαφορετικά τα συστατικά της λύσης τους.Και πάλι μπορούμε να χρησιμοποιήσουμε ένα διάνυσμα κατάστασης S που αντιπροσωπεύει την τρέχουσα κατάσταση της μερικής λύσης ,οπότε το k-οστό μυρμήγκι προσθέτει το j στη λύση με πιθανότητα:

$$\frac{\quad}{\quad}$$

,όπου πίνακας διαστάσεων όσων και τα στοιχεία του S,ο αριθμός δηλαδή των

διαφορετικών τρόπων που μπορεί να οριστούν τα συστατικά της λύσης συν μια ακόμα για το j. Χάριν απλότητας έχουμε θεωρήσει τα ίδια βάρη για όλες τις αντικειμενικές.Στην περίπτωση δυο αντικειμενικών όπως στο παράδειγμα μας, όπου το S περιέχει την θέση i και την εργασία h , η πληροφορία φερομόνης θα αντιπροσωπεύεται από ένα πίνακα φερομόνης $[\tau_{ihj}]$ διαστάσεων $IxJxJ$, όπου I το μέγεθος του χρονοπρογράμματος και J ο αριθμός των εργασιών.Τότε σύμφωνα με την Εξ. 3.34 ,για (I=2) ,το k-οστό μυρμήγκι προσθέτει το j στο χρονοπρόγραμμα με πιθανότητα:

$$h)_j = \frac{\qquad\qquad}{\qquad\qquad}$$

,ενώ για το JSP ,όπου S= { i } τα πράγματα απλουστεύουν ακόμα περισσότερο και ο πίνακας φερομόνης $[\tau_{ij}]$ θα έχει την ίδια δομή όπως στον μονοκριτηριακό ACS. Ολοκληρώνοντας ιδιαίτερης προσοχής χρίζει και ο ορισμός του διανύσματος βαρών.Όπως φαίνεται στην Εξ. 5.20 πολλαπλή πληροφορία φερομόνης συγωνευεται σε μια τιμή με την χρήση βαρών ή πιό τυπικά ενός διανύσματος βαρών.Στην περίπτωση δύο κριτηρίων, μια μόνο τιμή του λ αρκεί για να ορίσει το διάνυσμα βαρών { λ, $1-\lambda$ } ,το οποίο και ρυθμίζει την σχετική σημασία των διαφόρων αντικειμενικών.Έστω το άπειρο σύνολο Λ ορίζει όλες τις πιθανές κατευθύνσεις που μπορούν να προσεγγίσουν το βέλτιστο σύνολο Pareto και το πεπερασμένο υποσύνολο Γ ορίζεται ως μια υποπεριοχή του Λ .Εφόσον ο στόχος μας είναι η εξαγωγή ενός συνόλου Pareto παρά μιας μοναδικής λύσης ,θα πρέπει να χρησιμοποιήσουμε ένα πεπερασμένο σύνολο Γ , ($\Gamma \subseteq \Lambda$), μέγιστα διασκορπισμένων(dispersed) διανυσμάτων βαρών,πχ. τα διανύσματα βαρών στο Γ να είναι όσο το δυνατόν καλύτερα κατανεμημένα στον άπειρο χώρο που ορίζει το Λ.

Ο Iredi et al(2001). πρότεινε την ανάθεση κάθε διανύσματος βαρών $\lambda_k \in \Gamma$ σε διαφορετικό μυρμήγκι k,άρα σε κάθε κύκλο του αλγόριθμου χρησιμοποιείται το σύνολο των διανυσμάτων χωρίς να παίζει ρόλο η σειρά που ανατίθονται σε αυτά , εφόσον στους ACO αλγόριθμους τα μυρμήγκια κατασκευάζουν ανεξάρτητα τις λύσεις σε δεδομένο κύκλο.Εναλλακτικά θα μπορούσαμε να αναθέσουμε κάθε διάνυσμα $\lambda_t \in \Gamma$ σε κάθε διαφορετικό κύκλο t.Σε αυτή τη περίπτωση ο πληθύσμος των μυρμηγκιών χρησιμοποιεί τα ίδια βάρη σε συγκεκριμένο κύκλο και διαφορετικά στον επόμενο., και εφόσον οι πληθυσμοί των μυρμηγκιών σε δύο διαδοχικούς κύκλους δεν είναι ανεξάρτητοι είναι λογικό δύο διαδοχικά διανύσματα βαρών να διαφέρουν κατα z σε δύο οποιαδήποτε στοιχεία τους ,όπου z η ελάχιστη διαφορά αναμεσα σε δύο οποιεσδήποτε τιμές ενός στοιχείου για όλα τα διανύσματα βαρών στο Γ.

Το διάνυσμα βαρών είναι ουσιαστικά μια κατεύθυνση αναζήτησης του αντικειμενικού χώρου ,όπου η πρώτη μέθοδος εξερευνά όλες τις πιθανές κατευθύνσεις σε κάθε κύκλο,ενώ η δεύτερη προσπαθεί να βρει μια καλή λύση ,ως προς μια κατεύθυνση σε ένα κύκλο και τη χρησιμοποιεί ως σημείο εκκίνησης στον επόμενο ,αλλά προς ελαφρώς διαφορετική κατεύθυνση.

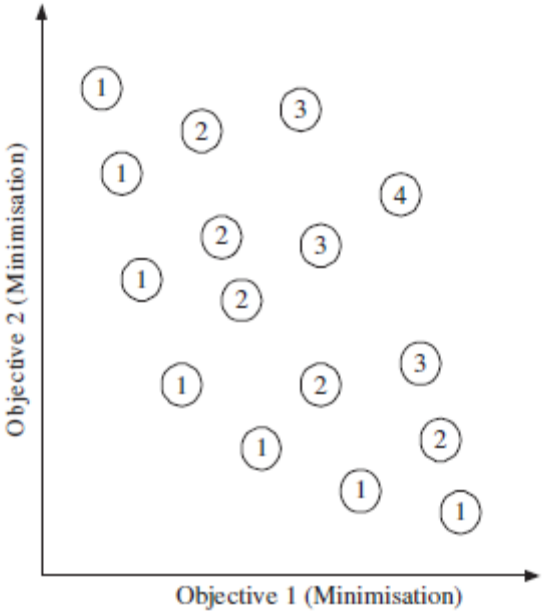
3.6.3.1.4 Αξιολόγηση Pareto μετώπου(Pareto ranking):*Η αξιολόγηση των λύσεων ενός Pareto μετώπου έχει προσεγγιστεί στο πεδίο της πολυκριτηριακής βελτιστοποίησης με διάφορες μεθόδους,όπως η μη-κυριαρχούμενη ταξινόμηση (non_dominated sorting) και Pareto βαθμονόμηση(Pareto ranking, Corne et al. 2001;*

Deb et al. 2000; Horn et al. 1994; Knowles and Corne 2000; Srinivas and Deb 1994; Zitzler et al. 2002).Η αξιολόγηση των λύσεων έχει ιδιαίτερη σημασία στην οικογένεια των MOACO αλγόριθμων ,καθώς αναθέτει μια τιμή στη ποιότητα των λύσεων που σύχνα χρησιμοποίειται κατα την διαδικασία ανανέωσης της φερομόνης(Angus &Woodward ,2007).Οι μέθοδοι αξιολόγησης ταξινομούνται ως εξής:

- ***Pareto:*** *Οι λύσεις αξιολογούνται λαμβάνοντας υπόψη όλες τις αντικειμενικές τιμές τους και τους ανατίθεται ένα σκόρ που αντιπροσωπεύει πόσο ικανοποιητικά αντιπροσωπεύουν αυτές τις αντικειμενικές πχ. κατα μια Pareto έννοια(dominance ranking).Χαρακτηριστικά παραδείγματα είναι οι αλγόριθμοι MOAQ(Romero & Manzanares 1999), CPACO (Angus 2007), PACO-MO (Guntsch & Middendorf 2003).Οι τελευταίοι συνδύαζουν τις τιμές των αντικειμενικών κάθε λύσης σε ένα συνολικό σκόρ λαμβάνοντας υπόψη μόνο τις μη-κυριαρχούμενες λύσεις του πληθυσμού σε κάθε επανάληψη(εκτός του CPACO) ,κατα παρόμοιο τρόπο με τους μονοκριτηριακούς ACO αλγ. οι οποίοι δημιουργούν μια ισχυρή πόλωση κοντά στις elite λύσεις.Ένα πιθανό ζήτημα που προκύπτει είναι η απώλεια διαφοροποίησης στον πίνακα/ες φερομόνης που μεταφράζεται σε μείωση διαφοροποίησης του τελικού Pareto μετώπου.*

Χαρακτηριστικό παράδειγμα είναι ο CPACO αλγόριθμος(Angus,2007) ο οποίος επιτρέπει την ανανέωση της φερομόνης και από τις κυριαρχούμενες λύσεις σε μικρότερο ωστόσο βαθμό από τις μη-κυριαρχούμενες ,εφαρμόζοντας μια τεχνική ταξινόμησης των λύσεων βασιζόμενη στη κυριαρχία.Το σχήμα διατηρεί έναν πληθύσμο S πεπερασμένου μεγέθους ο οποίος αρχικοποίειται με τυχαία παραγόμενες λύσεις.Συγκεκριμένα για κάθε νέο πληθυσμό λύσεων που παράγουν τα μυρμήγκια σε μια επανάληψη η κάθε λύση συγκρίνεται με ένα τυχαίο επιλεγμένο υποσύνολο S'' του συνόλου S . Κατόπιν επιλέγεται η κοντινότερη λύση από το S'' και την οποία η λύση απο τον πληθυσμό αντικαθιστά στο S εάν είναι καλύτερη υπό την έννοια της κυριαρχίας.Ο αλγόριθμος χρησιμοποιεί μονάδικο πίνακα φερομόνης γιά όλες τις αντικειμένικές ,ο οποίος σε κάθε επανάληψη ανανεώνεται ως εξής:

1. *Ανατίθονται σε όλες τις λύσεις (s) ∈ S ακέραιοι βαθμόι(ranks) σύμφωνα με μια διαδικασία βαθμονόμησης βασιζόμενη στη κυριαρχία,η οποία χρησιμοποίειται στον NSGA-II αλγόριθμο(Εικόνα 3.14).Για παράδειγμα οι μη-κυριαρχούμενες λύσεις θα έχουν βαθμό 1, οι περισσότερο κυριαρχούμενες λύσεις βαθμό 2 κ.ο.κ.*
2. *Όλα τα στοιχεία του πίνακα φερομόνης ανανεώνονται σε μια αρχική τιμή (τ_{init}).*
3. *Όλες οι λύσεις στον S αυξάνουν τα αντίστοιχα στοιχεία στον πίνακα φερομόνης κατα αντίστροφη αναλογία του βαθμού τους (πχ $\Delta\tau^{s}_{ij}=1/s_{rank}$).*



Εικόνα 3.14:Παράδειγμα των βαθμών(ranks) που ανατίθονται σε ένα σύνολο λύσεων με χρήση της μεθόδου αξιολόγησης κυριαρχίας του NSGA-II(Angus,2007).

- ***Non-Pareto****:Οι λύσεις αξιολογούνται βάση των τιμών μίας ή περισσοτέρων αντικειμενικών και τους ανατίθονται κάποια σκόρ ανάλογα με το πόσο καλά ικανοποιούν μια συγκεκριμένη αντικειμενική πχ.ένα σκόρ βασιζόμενο στην προσάθροιση όλων των τιμών των αντικειμενικών κριτηρίων της λύσης. Παραδείγματα αλγόριθμων που χρησιμοποιούν Non-Pareto αξιολόγηση των λύσεων τους είναι οι MOACOM (Gravel et al. 2002) και MACS (Barán & Schaerer 2003).Οι πρώτοι επιλύουν το πολυκριτηριακό πρόβλημα χρονοπρογραμματισμού της συνεχούς ροής αλουμινίου(continuous casting of aluminim) με 4 κριτήρια .Στο τέλος κάθε κύκλου υπολογίζονται οι τιμές που αντιστοιχούν σε κάθε αντικειμενική h(h=1,2,3,4) για το χρονοπρόγραμμα που έχει βρεθεί από το k-οστό μυρμήγκι.Έστω αυτές οι τιμές $L^h_k(t)$.Οι τέσσερις αντικειμενικές ταξινομούνται βάση της σημαντικότητάς τους από τον σχεδιαστή της εταιρίας(company planner,η απόφαση αυτή ανήκει στον σχεδιαστή ανάλογα με το κριτήριο που επιθυμεί να βελτιστοποίησει σε μεγαλύτερο βαθμό) και έστω h=h' η πιο σημαντική αντικειμενική και $L^{h'}_k(t)$ η τιμή της h' αντικειμενικής για το χρονοπρόγραμμα.Τότε η συνεισφορά του k-οστού μυρμηγκιού στον πίνακα φερομόνης θα είναι $\Delta\tau^k_{ij}=1/L^{h'}_k(t)$. Όπως βλέπουμε οι σχεδιαστές επιλέγουν την τιμή του πιό σημαντικού κριτηρίου δεδομένου ενός διανύσματος λύσης για να ανανεώσουν την φερομόνη.*
  Οι Barán & Schaerer από την άλλη συχγωνεύουν όλες τις τιμές των κριτηριών για δεδομένη λύση σε μία και ανανέωνουν τον πίνακα φερομόνης ως εξής $\Delta\tau^k_{ij}=1/L^{h1}_k(t)*L^{h2}_k(t)$.Οι συγγραφείς επιλύουν το Vehicle Routing Problem with time windows θεωρώντας δύο κριτήρια h1 και h2.

3.6.3.1.5 Ανανέωση φερομόνης: *Η ανανέωση φερομόνης μπορεί να επιτευχθεί με διάφορους τρόπους .Στην περίπτωση ενός πίνακα φερομόνης η ανανέωση γίνεται κατά παρόμοιο τρόπο με τους μονοκριτηριακούς ACO αλγορίθμους ,όπως για παράδειγμα επιλέγοντας την καλύτερη λύση σε κάθε επανάληψη (iteration-best) ή την καλύτερη εώς τώρα λύση (elite solution) και ανανεώνοντας τη φερομόνη στα σχετικά μονοπάτια του γράφου.Αυτή η διαδικασία μπορεί να εφαρμοστεί και όταν έχουμε πολλαπλούς πίνακες,επιλέγωντας αντίστοιχα ένα σύνολο καλύτερων λύσεων σε κάθε*

επανάληψη του αλγόριθμου ή το σύνολο των βέλτιστων λύσεων εώς τώρα κατα την εκτέλεση της διαδικασίας,όπου κάθε καλύτερη/βέλτιστη λύση του συνόλου αντιστοιχεί σε μια από τις αντικειμενικές που βελτιστοποιούμε (best-of-objective-solutions) .Ένας άλλος τρόπος είναι η συλλογή και αποθήκευση των μη κυριαρχούμενων λύσεων σε ένα εξωτερικό σύνολο σε περίπτωση που ο αλγόριθμος είναι Pareto-based.Η ανανέωση της φερομόνης γίνεται μόνο από τις λύσεις του μη κυριαρχούμενου συνόλου και μπορεί να περιλαμβάνει ανανέωση ενός ή πολλαπλών πινάκων φερομόνης,ενώ σε άλλες περιπτώσεις όλες οι παραγόμενες λύσεις σε κάθε επανάληψη μπορούν να ανανεώνουν τους πίνακες φερομόνης(Barbosa & Angelo,2011).

3.6.3.1.6 Αρχείο Pareto: *Όπως αναφέραμε παραπάνω, για τους Pareto-based MOACO αλγόριθμους,το Pareto σύνολο αποθηκεύεται και ανανεώνεται κατά τη διάρκεια της εκτέλεσης του αλγόριθμου και συχνά χρησιμοποιείται για την ανανέωση της φερομόνης.Υπάρχουν αρκετές επιλογές για τον τρόπο αποθήκευσης του αλλά και τον τρόπο που θα χρησιμοποιηθεί στην διάρκεια της διαδικασίας(Angus &Woodward ,2007):*

- *Offline storage:Αφού κατασκευαστούν οι νέες λύσεις σε μια επανάληψη του αλγόριθμου ,χρησιμοποιούνται για την ανανέωση της φερομόνης σε ένα ή περισσότερους πίνακες-ανάλογα με την υλοποίηση- και κατόπιν προστίθονται σε ένα εξωτερικό χώρο αποθήκευσης (archive) ,όπου διατηρείται ένα σύνολο Pareto βέλτιστων λύσεων .Οι μη κυριαρχούμενες λύσεις του αρχείου δεν χρησιμοποιούνται για μελλοντική κατασκευή λυσεων (πχ. δεν χρησιμοποιούνται για την ανανέωση της φερομόνης σε μελλοντικές επαναλήψεις του αλγόριθμου) ,ενώ στο τέλος της διαδικασίας το αρχείο επιστρέφεται ως η τελική λύση.*
- *Online storage: Ομοίως με την παραπάνω διαδικασία με την διαφορά ότι οι λύσεις του Pareto συνόλου (archive) χρησιμοποιούνται για την ανανέωση της φερομόνης σε κάθε επανάληψη του αλγόριθμου.Στο τέλος της διαδικασίας το αρχείο επιστρέφεται ως η τελική λύση.*
- *Elite:Διατηρείται μια ελιτίστικη λυση (elite solution) καθόλη τη διάρκεια της εκτέλεσης,η οποία χρησιμοποιείται για την ανανέωση της φερομόνης και επιστρέφεται ως η βέλτιστη λύση στον τερματισμό του αλγορίθμου.*
- *No storage:Οι νέες λύσεις σε κάθε επανάληψη χρησιμοποιούνται για την ανανέωση της φερομόνης και αγνοούνται για το υπόλοιπο της εκτέλεσης.Ο αλγόριθμος επιστρέφει το πιο πρόσφατο σύνολο λύσεων ως τη τελική λύση.*

Στον Πίνακα 3.3 παρουσιάζονται συνοπτικά τα αλγοριθμικά συστατικά των MOACO αλγόριθμων που συζητήσαμε παραπάνω.

Κεφάλαιο 4 Μεθοδολογία μοντελοποίησης του συστήματος

4.1 Σχεδίαση του ACO-πράκτορα

Καθώς οι λύσεις σε προβλήματα συνδυαστικής βελτιστοποίησης δημιουργούνται κατα βάση ακολουθώντας ένα επαναληπτικό αριθμό βημάτων ,μπορούμε να δούμε αυτήν την επαναλήπτική διαδικασία ως το μονοπάτι που ακολουθεί ένα τεχνητό μυρμήγκι εκκινώντας την διαδρομή του από την φωλιά(δηλαδή μια κενή λύση) και περπατώντας επαναληπτικά από το ένα βήμα στο άλλο καθώς προσπαθεί να φτάσει σε μια πλήρη και εφικτή λύση του προβλήματος.

Εξάγουμε το μοντέλο του ACO-πράκτορα του συστήματός μας σύμφωνα με την PAGE-περιγραφή που αναλύσαμε στην Ενότητα 2.2.Η PAGE-περιγραφή αναφέρεται στις Αντιλήψεις, Ενέργειες, Στόχους και Περιβάλλον ενός πράκτορα και περιγράφει τα πιο σημαντικά χαρακτηριστικά του.Εφόσον θεωρούμε τα φυσικά μυρμήγκια ως πράκτορες παραθέτουμε καταρχήν την PAGE-περιγραφή ενός πραγματικού μυρμηγκιού στον Πίνακα 4.1 .

Αντιλήψεις	*Συγκέντρωση φερομόνης στην τρέχουσα θέση του μυρμηγκιού->τροφή στην τρέχουσα θέση.*
Ενέργειες	*Περίπατος->Εναπόθεση φερομόνης.*
Στόχοι	*Εύρεση τροφής.*
Περιβάλλον	*Πραγματικός κόσμος.*

Πίνακας 4.1

Προκειμένου να μοντελοποιήσουμε τον τεχνητό πράκτορα μας ώστε να βρίσκει λύσεις σε προβλήματα συνδυαστικής βελτιστοποίησης αναλύουμε παρακάτω τα PAGE χαρακτηριστικά του βασιζόμενοι στην λειτουργία των αποικιών μυρμηγκιών της γενικής ACO μεθευρετικής διαδικασίας που περιγράψαμε στην Ενότητα 2.2.2 :

1. ***Περιβάλλλον**:Στο σύστημά μας οι τεχνητοί πράκτορες τοποθετούνται σε ένα γράφο.Κάθε κόμβος ή ακμή του συνιστά το συστατικό μιας λύσης ,ενώ οι κόμβοι συνδέονται μεταξύ τους ώστε να περιγραφεί η προσάρτηση του επόμενου συστατικού στην μερική λύση ενός πράκτορα.Υπάρχει επίσης και ένας ειδικός κόμβος ο οποίος αναπαριστά την κενή αρχική λύση και χρησιμεύει ώς το εκκινιτήριο σημείο του πλήθυσμού των πρακτόρων δηλαδή ως η φωλιά τους.*

   Καθώς δημιουργούν διαδρομές στο περιβάλλον οι πράκτορες κατασκευάζουν επαναληπτικά μια λύση προσαρτώντας τα διάφορα συστατικά – επιπλέον επικοινωνούν μεταξύ τους με ένα έμμεσο τρόπο κατα αναλογία με τα πραγματικά μυρμήγκια :κάθε ακμή του γράφου είναι σταθμισμένη με έναν αριθμό που αντιπροσωπεύει την τρέχουσα ποσότητα φερομόνης που έχει εναποθετηθεί εώς το σημείο αυτό στην ακμή.

   Επιπλέον το περιβάλλον παρέχει πληροφορία σχετικά με το κόστος κάθε βήματος ως πρός την ποιότητα της λύσης ,ώστε να εμποδίσει τους πράκτορες να εκτελούν τυχαία αναζήτηση στον χώρο του περιβάλλοντος.Για το λόγο αυτό σταθμίζουμε κάθε ακμή με έναν επιπλέον αριθμό που καλείται ευρετική τιμή.

Τέλος στο πλαίσιο των πολυπρακτορικών συστημάτων το ACO-περιβάλλον του συστήματος μας έχει τις εξής ιδιότητες:

Είναι μερικώς παρατηρίσιμο εφόσον κάθε πράκτορας έχει πρόσβαση στην πληροφορία φερομόνης και την ευρετική πληροφορία των ακμών που 'φεύγουν' από τον κόμβο στον οποίο βρίσκεται.

Επίσης είναι αιτιοκρατικό καθώς η αλλαγή των βαρών φερομόνης είναι ο μόνος τρόπος για να τροποποίησει ο πράκτορας το περιβάλλον.

Επιπλέον είναι μη επεισοδιακό εφόσον τα βάρη φερομόνης των ακμών μεταβάλλονται συνεχώς και η τιμή τους έχει σημαντική επίδραση στην μετακίνηση των πρακτόρων.

Είναι διακριτό καθώς τα συστατικά μιας λύσης σε προβλήματα συνδυαστικής βελτιστοποίησης είναι πεπερασμένου αριθμού.

Τέλος για τη στατικότητα του περιβάλλοντος θα πρέπει να λάβουμε υπόψη τη φύση του προβλήματος προς επίλυση.Στη συγκεκριμένη διπλωματική αντιμετωπίζουμε το στατικό JSP όπου τα δεδομένα εισόδου και οι περιορισμοί του πρόβλήματος δεν υπόκεινται σε αλλαγές κατα τη διάρκεια εκτέλεσης άρα έχουμε να κάνουμε με στατικό περιβάλλον.

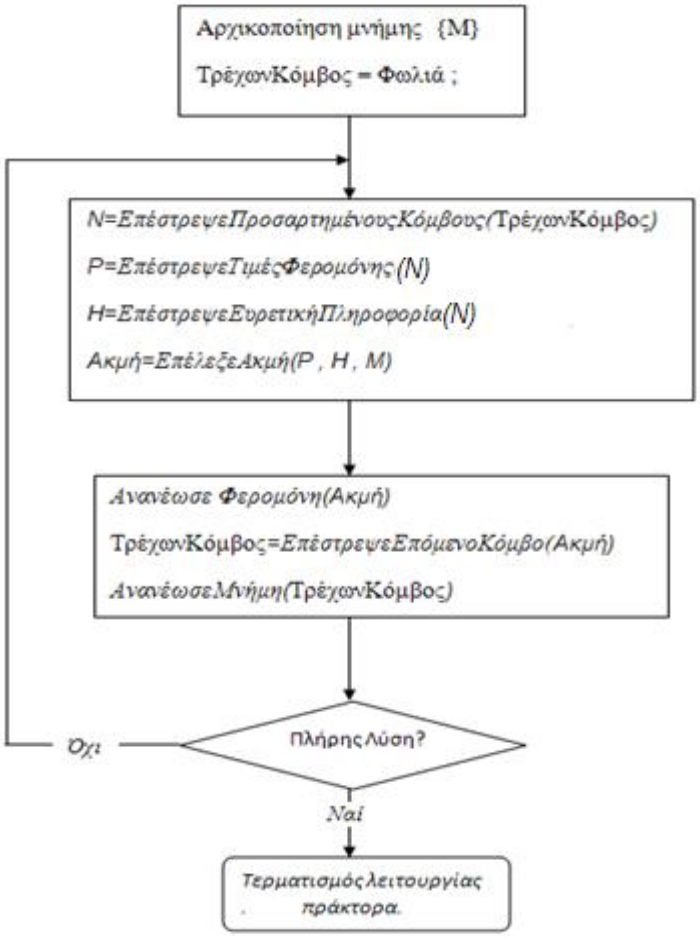
2. ***Αντιλήψεις:****Όπως και τα πραγματικά μυρμήγκια οι τεχνητοί πράκτορες αντιλαμβάνονται το περιβάλλον σε μια τοπική περιοχή γύρω από την τρέχουσα θέση τους πχ. έχουν πρόσβαση μόνο στην πληροφορία φερομόνης και ευρετική πληροφορία της ακμής που είναι προσκείμενη στον τρέχων κόμβο.*

3. ***Στόχοι:****Κάθε πράκτορας στοχεύει στην κατασκευή μιας ολοκληρωμένης και εφικτής λύσης διερχόμενος ενός ορισμένου αριθμού κόμβων.Σημειώνουμε ότι οι πράκτορες σε ατομικό επίπεδο δεν έχουν στόχο την κατασκευή υψηλής ποιότητας λύσεων.Επίσης το κόστος κάθε κίνησης συνάδει με την τιμή της ευρετικής πληροφορίας η οποία παρέχει στούς πράκτορες κατα την εκκίνηση της διαδικασίας αναζήτησης έναν αρχικό προσανατολισμό.Κατα την περαιτέρω ανάπτυξη της αναζήτησης όπου τα ίχνη φερομόνης σχηματίζονται , μειώνεται και η σημασία της ευρετικής πληροφορίας.Άρα κατα παρόμοιο τρόπο με μια πραγματική αποικία μυρμηγκιών η ικανότητα βελτιστοποίησης του συστήματος προκύπτει από την συλλογική δράση ολόκληρου του συνόλου των πρακτόρων.*

4. ***Ενέργειες:****Για την κατασκευή έγκυρων λύσεων κάθε πράκτορας εκτελεί μια σειρά ενεργειών:Καταρχάς εξετάζει τα βάρη φερομόνης και ευρετικής πληροφορίας κάθε ακμής προσκείμενης στη τωρινή του θέση και επιλέγει συγκεκριμένη ακμή είτε επειδή σταθμίζεται από υψηλή συγκέντρωση φερομόνης ή επειδή έχει καλή ευρετική τιμή.Κινείται στη συνέχεια ακολουθώντας την επιλεγμένη ακμή στον νεό κόμβο και ελλατώνει την τιμή φερομόνης της κατα μια σταθερή τιμή.Επιπρόσθετα κάθε πράκτορας χρειάζεται μια εσωτερική μνήμη των κόμβων που έχει επισκεφτεί ώστε να αποφύγει τον σχηματισμό κύκλων στον γράφο και να εγγυηθεί την κατασκευή έγκυρων λύσεων.*

Η λειτουργικότητα του τεχνητού πράκτορα περιγράφεται από το διάγραμμα ροής της Εικόνας 4.1. Σε κάθε επανάληψη ο πράκτορας αντιλαμβάνεται το περιβάλλον του μέσω των διαδικασιών ΕπέστρεψεΠροσαρτημένουςΚόμβους ,

ΕπέστρεψεΤιμέςΦερομόνης, ΕπέστρεψεΕυρετικήΠληροφορία.Τότε επιλέγει την επόμενη ακμή βάση των ακροαζόμενων τιμών και τις εσωτερικής του μνήμης,ένα βήμα αυτόνομο και κατα κάποιο τρόπο ευφυές.Τέλος ανανεώνει το βάρος φερομόνης στην επιλεγμένη ακμή και κινείται στον επόμενο κόμβο.Η διαδικασία επαναλαμβάνεται ωσότου παραχθεί μια πλήρης λύση.



Εικόνα 4.1:Λειτουργικότητα του ACO πράκτορα.

Αντιλήψεις	*Βάρη ευρετική πληροφορίας στις προσαρτημένες ακμές-> Βάρη φερομόνης στις προσαρτημένες ακμές.*
Ενέργειες	*Επιλογή επόμενης ακμής->Αύξηση φερομόνης στην επιλεγμένη ακμή->μετακίνηση στον επόμενο κόμβό.*
Στόχοι	*Κατασκευή πλήρων και έγκυρων λύσεων.*
Περιβάλλον	*Γράφος με τα συστατικά μιας λύσης.*

Πίνακας 4.2: PAGE-περιγραφή του ACO πράκτορα

Τέλος συνοψίζουμαι το μοντέλο του τεχνητού πράκτορά μας παραθέτωντας την PAGE-περιγραφή του στον Πίνακα 4.1.

Καθώς έχουμε ορίσει τα βασικά χαρακτηριστικά και την λειτουργικότητα του πράκτορά

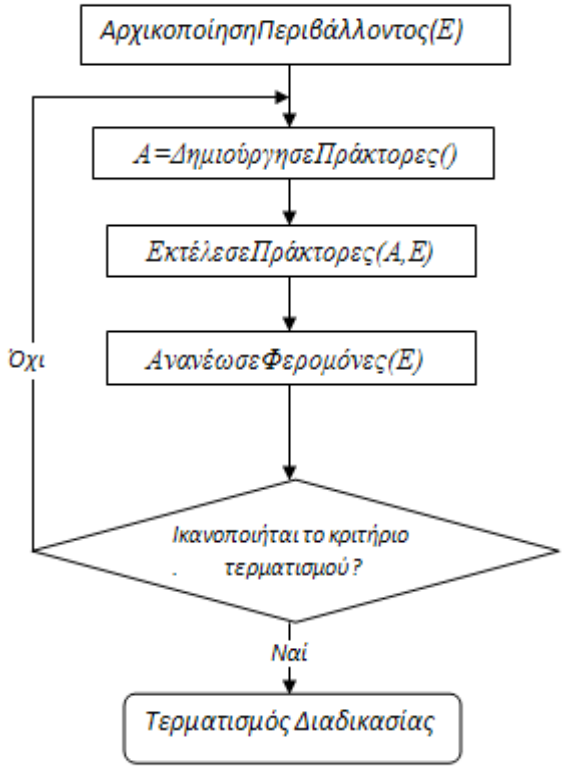
μας το επόμενο βήμα είναι να συνδιάσουμε τους πράκτορες σε μια κοινωνία.Θα πρέπει ωστόσο να σημειωθεί οτί το κύριο χαρακτηριστικό που οδηγεί την ικανότητα βελτιστοποίησης των πραγματικών αποικιών μυρμηγκιών,η στιγμεργία, δεν είναι παρούσα σε ένα τεχνητό περιβάλλον .Στον πραγματικό κόσμο τα μυρμήγκια που ακολουθούν την μικρότερη διαδρομή φτάνουν νωρίτερα στην πηγή τροφής και προσελκύουν περισσότερα μυρμήγκια να ακολουθήσουν την ελάχιστη διαδρομή, οδηγώντας έτσι σε υψηλότερη συγκέντρωση φερομόνης στα καλά μονοπάτια.

Στην περίπτωση των τεχνητών πρακτόρων η συγκεκριμένη αρχή δεν είναι εφαρμόσιμη ,για παράδειγμα η συγκέντρωση φερομόνης σε ένα καλό μονοπάτι μπορεί να είναι ίση με την φερομόνη σε ένα χειρότερο.Για τον λόγο αυτό είναι απαραίτητο να ανανεώνουμε την συγκέντρωση φερομόνης απο μόνοι μας αφότου κάθε πληθυσμός πρακτόρων έχει ολοκληρώσει τις εργασίες του. Υπάρχουν αρκετοί τρόποι για να το επιτύχουμε αυτό,όπως αναφέρθηκε και στην Ενότητα 2.2.2 ένας τρόπος είναι η αύξηση των βαρών φερομόνης στα μονοπάτια του πράκτορα που είχε την καλύτερη απόδοση από το σύνολο του πληθυσμού(Iteration-Best).Αναμένουμε έτσι η αναζήτηση για καλές λύσεις να επικεντρωθεί στην περιοχή των μονοπατιών που έχουν επιδείξει την καλύτερη απόδοση.

Εκτός από την ενίσχυση των καλών μονοπατιών είναι απαραίτητη και η μείωση των βαρών φερομόνης σε κάθε γενιά του πληθυσμού κατα μια μικρή τιμή.Η εξάτμιση της φερομόνης βοηθά τους πράκτορες να αποφύγουν τον πρόωρο εγλωβισμό σε μια μέτρια λύση(βλ. Ενότητα 2.2.2).

Οι διαδικασίες ανανέωσης της φερομόνης θα αναλυθούν αναλυτικά στην επόμενη ενότητα όπου θα γίνει η σύνδεση του συστήματος πρακτόρων με το πρόβλημα μας.

Ένα γενικό διάγραμμα του προτεινόμενου συστήματος πρακτόρων που παρουσιάσαμε στην παρούσα ενότητα φαίνεται στην Εικόνα 4.2.Αρχικά γίνεται η αρχικοποίηση του περιβάλλοντος Ε ο ορισμός δηλαδή του γράφου που αναπαριστά το πρόβλημά μας.Κατόπιν καλείται η μέθοδος ΔημιούργησεΠράκτορες η οποία αρχικοποιεί τον αρχικό πληθυσμό των πρακτόρων και στην συνέχεια καλούμε την συνάρτηση ΕκτέλεσεΠράκτορες η οποία έκτελεί τις ενέργειες που σχετίζονται με την λειτουργικότητα κάθε πράκτορα που περιγράψαμε προηγουμένως(Εικόνα 4.1). Ακολουθεί η ανανέωση των φερομόνων καλώντας την μέθοδο ΑνανέωσεΦερομόνες βαση κάποιου κανόνα ολικής ανανέωσης που θα περιγράψουμε στη συνέχεια.Εάν ικανοποιείται το κριτήριο τερματισμού το σύστημα τερματίζει την εκτελεσή του διαφορετικά επαναλαμβάνει τα παραπάνω βήματα με τις ανανεωμένες τιμές.



Εικόνα 4.2 : Γενικό διάγραμμα του προτεινόμενου συστήματος πρακτόρων.

4.2 Εφαρμογή του Συστήματος πρακτόρων στο JSP

Στα πλαίσια αυτής της διπλωματικής επιλύουμε το πρόβλημα του Job-Shop Scheduling,JSP ,το οποίο παρουσιάστηκε αναλυτικά στο 3° κεφάλαιο.Θεωρούμε δύο αντικειμενικά κριτήρια ,ονομαστικά 1) το makespan,C_{max} και 2) τον συνολικό χρόνο βραδύτερης περάτωσης,T_d τα οποία επιδιώκουμε να ελαχιστοποίησουμε ξεχωριστά σε κάθε περίπτωση.Από μαθηματικής πλευράς το προβλήμα του JSP θεωρώντας τα παραπάνω κριτήρια μοντελοποιήθηκε αναλυτικά στην Ενότητα 3.1.Το σύστημα πρακτόρων εφαρμόζεται στο JSP στοχεύοντας σε κάθε κριτήριο ξεχωριστά.

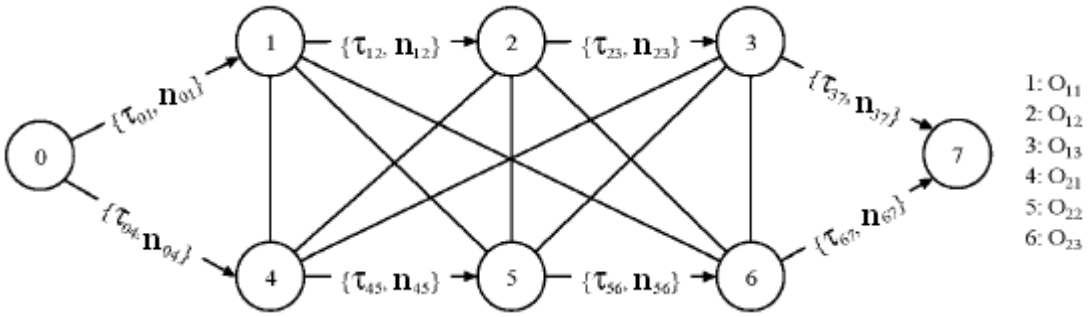
Σύμφωνα με την συζήτηση που αναπτύχθηκε στην Ενότητα 2.2.2 σελ 22 ,το σύστημα πρακτόρων εμπνευσμένο από αποικίες μυρμηγκιών μπορεί να εφαρμοστεί σε οποιοδήποτε πρόβλημα συνδυαστικής βελτιστοποίησης ,συμπεριλαμβανομένου του JSP ,αρκεί αυτό να μπορεί να ορίσει τα παρακάτω σημεία:

1. *Μια κατάλληλη **αναπαράσταση του προβλήματος** που να μπορει να αποδώσει το πρόβλημα σε ένα γράφημα (construction graph) και την λύση του με μια διαδρομή πάνω στο γράφημα αυτό.Η λύση πρέπει να προκύπτει από ένα στοχαστικό κανόνα μετάβασης ο οποίος θα χρησιμοποιεί την ποσότητα φερομόνης για την διαμόρφωση των πιθανότητων μετάβασης και μια ευρετική συνάρτηση η οποία θα φανερώνει την ελκυστικότητα της κάθε ακμής.*
2. *Την **ευρετική συνάρτηση** $\eta(\cdot)$ για την ελκυστικότητα των ακμών.*
3. *Μια μέθοδο **ικανοποίησης των περιορισμών** του προβλήματος, ώστε οι λύσεις που προκύπτουν να είναι εφικτές.*
4. *Έναν **κανόνα ανανέωσης της φερομόνης** $\tau(\cdot)$ των ακμών του γράφου.*
5. *Ένα **στοχαστικό κανόνα μετάβασης** .*

1. ***Αναπαράσταση του περιβάλλοντος του προβλήματος***:

Προκειμένου να εφαρμόσουμε το σύστημα πρακτόρων στο JSP είναι απαραίτητη η απόδοση του προβλήματος σε ένα γράφο:

Για την αναπαράσταση του περιβάλλοντος του JSP υιοθετήσαμε την μεθοδολογία των Colorni et al.'s (1993) και των συνεργατών του οι οποίοι πρότειναν την σχεδίαση του περιβάλλοντος του JSP για την εφαρμογή σε αυτό ενός γενικού ACO αλγόριθμου Η αναπαράσταση που περιγράφουμε διαφέρει ελαφρά από αυτή της Ενότητας 3.2.1 και έχει ως εξής : Το πρόβλημα του JSP με ένα σύνολο n εργασιών J, ένα σύνολο m μηχανών M και ένα σύνολο διεργασιών O ,όπου $N=|O|$ το πλήθος των διεργασιών ,μπορεί να αναπαρασταθεί σε ένα κατευθυνόμενο , σταθμισμένο γράφο $G=(U , E)$,όπου $U =O \quad \{ O_0 \}$;(Ο κόμβος-πηγή O_0 χρησιμοποιείται για να καθορίσει πια εργασία θα εκτελεστεί πρώτη στην περίπτωση όπου αρκετές εργασίες απαιτούν η πρώτη τους διεργασία να γίνει στην ίδια μηχανή) και $E =\{ (O_0, O_{i1})\} \quad \{ (O_{ij} , O_{kl}): O_{ij} , O_{kl} \quad O \}$; ,δηλαδή το E περιλαμβάνει α) το σύνολο των ακμών που συνδέουν την πηγή O_0 με την πρώτη διεργασία κάθε εργασίας O_{i1} , $Ji \quad J$ και β) το σύνολο των ακμών που συνδέουν πλήρως τους κόμβους του O εκτός από τους κόμβους που ανήκουν στην ίδια εργασία.Οι διεργασίες που ανήκουν σε διαφορετικές εργασίες συνδέονται με μη κατευθυνόμενες ακμές (Εικόνα 4.3),ενώ οι διεργασίες που ανήκουν στην ίδια εργασία συνδέονται με τόξα (κατευθυνόμενες ακμές) και έχουν προσανατολισμό ο οποίος δηλώνει την αλληλουχία των διεργασιών μέσα στην εργασία.Υπάρχουν λοιπόν $N+1$ κόμβοι και [———] $+n$ ακμές , όπου κάθε κόμβος ,εκτός του O_0 , αντιπροσωπεύει την επεξεργασία μιας εργασίας από μια μηχανή (διεργασία) και όλοι οι κόμβοι είναι συνδεδεμένοι αν ζεύγη εκτός του O_0 ,που συνδέεται μόνο με τις πρώτες διεργασίες κάθε εργασίας.Τέλος κάθε ακμή είναι σταθμισμένη από ένα ζεύγος αριθμών $\{ \tau_{ij} , n_{ij} \}$,όπου η τιμή τ_{ij} αντιπροσωπεύει το επίπεδο της φερομόνης στην ακμή $i \rightarrow j$ και η τιμή n_{ij} (ευρετική πληροφορία) αντίστοιχα την ελκυστικότητα της μετάβασης $i \rightarrow j$,την οποία υπολογίζουμε βάση κάποιου ευρετικού κανόνα (dispatching rule),όπως θα δούμε παρακάτω.Στη συνέχεια της εργασίας θα αναφερόμαστε στους κόμβους του γράφου και τις διεργασίες των εργασιών ταυτόσημη έννοια.



Εικόνα 4.3:Ο γράφος για ένα JSP 2x3 σύμφωνα με την αναπαράσταση των Colorni et al (1993).

1. ***Ευρετική συνάρτηση***

Η ευρετική συνάρτηση καθορίζει την ελκυστικότητα n_{ij} να ανήκει η ακμή (i,j) στην τελική λύση σύμφωνα με ένα απλό και άπληστο κριτήριο, βάση δηλαδή ενός κανόνα προτεραιότητας(dispatching rule).Καθώς η ευρετική πληροφορία διαφέρει ανάλογα με το κριτήριο που ελαχιστοποιούμε ο ορισμός της θα γίνει στο επόμενο κεφάλαιο όπου θα ασχοληθούμε με την ανάλυση και σχεδίαση του συστήματος.

2. Ικανοποίηση περιορισμών

Όπως αναφέραμε προηγούμενως κάθε πράκτορας χρειάζεται μια εσωτερική μνήμη των κόμβων που έχει επισκεφτεί ώστε να αποφύγει τον σχηματισμό κύκλων στον γράφο και να εγγυηθεί την κατασκευή έγκυρων λύσεων.Η μέθοδος ΑνανέωσηΦερομόνη() χρησιμοποιεί συγκεκριμένη δομή και επιστρέφει μια λίστα **N** *με τους επιτρεπόμενους κόμβους στους οποίους έχει δικαίωμα να μεταβεί ο πράκτορας σε κάθε βήμα του ταξιδιού του:*

Για να διατηρήσουμε την εφικτότητα των λύσων επιτρέπουμε στους πράκτορες να επισκέπτονται ορισμένους κόμβους κάθε φορά (Colorni et al.'s ,1993),από μια λίστα επιτρεπόμενων κόμβων **N** *(·). Καταρχήν, δεν επιτρέπουμε να επισκεφτούν κάποιον κόμβο τον οποίο έχουν ήδη επισκεφτεί. Επιπλέον, αν ο κόμβος που επιλέγουν δεν είναι ο τελευταίος στην σειρά της εργασίας, τότε και μόνο τότε στη λίστα των επιτρεπόμενων κόμβων προστίθεται και ο κόμβος της αμέσως επόμενης διεργασίας της εργασίας. Επίσης ένας κόμβος που έχει επιλεχθεί ,εφόσον δεν θα τον επισκεφτούμε ξανά ,διαγράφεται από τη λίστα* **N** *(·) και προστίθεται σε μια λίστα tabu η οποία θα περιέχει το τελικό χρονοπρόγραμμα δηλαδή την πλήρη λύση του κάθε πράκτορα . Για παράδειγμα στην Εικονα 4.3, οι επιτρεπόμενοι κόμβοι για τη χρονική στιγμή t=0 είναι: $\{O_{11}, O_{21}\}$ (πχ. οι πρώτες διεργασίες όλων των εργασιών).*

3. Κανόνας ανανέωσης της φερομόνης

.

Μετά το πέρας των διαδρομών όλου του πληθυσμού των πρακτόρων καλείται η μέθοδος ΑνανέωσεΦερομόνες() (Εικόνα 4.2) η οποία ανανεώνει την φερομόνη βάση κάποιου κανόνα ολικής ανανέωσης.Αντίστοιχα κατα την κατασκευή μιας λύσης ο πράκτορας καλεί την συνάρτηση ΑνανέωσεΦερομόνη() ώστε να ανανεώσει την φερομόνη στην τελευταία ακμή που διέσχισε βάση κάποιου κανόνα τοπικής ανανέωσης.Οι κανόνες ολικής και τοπικής ανανέωσης της φερομόνης και η λειτουργικότητά τους περιγράφονται παρακάτω:

Υιοθετούμε τους κανόνες ανανέωσης φερομόνης του ACS αλγόριθμου(Ant Colony System,Dorigo & Gambardella, 1997).Μόνο ο πράκτορας που πετυχαίνει την καλύτερη επίδοση σε κάθε κύκλο έχει το δικαίωμα να εκκρίνει φερομόνη στις ακμές που διέσχισε.Επομένως η αναζήτηση των πρακτόρων κατευθύνεται ώστε να αναζητήσουν μονοπάτια στην περιοχή της μέχρι τώρα βέλτιστης διαδρομής(πχ. της διαδρομής που αντιστοιχεί στο ελάχιστο makespan, C_{max} έαν ελαχιστοποίουμε το makespan).Επομένως σύμφωνα με τον ACS εφαρμόζεται ένας ***κανόνας ολικής ανανέωσης της φερομόνης*** *μόνο στις ακμές που ανήκουν στο βέλτιστο μονοπάτι που προκύπτει στο τέλος κάθε κύκλου.Ο νέος κανόνας ανανέωσης είναι:*

$$\tau_{ij}(t+n) \leftarrow (1-\alpha) \cdot \tau_{ij}(t) + \alpha \cdot \Delta \tau_{ij}(t,t+n) \quad , \forall (i,j) \in T^{bs} \qquad (4.1)$$

όπου i,j οι διεργασίες που ανήκουν στο T^{bs} ,το βέλτιστο μονοπάτι της επανάληψης ,

α είναι μια παράμετρος που καθορίζει το ρυθμό εξάτμισης της φερομόνης και $\Delta\tau_{ij}(t,t+n) = \frac{\quad}{\quad}$, ,με L^{bs} το κόστος της βέλτιστης διαδρομής (makespan,total tardiness).

Καθώς ο πράκτορας k επιλέγει την διεργασία j ενώ βρίσκεται στην διεργασία i η ποσότητα της φερομόνης στην ακμή (i,j) μεταβάλλεται σύμφωνα με τον **κανόνα τοπικής ανανέωσης της φερομόνης** του ACS:

$$\tau_{ij}(t+1) \leftarrow (1-\rho)\cdot\tau_{ij}(t) + \rho\cdot\tau_0 \qquad (4.2)$$

Η λειτουργία αυτού του κανόνα αποσκοπεί στο να μεταβάλλει δυναμικά την ελκυστικότητα των ακμών ,ώστε η αναζήτηση των πρακτόρων να μη περιορίζεται σε μια στενή περιοχή γύρω από τη βέλτιστη λύση της προηγούμενης επανάληψης.Η τιμή της παραμέτρου τ_0 είναι η αρχική ποσότητα εκκίνησης της φερομόνης σε όλες τις ακμές του γραφήματος και στο [39] έχει **βελτιστοποιηθεί πειραματικά** στην τιμή $\tau_0 = (1/N * L_{mn})$,όπου n ο αριθμός των κόμβων του κατασκευαστικού γράφου(στην περίπτωση αυτή ο αριθμός των πόλεων) και L_{mn} το μήκος της διαδρομής που δίνει ο αλγόριθμος του κοντινότερου γείτονα(Nearest neighbor heuristic) για το πρόβλημα του Πλανώδιου Πωλητή(TSP).Κατα αναλογία θέσαμε αυτή την τιμή ίση με $\tau_0 = (1/N * L_{SPT})^{-1}$,όπου N ο συνολικός αριθμός διεργασιών του προβλήματος και L_{SPT} το makespan του χρονοπρογράμματος που προκύπτει από την εφαρμογή του SPT(Shortest Processing Time) ευρετικού κανόνα(dispatching rule) εάν ελαχιστοποιούμε το makespan.Ανάλογα θα είναι $\tau_0 = (1/N * T_{EDD})^{-1}$ για το κριτήριο της βραδυτερης περάτωσης, όπου T_{EDD} η total tardiness του χρονοπρογράμματος που προκύπτει από την εφαρμογή του EDD κανόνα(Merkle & Middendorf,2000).

Η συγκεκριμένη τιμή ,ανεξάρτητά του κριτηρίου που ελαχιστοποιούμε, παρέχει ένα ικανοποιητικό επιπεδό αρχικής φερομόνης λαμβάνοντας υπόψη τις παραμέτρους του προβλήματος:Η τ_0 ισορροπεί σε μια τιμή επαρκώς μικρή και επαρκώς μεγάλη ,ούτως ώστε η αύξηση της φερομόνης στα καλά μονοπάτια να έχει νόημα ,δηλαδή να οδηγεί μακροπρόθεσμα στην εύρεση καλών λύσεων.Θυμίζουμε ότι η αύξηση της φερομόνης στην ακμή μιας καλής λύσης σύμφωνα με τον κανόνα ολικής ανανέωσης είναι της τάξης του $\frac{\quad}{\quad}$,όπου T^{bs} το κόστος της διαδρομής στη οποία ανήκει η ακμή. Μια πολύ μικρή τιμή της τ_0 θα ευνούσε τα αρχικά μονοπάτια που θα κατασκεύαζαν οι πράκτορες στις πρώτες επαναλήψεις του αλγόριθμου τα οποία βεβαίως δεν θα είναι βέλτιστα.Αντίστοιχα μια αρκετά μεγάλη τιμή για την παράμετρο θα εμπόδιζε να ξεχωρίσουν οι καλές λύσεις στην διάρκεια της εκτέλεσης, εφόσον η προσάθροιση της φερομόνης θα ήταν αμελητέα σε σχέση με την τιμή της τ_0.Κατεπέκταση ο κανόνας μετάβασης θα επέλεγε την επόμενη διεργασία οδηγούμενος ουσιαστικά από την ευρετική της πληροφορία.

4. Κανόνας μετάβασης

Σύμφωνα με το διάγραμμα ροής της Εικόνας 4.2 η μέθοδος ΕπέλεξεΑκμή() επιλέγει την επόμενη ακμή πάνω στην οποία θα κινηθεί ο πράκτορας για να μεταβεί στην επόμενη διεργασία.Είναι αναγκαίος λοιπόν ένας κανόνας μετάβασης βάση του

οποίου θα επιλέξει ο πράκτορας την επόμενη διεργασία με συγκεκριμένη πιθανότητα.

*Ο **κανόνας μετάβασης** επιτρέπει ρητώς την περαίτερω διερεύνηση .Σύμφωνα με το νέο κανόνα που ορίζει ο ACS αλγόριθμος ο πράκτορας k που βρίσκεται στον κόμβο-διεργασία i επιλέγει τον κόμβο-διεργασία j, με j*

(4.3)

,όπου q μια τυχαία μεταβλητή,ομοιόμορφα κατανεμημένη στο $[0,1)$, $q_0 \in [0,1)$ παράμετρος καθοριζόμενη από τον χρήστη και $J \in \mathbf{N}(s^p)^k$ μια διεργασία η οποία επιλέγεται τυχαία βάση του κανόνα αναλογίας·

Ο κανόνας αναλογίας είναι η πιθανότητα ένας πράκτορας k να μεταβεί από μια διεργασία i στην διεργασία j στη διάρκεια της επανάληψης t. Ο κανόνας αυτός καλείται «κανόνας τυχαίας αναλογικής μετάβασης» (random proportional transition rule) και η αντίστοιχη πιθανότητα ισούται με:

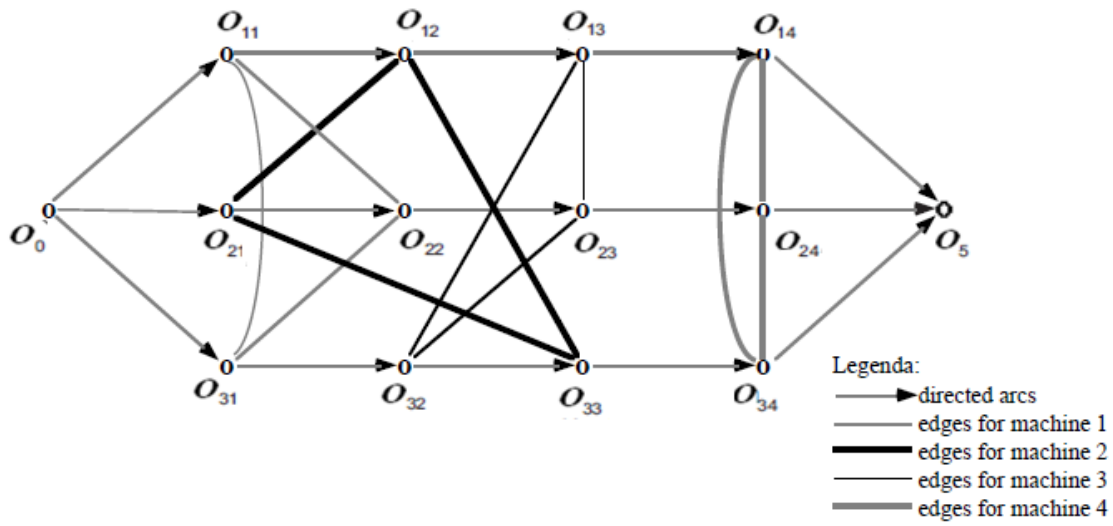
,όπου $\tau_{ij}(t)$ το ποσό της φερομόνης στην ακμή (i,j) στην επανάληψη t.

n_{ij} ,η a-priori ευρετική τιμή ,

α και β δύο παράμετροι οι οποίες ελέγχουν τη σχετική βαρύτητα της φερομόνης ως προς την ορατότητα.

$\mathbf{N}(s^p)^k$ το σύνολο των υποψήφιων διεργασιών προς μετάβαση,για τον k-οστο πράκτορα, βάση της τρέχουσας μερικής λύσης s^p .

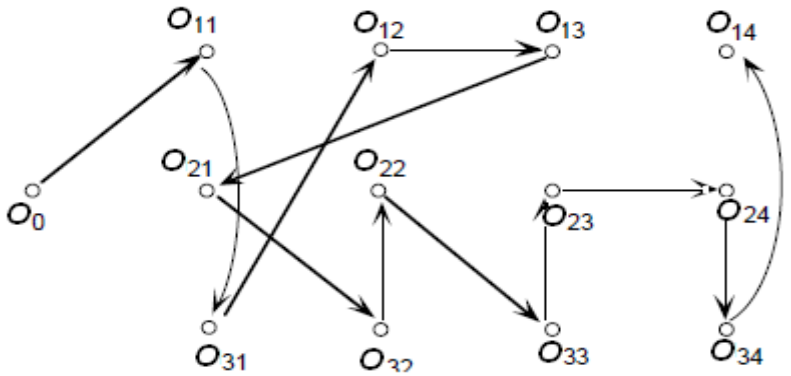
Επομένως όταν $q \leq q_0$ το σύστημα εκμεταλλεύεται τη γνώση που έχει συσσωρευθεί για το πρόβλημα , δηλαδή τα ίχνη της φερομόνης πάνω στις ακμές . Έαν $q > q_0$ τότε ο αλγόριθμος ευνοεί την εξερεύνηση νέων μονοπατιών. Προσαρμόζοντας την παράμετρο q_0 κοντά στη μονάδα αναγκάζουμε το σύστημα να επιλέξει την τοπικά βέλτιστη λύση, ενώ προσαρμόζοντας την κοντά στο 0 εξερευνούμε όλες τις εναλλακτικές.



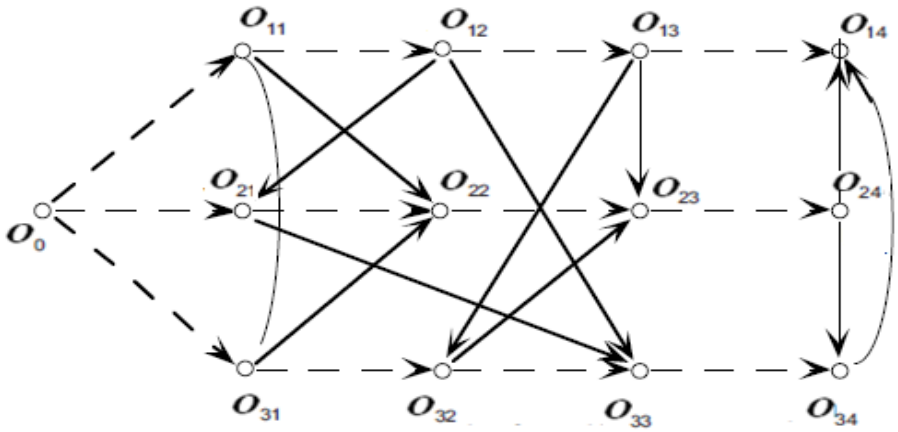
Εικόνα 4.4: Διαζευκτικός γράφος ενός 3x4 JSP(Colorni et al ,2003).

Στο τέλος ενός κύκλου κάθε πράκτορας έχει ολοκληρώσει μια διαδρομή στον γράφο , έχοντας επισκεφτεί κάθε κόμβο ακριβώς μια φορά ,σχηματίζοντας έτσι μια ακολουθία από διεργασίες η οποία ορίζει και την ακολουθία των διεργασιών σε μια μηχανή(Siang Lyn ,2005) .Σε αυτό το σημείο μπορούμε να υπολογίσουμε το μήκος του κρίσιμου μονοπατιού στον κατευθυνόμενο γράφο και άρα να υπολογίσουμε την τιμή της αντικειμενικής συνάρτησης για τη λύση που προκύπτει από κάθε πράκτορα.Ένα οπτικό παράδειγμα φαίνεται παραπάνω θεωρώντας τον διαζευκτικό γράφο της Εικόνας 4.4(Colorni et al ,2003).

Μια πλήρης λύση αντιστοιχεί σε μια πλήρη διαδρομή στον γράφο (Εικόνα 4.4α),όπου η ακολουθία των διεργασιών είναι η O_{11} O_{31} O_{12} O_{13} O_{21} O_{32} O_{22} O_{33} O_{23} O_{24} O_{34} O_{14}. *'Φιλτράρωντας' της διεργασίες που ανήκουν στη μηχανή 1* (*machine* 1) *παίρνουμε την ακολουθία* O_{11}- O_{31}- O_{22} *,αντίστοιχα για τη μηχανή 2* O_{12}- O_{21}- O_{33} *κ.ο.κ. Τέλος εάν θέλαμε να συσχετίσουμε την αναπαράσταση της λύσης που προκύπτει από το σύστημα μας(Εικόνα 4.4α) με την κλασσική αναπαράσταση διαζευκτικού γράφου για το JSP που παρουσιάσαμε στην Ενότητα 3.2.1,αναθέτωντας τις παραπάνω ακολουθίες στον διαζευκτικό γράφο της Εικόνας 4.4 προκύπτει το γράφημα της Εικόνας 4.4β που αντιστοιχεί στην απεικόνιση διαζευκτικού γράφου μιας πλήρους και έγκυρης λύσης .*



Εικόνα 4.4α:Παράδειγμα της διαδρομής(πλήρους λύσης) ενός πράκτορα στον γράφο (Colorni et al ,2003).



Εικόνα 4.4β :Αναπαράσταση πλήρους λύσης στον διαζευκτικό γράφο με πλήρη επιλογή των μη κατευθυνόμενων ακμών Colorni et al ,2003).

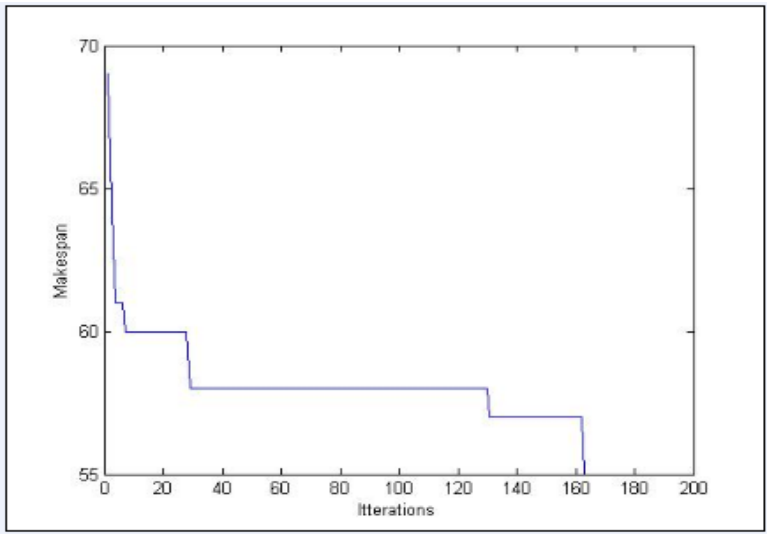
4.3 Κωδικοποίηση λύσεων και έξοδος του συστήματος

Έστω το JSP μεγέθους {n x m}.Υπάρχουν δυό κυριώς τρόποι να αναπαρασταθούν οι λύσεις στο JSP(Yamada,1992).Ο πρώτος τρόπος είναι η χρήση του διαγράμματος Gantt,ώστε να καθοριστούν οι χρόνοι ολοκλήρωσης για όλες τις εργασίες.Ο δεύτερος τρόπος είναι η χρήση του disjunctive γράφου,ώστε να καθοριστεί η σειρά προτεραιοτήτων για κάθε ζευγάρι εργασιών σε κάθε μηχανή.Και στις δύο περιπτώσεις η λύση κωδικοποιήται ,όπως αναφέραμε παραπάνω, ως μια ακολουθία διεργασιών όπου διαφέρει ο συμβολισμός τους.Επιλέξαμε απεικόνιση των λύσεων βάση του διαγράμματος Gantt,όπου κάθε εργασία αναπαριστάται από μοναδικό αριθμό στο διάστημα [1, n x m].Η κωδικοποίηση των λύσεων θα περιγραφεί πιο αναλυτικά στο επόμενο κεφάλαιο.

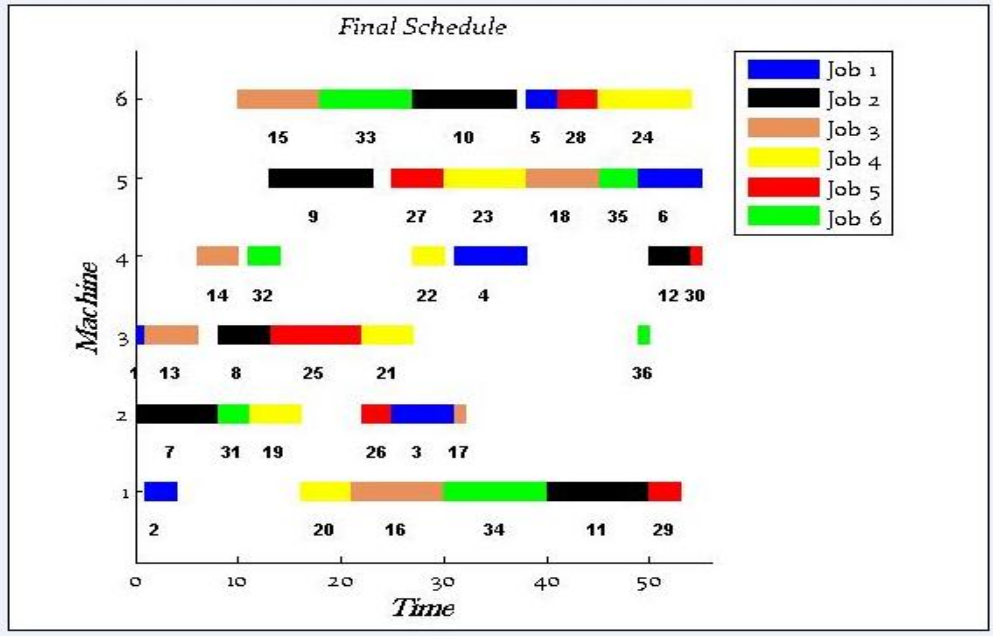
Η έξοδος του συστήματος περιλαμβάνει :

- *την τελική βέλτιστη λύση που βρέθηκε από το σύστημα ,εκείνη τη λύση δηλαδή που ελαχιστοποιεί το εκάστοτε κριτήριο που μας ενδιαφέρει.*

- *Ένα πίνακα με την βέλτιστη λύση που βρίσκει ο πληθυσμός των πρακτόρων σε κάθε κύκλο ,ο οποίος χρησιμεύει για να δείξουμε την ταχύτητα σύγκλισης του συστήματος στη τελική βέλτιστη λύση.Ένα παράδειγμα φαίνεται παρακάτω όπου το διάγραμμα σύγκλισης του σύστήματος με στόχο το makespan για το Ft06 στιγμιότυπο.Όπως φαίνεται το σύστημα συγκλίνει σε μια τιμή 55 υστέρα από 200 κύκλους που είναι και η βέλτιστη θεωρητική τιμή για το συγκεκριμένο στιγμιότυπο.*



- *Τέλος για τις ανάγκες της υλοποίησης δημιουργήθηκε μια διεπαφή η οποία αναπαριστά το διάγραμμα Gantt ενός χρονοπρογράμματος ώστε να παρέχεται στον χρήστη μια οπτική επαφή με το αποτέλεσμα.Παρακάτω παρατίθεται το διάγραμμα Gantt της τελικής βέλτιστης λύσης για το Ft06 που βρέθηκε από το σύστημα με makespan 55.Ο αναγνώστης μπορεί να παρατηρήσει ότι η διεργασία 30 (η 6η διεργασία της 5ης εργασίας) ευθύνεται για την τελική τιμή του makespan,με χρόνο ολοκλήρωσης 55 μονάδων χρόνου.*



Φυσικά το σύστημα μας μπορεί να εκτυπώσει οτιδήποτε άλλο του ζητηθεί,όπως τα διαγράμματα σύγκλισης του βάση διαφόρων συνδυασμών των παραμέτρων του,τους χρόνους σύγκλισης στη βέλτιστη λύση του σε κάθε κύκλο κλπ.

4.4 Μοντελοποίηση του συστήματος για πολυκριτηριακή βελτιστοποίηση.

Το σύστημα πρακτόρων για την περίπτωση του πολυκριτηριακού JSP που ορίσαμε στην ενότητα 3.1 αποτελεί ουσιαστικά προέκταση του συστήματος για την single-objective περίπτωση που παρουσιάσαμε στις προηγούμενες ενότητες.Όπως αναφέρθηκε στην Ενότητα 3.6.3.1 η σχεδίαση ενός ACO πολυκριτηριακού αλγόριθμου αποτελεί συγχώνευση του single objective αλγόριθμου στον οποίο βασίζεται η δομή του με τις διάφορες παραλλαγές των αλγοριθμικών του συστατικών.

Πρός αυτή την κατεύθυνση το σύστημα μας έχει την ίδια αρχιτεκτονική με το single objective σύστημα αλλά με διαφορετική υλοποίηση των δομών του όπου κρίνεται αναγκαίο,ονομαστικά οι δομές του όπως ορίστηκαν και για το single objective συστήμα:

- ***Λειτουργικότητα Πράκτορα***
- ***Αναπαράσταση του περιβάλλοντος***
- ***Ευρετική συνάρτηση και Πληροφορία φερομόνης***
- ***Κανόνες ανανέωσης της φερομόνης***
- ***Κανόνας μετάβασης***

Το σύστημα χρησιμοποιεί επιπλέον δυό δομές απαραίτητες λόγω της πολυκριτηριακής φύσης του προβλήματος:

- ***Αξιολόγηση Pareto λύσεων***
- ***Αρχείο Pareto***

Σχεδίαση της λειτουργικότητας του πράκτορα

Η λειτουργικότητα του πράκτορα είναι ίδια με αυτή στο μονοκριτηριακό σύστημα που παρουσιάσαμε στις προηγούμενες ενότητες,η κατασκευή δηλαδή πλήρων και έγκυρων λύσεων.Η ουσιαστική διαφορά ωστόσο με την μονοκριτηριακή περίπτωση είναι ο τρόπος με τον οποία ο πράκτορας επίλέγει το επόμενο συστατικό λύσης οδηγούμενος από τους κανόνες μετάβασης και ανανέωσης της φερομόνης,όπως θα δούμε παρακάτω.

Αναπαράσταση Περιβάλλοντος

Οι πράκτορες κινούνται στο ίδιο περιβάλλον που περιγράψαμε στην Ενότητα 4.2.

Ευρετική συνάρτηση και Πληροφορία φερομόνης

Χρησιμοποιούμε έναν μοναδικό πληθυσμό πρακτόρων(κατα αντιστοιχία με μια μοναδική αποικία μυρμηγκιών) για να ελαχιστοποιήσουμε ταυτόχρονα όλα τα κριτήρια και κατεπέκταση έναν μοναδικό πίνακα φερομόνης $[\tau_{ij}]$. *Κατα αυτόν τον τρόπο οι αντικειμενικές μοιράζονται την ίδια πληροφορία φερομόνης, άρα η γνώση για καλές λύσεις είναι εξίσου σημαντική για όλες τις αντικειμενικές συναρτήσεις κατα τη διαμόρφωση του μετώπου Pareto.*

Η επιλογή μας δικαιολογείται ως εξής :Στην πλειοψηφία των περιπτώσεων η χρήση ενός μοναδικού πίνακα φερομόνης εφαρμόζεται κυρίως σε non-Pareto based

αλγόριθμους οι οποίοι σχεδιάζονται για να συγκλίνουν σε μια συγκεκριμένη περιοχή του Pareto μετώπου.Εφόσον λοιπόν οι αλγόριθμοι αυτοί δεν χρειάζεται να αποθηκεύουν κατανεμημένη(diverse) πληροφορία ,η οποία θα είναι συγκεκριμένη για κάθε αντικειμενική ,ανανεώνουν και ελλατώνουν την ποσότητα φερομόνης κατα παρόμοιο τρόπο με τους single objective ACO αλγόριθμους και άρα η επιλογή ενός μοναδικού πίνακα φερομόνης φαίνεται λογική(Angus & Woodward,2009).Από την άλλη εφόσον στοχεύουμε στην εύρεση ενός άριστου και καλά κατανεμημένου Pareto μετώπου,μπορούμε να οδηγήσουμε την κατασκευή των λύσεων ως προς κάθε αντικειμενική βάση δύο πινάκων ευρετικής πληροφορίας n_{ij}^{1} *και* n_{ij}^{2} *οι οποίοι αντιπροσωπεύουν την ελκυστικότητα των ακμών για κάθε αντικειμενική ξεχωριστά ,αντί πολλαπλών πινάκων φερομόνης.Παρέχουμε βάρη στην ευρετική πληρόφορία με κατάλληλο τρόπο για κάθε πράκτορα ,ανάλογα σε πιο κριτήριο θέλουμε να δώσουμε μεγαλύτερη βαρύτητα (όπως θα δούμε παρακάτω),πολώνοντας την αναζήτησή του σε διαφορετικές περιοχές του χώρου αναζήτητης.Με άλλα λόγια προσπαθούμε να επιτύγχουμε διαφοροποιήση του Pareto μετώπου μέσω της ευρετικής πληροφορίας αντί της πληροφορίας φερομόνης, διατηρώντας παράλληλα έναν κοινό πίνακα φερομόνης.*

Επιπλέον η μείωση στη μνήμη που σχετίζεται με την χρήση ενός αντί πολλαπλών πινάκων φερομόνης είναι ένας παράγοντας που κάναμε την συγκεκριμένη επιλογή:Η ανανέωση (ολική και τοπική) της φερομόνης σε έναν αντί για δύο πίνακες,ειδικά σε προβλήματα μεγέθους μεγαλύτερα-ίσα των 100 διεργασιών μειώνουν τον χρόνο εκτέλεσης του συστήματος δραστικά.Τέλος στη χρήση διαφορετικών πινάκων ευρετικής πληροφοριάς για κάθε αντικειμενική συνάρτηση μας οδήγησε και το γεγονός ότι η ευρετική πληροφορία ορίζεται διαφορετικά για κάθε κριτήριο που εξετάζουμε στο JSP,ονομαστικά το makespan και ο συνολικός χρόνος βραδύτερης περάτωσης και άρα η συγχωνευσή τους σε έναν πίνακα δεν θα είχε πρακτική σημασία.

Κανόνες ανανέωσης της φερομόνης

Και πάλι εφαρμόζουμε ολική και τοπική ανανέωση της φερομόνης. Οι διαφορά ωστόσο του πολυκριτηριακού συστήματος με το μονοκριτηριακό κατα τη ανανέωση της φερομόνης ,διαφέρει στα εξής δυό σημεία :1^{ov} το σύστημα δίνει σαν έξοδο ένα σύνολο από Pareto λύσεις παρά μια βέλτιστη λύση σε κάθε κύκλο άρα είναι αναγκαια η χρήση ενός ***Αρχείου Pareto****(βλ ενότητα 3.6.3.1.6) όπου θα διατηρούμε της μη κυριαρχούμενες λύσεις .2^{ov} είναι απαράιτητος ένα μηχανισμος* ***Αξιολόγησης των λύσεων****(βλ ενότητα 3.6.3.1.4) εφόσον κάθε λύση χαρακτηρίζεται πλεόν και από τα δυό κρίτήρια(C_{max},T_d),*

- ***Αρχείο Pareto***

  Το σύστημά μας επιστρέφει ένα σύνολο μη κυριαρχούμενων λύσεων(Pareto front) άρα η διατήρηση ενός Pareto αρχείου P στο οποίο θα αποθηκεύονται οι μη-κυριαρχούμενες λύσεις είναι απαραίτητη.Έκτος του ότι οι λύσεις του αρχείου αποτελούν το τελικό αποτέλεσμα του συστήματος ,μπορούν να χρησιμοποιηθούν από το σύστημα κατα την διάρκεια εκτελεσής του για την ανανέωση της φερομόνης . Οι επιλογές μας περιορίζονται σε offline και online αποθήκευση των μη κυριαρχούμενων λύσεων.Χώρις να λάβουμε κάποιον ιδιαίτερο παράγοντα υπόψη επιλέξαμε online αποθήκευση των μη κυριαρχούμενων λύσεων σε κάθε κύκλο της εκτέλεσης ίσως επειδή η συγκεκριμένη επιλογή παρέχει ένα πιό αυστηρό ταίριασμα του πληθυσμού των λύσεων με τη πληροφορία φερομόνης(σε αντίθεση με την offline αποθήκευση οι

λύσεις του Pareto αρχείου ανανεώνουν την φερομόνη σε κάθε κύκλο της εκτέλεσης):

Online αποθήκευση

Το Pareto αρχείο αλληλεπιδρά με τον πληθυσμό των λύσεων σε κάθε κύκλο , ώστε να αποθηκέυονται οι μη-κυριαρχούμενες λύσεις και να απορρίπτονται οι κυριαρχούμενες που ήδη υπάρχουν.Οι λύσεις του πληθυσμού συγκρίνονται μια προς μια με τις μη κυριαρχούμενες λύσεις του Pareto συνόλου και επιλέγονται σύμφωνα με τα παρακάτω κριτήρια:

- *Το Pareto συνόλο είναι κενό.*
- *Η υποψήφια λύση δεν κυριαρχείται ή δεν είναι ίση με κάποια λύση που ήδη υπάρχει στο σύνολο.*
- *Η υποψήφια λύση κυριαρχεί κάποιας λύσης του συνόλου.*

Προσεγγίζοντας την παραπάνω δομή στο προβλήμά μας ,έστω ψ^k το χρονοπρόγραμμα του k-οστού πράκτορα το οποίο αντιστοιχεί στο διάνυσμα αντικειμενικών $[F^1{}_{\psi}{}^k \quad F^2{}_{\psi}{}^k]$. Το διάνυσμα συγκρίνεται με το βέλτιστο σύνολο Pareto,P.Εάν είναι μη κυριαρχούμενο από το σύνολο των λύσεων προστίθεται στο P (μαζί με το χρονοπρόγραμμα ψ^k)και όλες οι κυριαρχούμενες από την ψ^k λύσεις που ανήκουν ήδη στο P διαγράφονται.Όπως αναφέραμε παραπάνω σε περίπτωση που το διάνυσμα λύσης $[F^1{}_{\psi}{}^k \quad F^2{}_{\psi}{}^k]$ =$F_{\psi}{}^k$ ανήκει ήδη στο P ,ισχύει δηλαδή $F_{\psi}{}^k = F_{\psi}{}^P$ για κάποια μη-κυριαρχούμενη λύση ψ^P , τότε η ψ^P διαγράφεται από το P ,εφόσον σύμφωνα με τα κριτήρια της Pareto βελτιστότητας απαιτείται τουλάχιστον μια αυστηρή ανισότητα για ένα σημείο ώστε να είναι Pareto βέλτιστο.Όμοίως σε αυτή την περίπτωση η ψ^k αγνοείται.

- ***Αξιολόγηση Pareto λύσεων***

Το πολυκριτηριακό σύστημα έχει ως στόχο την έξοδο ένος συνόλου απο Pareto λύσεις .Προς αυτή την κατέυθυνση θα πρέπει να της αξιολογήσουμε κατάλληλα.Στην μονοκριτηριακή περίπτπωση δεν αντιμετωπίζαμε τέτοιο πρόβλημα καθώς κάθε λύση αξιολογούταν βάση του κριτηρίου που επιθυμούσαμε να ελαχιστοποιήσουμε.Θυμίζουμε τον κανόνα ολικής ανανέωσης του μονοκριτηριακού συστήματος:

$$\tau_{ij}(t+n) \leftarrow (1-\alpha) \cdot \tau_{ij}(t) + \alpha \cdot \Delta\tau_{ij}(t,t+n) \quad , \ \forall (i,j) \in T^{bs}$$

$\Delta\tau_{ij}(t,t+n) = \frac{\quad}{\quad}$, *,με L^{bs} το κόστος της βέλτιστης διαδρομής.*

Βλέπουμε σύμφωνα με την παραπάνω εξίσωση ότι κάθε λύση αξιολογείτο βάση του κόστους της L^{bs}(πχ L^{bs} =Cmax εαν στοχεύουμε στο makespan).Κατα αυτόν τον τρόπο η λύση με μικρότερο Cmax θα πρόσθετε φερομόνη στα αντίστοιχα συστατικά της κατα μια ποσότητα 1/ Cmax.Οσο μικρότερο το Cmax τόσο μεγαλύτερη και η ποσότητα προστιθέμενης φερομόνης.Στην πολυκριτηριακή περίπτωση κάθε λύση χαρακτηρίζεται και από τα δυό κριτήρια (Cmax,Td),άρα είναι αναγκαία η αξιολόγηση της βάση ενός μηχανισμού αξιολόγησης.

Σύμφωνα με τη συζήτηση που αναπτύξαμε στην Ενότητα 3.6.3.1.4 έχουμε δύο επιλόγές για την αξιολόγηση του διανύσματος λύσης ενός πράκτορα.Χάριν απλότητας επιλέξαμε Non-pareto αξιολόγηση των λύσεων σε αντίθεση με τις εξηζητημένες Pareto τεχνικές(dominance ranking).Σημειώνεται ότι αν και οι Pareto τεχνικές επιτρέπουν σε πολλαπλές μη-κυριαρχούμενες λύσεις να ξεχωρίσουν βάση διαφόρων παραγόντων όπως το πόση διαφοροποίηση προσθέτουν σε ένα πληθυσμό μη-κυριαρχούμενων λύσεων,εντούτοις για την παραγωγή ενός καλά κατανεμημένου Pareto μετώπου συνιστάται η αξιολόγηση(ranking) και των κυριαρχούμενων λύσεων σε κάθε επανάληψη για την ανανέωση της φερομόνης(CPACO, Angus 2007) ,ώστε να διατηρηθεί η διαφοροποίηση του συνόλου σε ικανοποιητικά επίπεδα.Ως αποτέλεσμα θα πρέπει να αποθηκέυονται και οι κυριαρχούμενες λύσεις σε κάθε κύκλο του αλγορίθμου κάτι που αυξάνει το υπολογιστικό κόστος της διαδικασίας(Angus & Woodward,2009).Προς αποφυγή των παραπάνω και έχοντας γνώση ότι η επιλογή μας δεν είναι η πλεόν βέλτιστη που θα μπορούσε να ληφθέι,χρησιμοποιούμε Pareto αξιολόγηση. Όποτε λοιπόν ο k-$οστός$ πράκτορας ολοκληρώνει ένα tour επιστρέφει μια πλήρη λύση ψ^k (δηλαδή μια ακολουθία από διεργασίες) στην οποία αντιστοιχεί ένα διάνυσμα λύσης $[F^1{}_\psi{}^k \quad F^2{}_\psi{}^k]$ (η τιμή $F^1{}_\psi{}^k$ αντιστοιχεί στο $makespan$ και η $F^2{}_\psi{}^k$ στον T_d για το χρονοπρόγραμμα ψ^k) και συγκρίνεται με το βέλτιστο σύνολο Pareto,P. Εάν δεν κυριαρχείται από τον πληθυσμό των μη-κυριαρχούμενων λύσεων προστίθεται στο P και η συνεισφορά του k-$οστού$ πράκτορα στον πίνακα φερομόνης θα έιναι ίση με { $1/(F^1{}_\psi{}^k \,.\, F^2{}_\psi{}^k)$ }. Από την τελευταία σχέση βλέπουμε ότι κάθε αντικειμενική λαμβάνει ίσο μερίδιο κατά την ανανέωση της φερομόνης.

Εφόσον το σύνολο P έχει ανανεώθει στον τρέχων κύκλο της εκτέλεσης ,ακολουθώντας το αλγόριθμικό μοτίβο το μονοκριτιριακού σύστήματος εφαρμόζουμε τοπική και ολική ανανέωση της φερομόνης.Ο κανόνας ολικής ανανέωσης της φερομόνης πραγματοποιείται από τους πράκτορες των οποίων οι λύσεις ανήκουν στο Pareto αρχείο P και είναι ο εξής:

$$\tau_{ij} = (1-\rho)^{\bullet}\, \tau_{ij} + \rho / F^1{}_\psi{}^P \,.\, F^2{}_\psi{}^P \,, \quad \forall\, (i, j) \in \psi^P \qquad (4.5)$$

,όπου $F^1{}_\psi{}^P$ είναι η τιμή του makespan για ένα δοσμένο χρονοπρόγραμμα ψ από το σύνολο P $και$ αντίστοιχα $F^2{}_\psi{}^P$ η τιμή της συνολικής βραδύτερης περάτωσης για το ίδιο χρονοπρόγραμμα.

ρ είναι η σταθερά ενίσχυσης και εξάτμισης της φερομόνης .

{ $1/(F^1{}_\psi{}^P \,.\, F^2{}_\psi{}^P)$ }.είναι η συνεισφορά του ψ^P χρονοπρογράμματος στον πίνακα φερομόνης,όπως δηλαδή την ορίσαμε στην αξιολόγηση Pareto λύσεων .

Την εξερεύνηση ενισχύει μια διαδικασία τοπικής ανανέωσης της φερομόνης καθώς ο πράκτορας κινείται από την διεργασία i στην διεργασία j και η τιμή τ_{ij} ανανεώνεται τοπικά βάση του κανόνα τοπικής ανανέωσης ως εξής :

$$\tau_{ij} = (1-\rho)^{\bullet}\, \tau_{ij} + \rho^{\bullet}\, \tau_0 \qquad (4.7)$$

,όπου τ_0 η αρχική ποσότητα φερομόνης σε όλες τις ακμές του γράφου.Εφόσον η συνεισφορά της φερομόνης από κάθε μη-κυριαρχούμενη λύση είναι της τάξης του $1/(F^1{}_\psi{}^P \cdot F^2{}_\psi{}^P)$ μια ποσότητα ίση με $1/n*(F^1{}_\psi{}^h \cdot F^2{}_\psi{}^h)$ για την τιμή τ_0 είναι επαρκής(υπενθυμίζουμε την συζήτηση για τον ορισμό της τ_0 στην Ενότητα 4.2) ,όπου $F^1{}_\psi{}^h$ και $F^2{}_\psi{}^h$ οι αρχικές εκτιμήσεις του κάθε κριτηρίου.

Κανόνας μετάβασης

Σύμφωνα με τη συζήτηση που αναπτύχθηκε στη Ενότητα 3.6.3.1.3 η κατασκευή των λύσεων σε ένα γενικό πολυκριτηριακό ACO αλγόριθμο γίνεται δυναμικά προσαθροίζοντας την πληροφορία φερομόνης και την ευρετική πληροφορία με την μέθοδο σταθμισμένου γινομένου.Η τεχνική αυτη φαίνεται να είναι προς όφελος της εκάστοτε ACO μεθόδου βελτιστοποίησης σε περιπτώσεις ,όπου το σχήμα του τελικού Pareto μετώπου δεν είναι γνωστό ή δεν έχουν οριστεί βάρη a priori που να κατευθύνουν την διαδικασία αναζήτησης(Angus & Woodward,2009) κάτι που ισχύει και για το πολυκριτηριακό JSP που εξετάζουμε.

Επιπλέον σύμφωνα με την ίδια ενότητα προκειμένου να ωθήσουμε τους πράκτορες να κατευθύνουν την αναζητησή τους σε διαφορετικές περιοχές του μετώπου Pareto , ο κάθε πράκτορας "ζυγίζει'' διαφορετικά τα δύο κριτήρια ,καθώς αποφασίζει σε ποιον κόμβο θα μετακινηθεί βάση ενός διανύσματος βαρών {λ,1-λ}.Ακολουθούμε την ανάθεση μοναδικού διανύσματος βαρών σε κάθε πράκτορα σε κάθε κύκλο (Iredi et al,2001).Υπάρχουν αρκετοί τρόποι να ορίσουμε το συγκεκριμένο διάνυσμα (Angus,2007 ,Baran & Schaerer, 2003, Iredi et al., 2001).Σύμφωνα με τους Iredi et al., 2001 το βάρος λ κάθε αντικειμενικής υπολογίζεται για κάθε πράκτορα k ως $\lambda=k-1/m-1$ από ένα σύνολο m πρακτόρων ,$k \in [1,m]$.Από την τελευταία σχέση παρατηρούμε ο 1ος πράκτορας κατασκευάζει λύσεις λαμβάνοντας υπόψη μόνο την ευρετική πληροφορία n_{ij}^2 και αντίστοιχα ο m-οστος πράκτορας δίνει προτεραιότητα μόνο στη n_{ij}^1.Ο ενδιάμεσος πληθυσμός των πρακτόρων θα κατευθύνει την αναζήτηση του σε διαφορετικές περιοχές του Pareto μετώπου ανάλογα με την τιμή του διανύσματος βαρών που του έχει ανατεθεί,διατηρώντας έτσι την διαφοροποίηση των λύσεων στο τελικό Pareto σύνολο.

Λαμβάνοντας υπόψη τα παραπάνω η διαδικασία κατασκευής μιας λύσης από τον k-οστό πράκτορα στο πολυκριτηριακό JSP,υιοθετώντας τους κανόνες μετάβασης του μονοκριτηριακου συστήματος, θα έχει ως εξής :

Ο k-οστός πράκτορας επισκέπτεται τον επόμενο κόμβο-διεργασία j από το σύνολο των διαθέσιμων διεργασιών ακολουθώντας τον ίδιο κανόνα μετάβασης με το μονοκριτηριακό συστήμα προσαρμοσμένο στο πολυκριτηριακό πρόβλημα(βλ. εξ. 3.36), δηλαδή:

-

(4.8)

,όπου q, και β παράμετροι όπως έχουν οριστεί στο προηγούμενο κεφάλαιο , $\lambda=(k-1/m-1)$, ενώ $J \in$ μια διεργασία η οποία επιλέγεται τυχαία με πιθανότητα j) βάση του κανόνα αναλογίας ,δηλαδή :

Η ευρετική πληροφορία n_{ij}^1 για το C_{max} και αντίστοιχα η ευρετική πληροφορία n_{ij}^2 για το T_d θα οριστούν στο κεφάλαιο Ανάλυση και Σχεδίαση του συστήματος .

Επιπλέον σχεδιαστικά ζητήματα

Σε αυτό το σημείο θα αναλύσουμε ένα σημαντικό ζήτημα που προκύπτει από την χρήση ενός πίνακα φερόμόνης και σχετίζεται με την τιμή τ_0 .Στην state of the art μελέτη των Angus & Woodward ,2009 για την σχεδίαση πολυκριτηριακών ACO αλγορίθμων αναφέρεται ένα συχνό πρόβλημα που σχετίζεται με την παρουσία λανθασμένης πληροφορίας στον πίνακα φερομόνης όταν χρησιμοποιείται μοναδικός πίνακας φερομόνης.Σε αυτήν την περίπτωση όπως ήδη έχουμε δεί οι αντικειμενικές τιμές από μια λύση του πληθυσμού συναθροίζονται σε μια (βλ. κανόνα ολικής ανανέωση φερομόνης) και αν ικανοποιεί τα κριτηρία κυριαρχίας η λύση εισάγει αυτή την πληροφορία στον πίνακα φερομόνης. Εάν ωστόσο το Pareto μέτωπο βελτιωθεί(πχ. νεές μη-κυριαρχούμενες λύσεις αντικατέστησαν κάποιες από τις ήδη υπάρχουσες) η πληροφορία που είχαν εισάγει οι παλιές λύσεις δεν μπορεί να εξαχθεί από τον πίνακα φερομόνης,με άλλα λόγια ο πίνακας φερομόνης δεν μπορεί να ανακατασκευαστεί ώστε να αντικατοπτρίσει αυτή την αλλαγή ,χωρίς την χρήση των πρωτότυπων λύσεων ,οι οποίες στις περισσότερες των περιπτώσεων, όπως και στο σύστημα μας δεν αποθηκεύονται.Στην περίπτωση πολλαπλών πινάκων φερομόνης δεν πραγματοποιείται κάποια συνάθροιση των αντικειμενικών τιμών πριν την ανανέωση της φερομόνης πχ.κάθε αντικειμενική τιμή μιας λύσης συνεισφέρει σε συγκεκριμένο πίνακα φερομόνης,και η συγκέντρωση της πληροφορίας φερομόνης μπορεί να αλλαχθεί δυναμικά κατα την κατασκευή των λύσεων.
Για τις ανάγκες του προβλήματος μας υιοθετήσαμε την μεθοδολογία των Baran & Schaerer et al., 2003. Οι τελευταίοι αντιμετωπίζουν το παραπάνω πρόβλημα επιτρέπωντας την επαναρχικοποίηση του πίνακα φερομόνης με μία νέα τιμή τ_0' σε περίπτωση που το Pareto αρχείο έχει βελτίωθεί. Η επαναρχικοποίηση της τ_{ij} αποσκοποπεί στην εξερεύνηση νέων λύσεων από τους πράκτορες χωρίς την πιθανώς λανθασμένη πληροφορία που έχει εισαχθεί απο λύσεις που έχουν ήδη διαγραφεί από το Pareto αρχείο ,επειδή κυριαρχήθηκαν από νέες.Οι συγγραφείς επιλύουν στην έρευνά τους το VPRTW πρόβλημα θεωρώντας δυό κριτήρια.

Εφαρμόζοντας την μέθοδο των Baran & Schaerer στο πρόβλημα μας στο τέλος κάθε επανάληψης (κύκλου),υπολογίζεται εκ νέου η τιμή της ποσότητας εκκίνησης της φερομόνης τ_0' σε όλες τις ακμές του γραφήματος χρησιμοποιώντας τις μέσες τιμές του P ως εξής:

$$\tau_0' = 1/\ n \qquad (4.10)$$

,όπου n ο συνολικός αριθμός των διεργασιών του προβλήματος, το μέσο makespan και ο μέσος χρόνος συνολικής βραδύτητας των λύσεων στο P. Εάν ισχύει $\tau_0 > \tau_0'$,επειδη το Pareto σύνολο P έχει βελτιωθεί ,η φερομόνη σε όλα τα μονοπάτια του κατασκευαστικού γράφου αρχικοποιείται με την τιμή τ_0' ,ώστε να ενισχυθεί η εξερεύνηση(exploration) νέων λύσεων,διαφορετικά ευνοείται η

εκμετάλλευση(exploitation) της συσσωρευμένης γνώσης με ολική ανανέωση των μονοπατιών φερομόνης στο γράφημα χρησιμοποιώντας κάθε λύση στο τρέχων σύνολο P.

Έξοδος συστήματος

Η έξοδος του συστήματος περιλαμβάνει το τελικό Pareto αρχείο P και ένα πίνακα με τον μέγεθος του P,το πλήθος δηλαδή των μη κυριαρχούμενων λύσεων, σε κάθε κύκλο της εκτέλεσης,ο οποίος χρησιμεύει για να δέιξουμε στον αναγνώστη πως το σύστημα προοδευτικά βρίσκει σε κάθε κύκλο νέες μη κυριαρχούμενες λύσεις.

Ανακεφαλαιώνοντας το σύστημα χρησιμοποιεί:

- *Έναν μοναδικό πληθυσμό πρακτόρων ,*
- *προσαθροίζει την πληροφορία φερομόνης και την ευρετική πληροφορία με τη μέθοδο σταθμισμένου γινομένου(w.p),*
- *αποθηκεύει την πληροφορία των καλών λύσεων σε ένα μοναδικό πίνακα φερομόνης,*
- *ενώ διατηρεί την ευρετική πληροφορία για κάθε κριτήριο σε διαφορετικούς.*
- *αναθέτει δυναμικά τα βάρη σε κάθε αντικειμενική και καθοδηγεί την ανανέωση της φερομόνης βάση ενος εξωτερικού Pareto συνόλου P (online storage) ,όπου διατηρεί όλες τις μη κυριαρχούμενες λύσεις σε κάθε κύκλο του.*
- *Τέλος χρησιμοποιεί μηχανισμό επαναρχικοποίησης του πίνακα φερομόνης για την διαγραφή της λανθασμένης πληροφορίας που έχει εισαχθεί από κυριαρχούμενες λύσεις.*

Υπολογιστική Διαδικασία

Καθώς έχουμε συνδέσει το περιβάλλον του συστήματος μας με το περιβάλλον του προβλήματος, ακολουθεί η υπολογιστική διαδικασία της εκτέλεσης του συστήματος που αναπτύξαμε.

- ***Βήμα 1:****Αρχικά εισάγονται τα δεδομένα και καθορίζονται το περιβάλλον και οι παράμετροι του συστήματος καλώντας την συνάρτηση ΑρχικοποίησηςΠεριβάλλοντος().Κατόπιν αρχικοποιείται ο πληθυσμός των πρακτόρων καλώντας την συνάρτηση ΔημιούργησεΠράκτορες().*

- ***Βήμα 2:****Ακολουθεί η κατασκευή των λύσεων καλώντας την συνάρτηση ΕκτέλεσεΠράκτορες().*
-
- ***Βήμα 3:****Ο πληθυσμός των λύσεων ως είσοδος στην συνάρτηση ΕνημέρωσεΑρχείοPareto().Υπολογίζεται η τιμή των κριτηριών* $F^1{}_{\psi}{}^k$, $F^2{}_{\psi}{}^k$ *για κάθε πλήρη λύση* ψ^k *και ανανεώνεται το σύνολο P σύμφωνα με τα κριτήρια που αναφέραμε παραπάνω.*

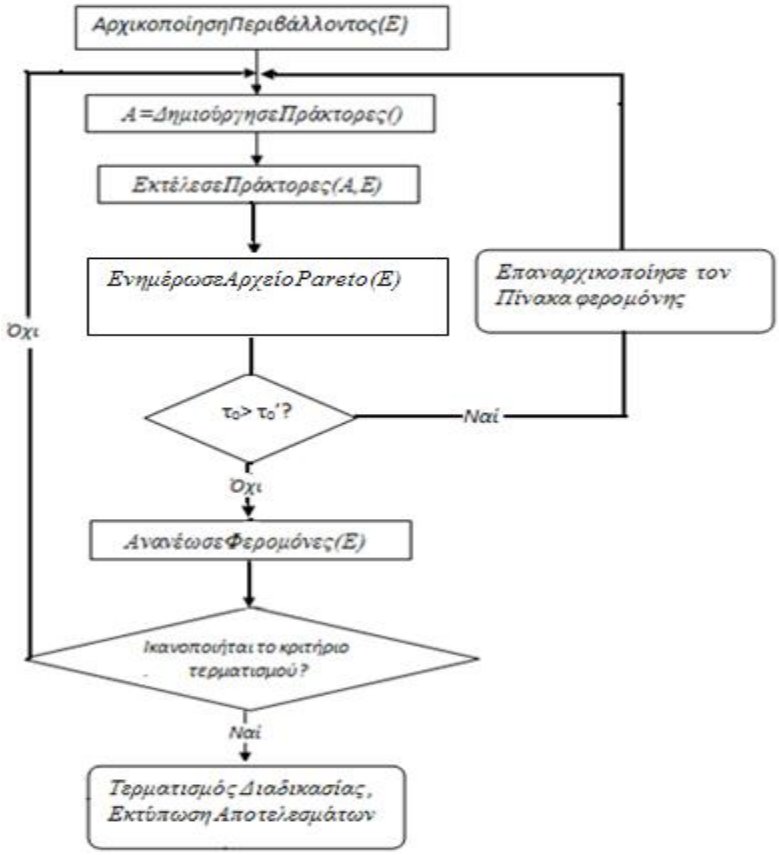
- ***Βήμα 4:*** *Υπολογίζεται η νέα τιμή εκκινησης της φερομόνης* τ_0' *συμφ. με την εξ. 4.10.Εάν ισχύει* $\tau_0 > \tau_0'$ *το Pareto σύνολο έχει βελτιωθεί και τα μονοπάτια φερομόνης* $\{\tau\}$ *επαναρχικοποιούνται με την τιμή* τ_0' *,ώστε ενισχυθεί η εξερεύνηση(exploration) νέων λύσεων.Εάν* $\tau_0 < \tau_0'$ *καλείται η συνάρτηση ΑνανέωσεΦερομόνες() και πραγματοποιείται ολική ανανέωση της φερομόνης ώστε να ευνοηθεί η εκμετάλλευση(exploitation) της συσσωρευμένης γνώσης*

στον πίνακα φερομόνης.

- ***Βήμα 5:*** *Εάν τα κριτήρια τερματισμού καλύπτονται, τότε ακολουθεί η εκτύπωση των αποτελεσμάτων, διαφορετικά επαναλαμβάνεται η παραπάνω διαδικασία με τις ανανεωμένες τιμές ,αρχίζοντας από το Βήμα 2. Το σύστημα τερματίζει μετά από πεπερασμένο αριθμό επαναλήψεων και επιστρέφει το τελικό σύνολο P με τις μη κυριαρχούμενες λύσεις(Pareto βέλτιστες) που έχουν βρεθεί κατα την διάρκεια της εκτέλεσης και έναν πίνακα με το πλήθος των κυριαρχούμενων λύσεων στο P σε κάθε κύκλο του αλγόριθμου.*

  Η διαδικασίες που ακολουθούνται σε κάθε βήμα θα αναλυθούν πιο λεπτομερώς στο Κεφάλαιο 5 ανάλυση και σχεδίαση συστήματος.

Το διάγραμμα ροής του προτεινόμενου πολυκριτηριακού συστήματος δίνεται στην Εικόνα 4.5 :



Εικόνα 4.5 : Διάγραμμα ροής του προτεινόμενου πολυκριτηριακού συστήματος.

Κεφάλαιο 5 Ανάλυση και Σχεδίαση συστήματος

5.1 Δεδομένα στο πρόβλημα μας

Σκοπός της εργασίας αυτής είναι η σχεδίαση συστήματος ευφυών πρακτόρων εμπνευσμένων από τις αποικίες μυρμηγκιών για την επίλυση του JSP .Σκοπός μας είναι η εφαρμογή του προτεινόμενου single objective συστήματος για κάθε κριτήριο ξεχωριστά σε μια σειρά από benchmarks ώστε να ελεγχθεί η απόδοση του στη ελαχιστοποίηση τους.
Επιπλέον σκοπός της εργασίας είναι και η σχεδίαση ενος πολυκριτηριακού συστήματος πρακτόρων και η εφαρμογή του στο πολυκριτηριακό JSP με σκοπό την εύρεση ενός άριστου pareto συνόλου για κάθε benchmark.
Δοκιμάσαμε τα προτεινόμενα συστήματα σε εφτά κλασσικά προβλήματα χρονοπρογραμματισμού Ft06, Ft10 ,La01, La06, La10, La21 και Orb04 τα οποία αντλήθηκαν απο την OR-Library του J E Beasley, που είναι κλιμακούμενης δυσκολίας.

5.2 Ανάλυση και Σχεδίαση Μονοκριτηριακού σύστηματος

Στα πλαίσια αυτής της διπλωματικής επιλύουμε το πρόβλημα του Job-Shop Scheduling,JSP ,το οποίο παρουσιάστηκε αναλυτικά στην Ενότητα 3.1.Θεωρούμε δύο αντικειμενικά κριτήρια ,ονομαστικά 1) το makespan,C_{max} και 2) τον συνολικό χρόνο βραδύτερης περάτωσης,T_d τα οποία επιδιώκουμε να ελαχιστοποίησουμε ξεχωριστά σε κάθε περίπτωση.

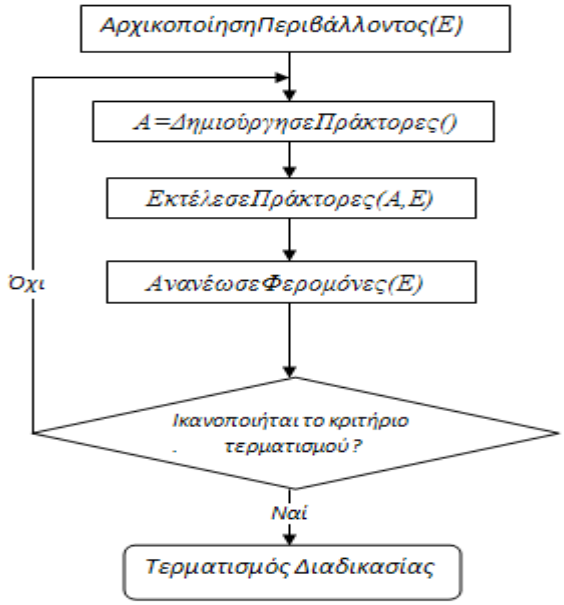
Καταρχήν θα πρέπει να αναφερθεί ότι το πρόγραμμα είναι γραμμένο στο περιβάλλον της πλατφόρμας MATLAB..Αρχικά φορτώνονται τα δεδομένα του προβλήματος, δηλαδή τα .txt αρχεία που αναπαριστούν ένα συγκεκριμένο στιγμιότυπο του Job-Shop προβλήματος. Τα δεδομένα περιλαμβάνουν ένα πίνακα M μεγέθους n (εργασίες) x m (μηχανές) ο οποίος περιέχει τις μηχανές στις οποίες πρέπει να δρομολογηθούν οι διεργασίες και ένα πίνακα T ίδιου μεγέθους ο οποίος περιέχει τους χρόνους επεξεργασίας των διεργασιών. Επιπλέον δεδομένα είναι ένα διάνυσμα DDate μεγέθους (1 x n) που περιλαμβάνει τους χρόνους παράδοσης των εργασιών.Για παράδειγμα οι πίνακες M και T για ένα πρόβλημα 6x6 έχουν την εξής μορφή(Ping & Mingai et al,2011):

$$M_{FT06} = \begin{bmatrix} 3 & 1 & 2 & 4 & 6 & 5 \\ 2 & 3 & 5 & 6 & 1 & 4 \\ 3 & 4 & 6 & 1 & 2 & 5 \\ 2 & 1 & 3 & 4 & 5 & 6 \\ 3 & 2 & 5 & 6 & 1 & 4 \\ 2 & 4 & 6 & 1 & 5 & 3 \end{bmatrix}, \quad T_{FT06} = \begin{bmatrix} 1 & 3 & 6 & 7 & 3 & 6 \\ 8 & 5 & 10 & 10 & 10 & 4 \\ 5 & 4 & 8 & 9 & 1 & 7 \\ 5 & 5 & 5 & 3 & 8 & 9 \\ 9 & 3 & 5 & 4 & 3 & 1 \\ 3 & 3 & 9 & 10 & 4 & 1 \end{bmatrix};$$

Κάθε γραμμή αντιστοιχεί σε μια εργασία,άρα ο παραπάνω M πίνακας μας πληροφορεί ότι η 1η διεργασία της 1ης εργασιάς δρομολογείται στη μηχανή 3, η 2η διεργασία της 1ης εργασίας στη μηχανή 1, η 3η διεργασία της 1ης εργασίας στην

μηχανή 2 , κ.ο.κ. Αντίστοιχα για τον πίνακα Τ η 1η διεργασία της 1ης εργασιάς έχει χρόνο επεξεργασίας 1, η 2η διεργασία της 1ης εργασίας έχει χρόνο επεξεργασίας 3 κ.ο.κ.

Παρακάτω θα περιγράψουμε τις διάφορες σχεδιαστικές λεπτομέρειες του single objective συστήματος που προτείναμε στο προηγούμενο κεφάλαιο .Παραθέτουμε και πάλι το διάγραμμα ροής του προτεινόμενου συστήματος :



Εικόνα 5.1:Διάγραμμα ροής προτεινόμενου μονοκριτηριακού συστήματος.

ΑρχικοποίησηΠεριβάλλοντος():*Το σύστημα αρχικά χρησιμοποιεί μια συνάρτηση ΑρχικοποίησηΠεριβάλλοντος() η οποία επιστρέφει το περιβάλλον Ε τον γράφο του προβλήματος.Το περιβάλλον αντιπροσωπεύεται από έναν πίνακα Graph μεγέθους n(εργασίες) x m(μηχανες).Κάθε γραμμη του πίνακα αντιστοιχεί σε μία εργασία και κάθε στήλη σε μια διεργασία.Για παράδειγμα σε ένα πρόβλημα 6x6 η 1η γραμμη θα περιέχει τους αριθμούς 1 εως 6 που συμβολίζουν τις διεργασίες της 1ης εργασίας,η 2η τους αριθμούς 7 εως 11 κλπ.*

Επιπλέον μέσα στη συνάρτηση αρχικοποιούνται και οι παράμετροι της αλγοριθμικής διαδικασίας ονομαστικά :η τιμή αρχικής φερομόνης τ_0, η τιμή q_0 , οι ρυθμοί εξάτμισης α και ρ και ο αριθμός των κύκλων maxit.

Όπως αναφέραμε στην Ενότητα 4.2 η τ_0 αρχικοποιήται στην τιμή $\tau_0 = (1/N_* L_{SPT})^{-1}$ και $\tau_0 = (1/N_* T_{EDD})^{-1}$ανάλογα με το κριτήριο που ελαχιστοποιούμε C_{max} και T_d αντίστοιχα ,όπου N ο αριθμός των διεργασιών του benchmark .Οι υπόλοιπες τιμές ορίζονται από εμάς.

Για τον υπολογισμό των L_{SPT} και T_{EDD} εφαρμόσαμε τον αλγόριθμο των Giffler and Thompson(GT algorithm,1969),που ανήκει στην γνωστή οικογένεια των List scheduling αλγορίθμων και είναι μια κλασσική και χρήσιμη μέθοδος για την παραγωγή ενεργών(active) χρονοπρογραμματών κατα την οποία προστίθονται διαδοχικά διεργασίες στο χρονοπρόγραμμα από ένα σύνολο εφικτών(διαθέσιμων) διεργασιών Q ,αρχίζοντας με τις πρώτες διεργασίες κάθε εργασίας.

Αρχίζοντας λοιπόν από το σύνολο αυτό ο αλγόριθμος καθορίζει σε κάθε βήμα του το σύνολο των διαθέσιμων διεργασιών προς προγραμματισμό.Για κάθε διεργασία του συνόλου υπολογίζεται ο ενωρίτερος χρόνος περάτωσης (earliest completion time) της .Έστω O_{min} η διεργασία με τον ελάχιστο ενωρίτερο χρόνο περάτωσης c_{min}. Τότε οι διεργασίες που αρχίζουν την επεξεργασίας τους πριν τον c_{min} στην ίδια μηχανή με την διεργασία O_{min} συνιστούν το σύνολο των διεργασιών σε διαμάχη(conflict set), K . Αν λοιπόν δύο ή περισσοτέρες διεργασίες συναγωνίζονται για την ίδια μηχανή (ανήκουν στο conflict set, K) η διαμάχη λύεται επιλέγοντας μια διεργασία βαση κάποιου κανόνα ανάθεσης(στην περίπτωση μας των SPT,EDD κανόνων).Η επιλεγμένη διεργασία ανατίθεται στη συνέχεια στο χρονοπρόγραμμα.

Ο Αλγόριθμος 5.1 δείχνει τη βασική διαδικασία χρησιμοποιώντας τους ακόλουθους συμβολισμούς:

Q *Το σύνολο των εφικτών για προγραμματισμό διεργασιών*

$first\ (i)$ *Η πρώτη διεργασία της εργασίας* J_i .

$succ^T(o)$ *Το σύνολο των διαδόχων της διεργασίας* o *στην ίδια εργασία.*

$s^e(o)$ *Ο ενωρίτερος χρόνος εκκίνησης της διεργασίας* o.

$c^e(o)$ *Ο ενωρίτερος χρόνος περάτωσης της διεργασίας* o,*(πχ.* $s^e(o) + p(o)$ *).*

K *Το σύνολο των διεργασιών σε διαμάχη .*

Q' *Το σύνολο των εφικτών για προγραμματισμό διεργασιών στη μηχανή*

$\mathcal{M}(o)$ *Η μηχανή στην οποία ανατίθεται η διεργασία* o.

Ο ενωρίτερος χρόνος κατα τον οποίο η μηχανή k *γίνεται διαθέσιμη.*

Αλγόριθμος 5.1:Ο αλγόριθμος των Giffler- Thompson για το JSP.

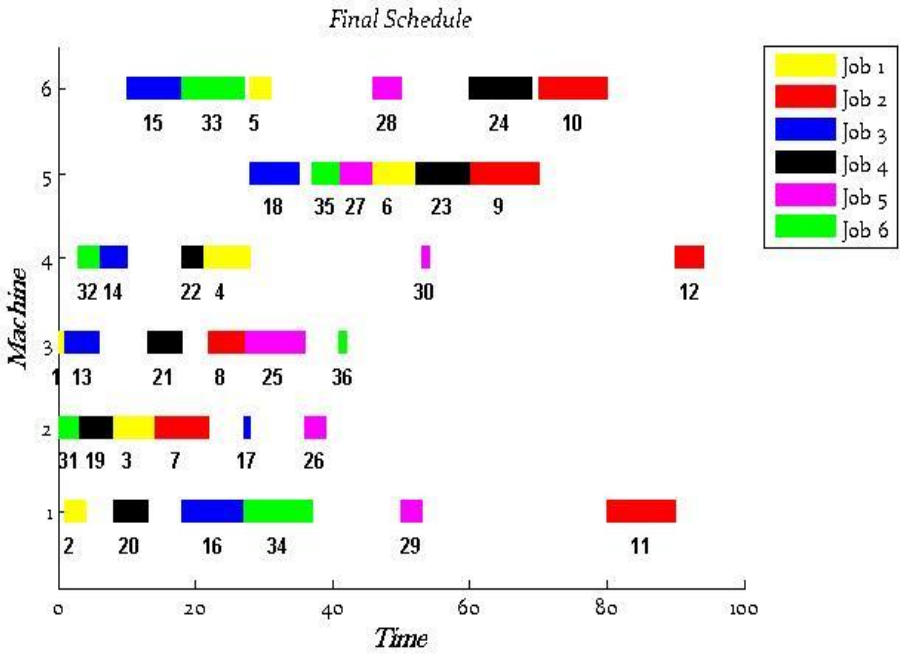
Initialize $Q \leftarrow$;
$s^e(o) \leftarrow 0,\ \forall\, o \in Q$; $\leftarrow 0$, $\forall\, 1 \le k \le m$;

while $Q \neq$ **do**
Determine $c_{min} = \min_{o \in Q} (s^e(o) + p(o))$;
Build conflict set $K = \{o \mid o \in Q \quad \mathcal{M}(o) = \mathcal{M}(o_{min}) \quad s^e(o) < c_{min}\}$;
Choose operation o' to be scheduled next according to SPT/EDD rule.
Remove o' from $Q \leftarrow Q \setminus \{o'\}$;

Set $= c^e(o)$; .
Determine set $Q' = \{o \mid o \in Q \quad \mathcal{M}(o) = \mathcal{M}(o')\}$;

forall $o \in Q'$ **do**
. $s^e(o) \leftarrow \max(s^e(o),$ $)$; .
. **end**
. **forall** $o \in succ^T(o')$ **do** .

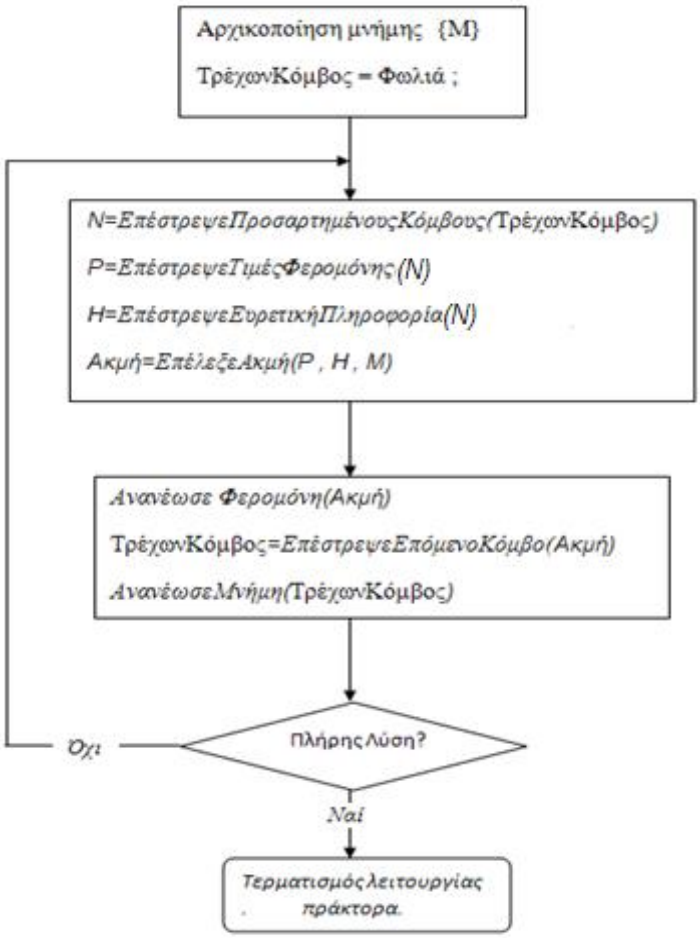
. $Q \leftarrow Q \quad \{o\}$; .
. $s^e(o) \leftarrow \max(\ s^e(o),\qquad\qquad)$; .
. **end**
end



Εικόνα 5.2:Το διάγραμμα Gantt για το $FT06(6x6)$ πρόβλημα ύστερα από εφαρμογή του GT αλγορίθμου(makespan 94).

ΔημιούργησεΠράκτορες():*Η συνάρτηση αυτή απλά καθορίζει τον αρχικό πληθυσμό A των πρακτόρων ,τιμή καθοριζόμενη από εμάς.*

ΕκτέλεσεΠράκτορες():*Η συνάρτηση περιλαμβάνει την λειτουργικότητα του πράκτορα(Εικόνα 5.3) και είναι υπέυθυνη για την κατασκεύη των λύσεων από τον πληθυσμό A των πρακτόρων στο περιβάλλον E.*



Εικόνα 5.3:Διάγραμμα ροής της λειτουργίας ενός πράκτορα.

1. *Καταρχήν όπως έχουμε αναφέρει ο κάθε πράκτορας χρειάζεται μια εσωτερική μνήμη M όπου διατηρεί μιας δομή για τον έλεγχο των περιορισμών διαδοχής/προκατοχής ανάμεσα στις διεργασίες μιας εργασίας .Όπως σημειώθηκε στην Ενότητα 4.2 αρκεί να διατηρήσουμε μια λίστα επιτρεπόμενων διεργασιών* **N** *(·) για κάθε πράκτορα από την οποία επιλέγουμε την επόμενη διεργασία-κόμβο που θα μεταβούμε.Θυμίζουμε η συγκεκριμένη δομή έχει ως εξής:Η λίστα* **N** *αρχικοποιείται αρχικά με τις πρώτες διεργασίες κάθε εργασίας,* **N**={ O_{i1}|i [1,n] } *,όπου n ο αριθμός των εργασιών.Κάθε φορά που το μυρμήγκι μεταβαίνει από την διεργασία O_{ij} στην O_{kl} ο κόμβος που αντιστοιχεί στην διεργασία O_{ij} διαγράφεται, ώστε να μην τον επισκεφτούμε ξανά ,διασφαλίζοντας έτσι την απουσία κύκλου στον γράφο –επιπλέον εάν η O_{kl} δεν είναι η τελευταία διεργασία στην σειρά της εργασίας k,προστίθεται στην λίστα* **N** *ο κόμβος της αμέσως επόμενης διεργασίας της εργασίας, πχ. η O_{kl+1} διεργασία ,ώστε να ικανοποιηθούν οι περιορισμοί διαδοχής ανάμεσα στις διεργασίες μιας εργασίας.Οι διεργασίες που έχει επισκεφθεί μέχρι την χρονική στιγμή t ο πράκτορας θα μπορούσαν $\nu\alpha$ αποθηκεύτουν σε μια λίστα $tabu$,η οποία θα περιέχει το τελικό χρονοπρόγραμμα .*

Χρησιμοποιώντας την παραπάνω λογική για την υλοποίηση της μνήμης M

*δημιουργήσαμε μια μοναδική λίστα την οποία ονόμασαμε Allowed η οποία περιέχει τις διεργασίες που έχουν ηδη προγραμματιστεί(tabu), την τρέχουσα διεργασία στην οποία βρίσκεται ο πράκτορας(ΤρέχωνΚόμβος) και τις διεργασίες στις οποίες μπορεί να μεταβεί στην επόμενη επανάληψη της εκτέλεσης(**N**).*

*Διατηρούμε ένα δείκτη, έστω j [1,N+1](όπου N ο συνολικός αριθμός των διεργασιών) που δείχνει στην τρέχουσα διεργασία Allowed{j} από την οποία επιθυμεί να μεταβεί ο πράκτορας.Όλες οι Allowed{2,j-1} διεργασίες ανήκουν στην tabu λίστα (αυτές δηλαδή που έχουν ήδη προγραμματιστεί),ενώ οι επιτρεπόμενες διεργασίες της λίστας **N** θα αντιστοιχούν στις Allowed{j+1,end} διεργασίες. Κατα την αρχικοποίηση της λίστας η πρώτη διεργασία στην λίστα είναι ο dummy node της πηγής δηλαδή η φωλιά και επιπλέον οι πρώτες διεργασίες από κάθε εργασία .Το διάνυσμα Allowed αλλάζει μέγεθος δυναμικά καθώς οι διεργασίες προστίθονται σε αυτό και σκοπός ειναι στο τέλος ενος κύκλου να περιέχει το τελικό χρονοπρόγραμμα, δηλαδη μια ακολουθία άπό όλες τις διεργασίες.*

2. *Η συνάρτηση ΕπέστρεψεΠροσάρτημένουςΚόμβους() δέχεται ως όρισμα τον ΤρέχωνΚόμβο , Allowed{j} ,και επιστρέφει το σύνολο **N** από την Allowed.*

3. *Κατόπιν καλούνται οι συναρτήσεις ΕπέστρεψεΦερομόνη() και ΕπέστρεψεΕυρετικήΠληροφορία().Η συνάρτηση ΕπέστρεψεΦερομόνη επιστρέφει τις τιμές φερομόνης τ_{ij} που σχετίζονται με τις ακμές που συνδεόυν τον ΤρέχωνΚόμβο με κάθε διεργασία του συνόλου **N**.Για το λόγο αυτό διατηρεί έναν πίνακα φερομόνης phmone μεγέθους {nxm (διεργασίες)+1} x {nxm (διεργασίες)}. Η {n x m +1} γραμμή αντιπροσωπεύει τις τιμές φερομόνης από τον dummy node πρός τους αρχικές διεργασίες(στήλες) των εργασιών.Κάθε στοιχείο λοιπόν του phmone περιέχει την τιμή φερομόνης που σχετίζεται με την μετάβαση από μια διεργασία σε μια άλλη.*

   Η συνάρτηση ΕπέστρεψεΕυρετικήΠληροφορία() επιστρέφει ανίστοιχα τις τιμές ευρετικής πληρόφορίας n_{ij} και έχει διττή λειτουργικότητα ανάλογα με το κριτήριο που ελαχιστοποιούμε.

   - *Θεωρώντας το C_{max} ως αντικειμενική συνάρτηση χρησιμοποιήσαμε το κριτήριο του μεγαλύτερου εναπομείναντος χρόνου κατεργασίας (Longest Remaining Processing Time,LRT)), σύμφωνα με το οποίο επιλέγουμε με μεγαλύτερη πιθανότητα την διεργασία O_j από το σύνολο **N** ανάμεσα σε αυτές δηλαδή που συνδέονται με την Allowed{j} και έχει τη μεγαλύτερη τιμή στο κριτήριο,δηλαδη το άθροισμα του χρόνου επεξεργασίας της O_j και των διαδόχων της μέσα στην ίδια έργασια.Κατα αυτή την έννοια μια εργασία που υπολείπεται περισσότερο χρόνο για να ολοκληρώσει την επεξεργασία των διεργασιών της θα προτιμάται από το σύστημα.Ο κανόνας έχει δειχτεί ότι είναι είναι αρκετά αποδοτικός σε προβλήματα με στόχο το C_{max} (Zhang, Gen & Fujimura,2004).*

     Εφόσον η τιμή της ευρετικής πληροφορίας των διεργασιών σε αυτή την περίπτωση είναι σταθερή και γνωστή a-priori καθόλη την διάρκεια του

πειράματος,την θέσαμε σαν έισοδο στο σύστημα μέσω ενός πίνακα vis μέγεθους ίδιου με τον πίνακα φερομόνης.

- *Αντίστοιχα για το πρόβλημα βραδύτερης περάτωσης T_d χρησιμοποιήσαμε τον MDD (Modified Due Date) κανόνα ,μια παραλλαγή του EDD (Earliest Due Date) κανόνα , σύμφωνα με τον οποίο επιλέγουμε με μεγαλύτερη πιθανότητα την διεργασία με τον μικρότερο χρόνο ύφεσης (slack time).Ο κανόνας αυτός χρησιμοποιείται συχνά για την ελαχιστοποίηση αντικειμενικών που περιλαμβάνουν χρόνους παράδοσης και αναλύεται παρακάτω:*

  Ο MDD κανόνας χρησιμοποιείται στην βιβλιογραφία στην πλειοψηφία των περιπτώσεων για την ελαχιστοποίηση της συνολικής βραδύτερης περάτωσης σε προβλήματα μιας μηχανής(SMTTP, Bauer,Bullnheimer,Hartl & Strauss,2000) .Οι παραλλαγμένοι χρόνοι παράδοσης των εργασιών ορίζονται ως $d_j' = max\{T+p_j, d_j\}$, όπου T ο συνολικός χρόνος ολοκλήρωσης των ήδη προγραμματισμένων εργασιών στην μηχανή, p_j ο χρόνος επεξεργασίας της εργασίας j προς προγραμματισμό και d_j ο σχετικός χρόνος παράδοσης της(due date).Τότε θα έχουμε $n_{ij} = 1/\ max\{T+p_j, d_j\}$.

  Προσαρμόσαμε εύκολα τον κανόνα στο JSP όπου έχουμε πολλές μηχανές ,καθώς η μόνη διαφορά είναι η τιμή T που μπορει να διαφέρει σε κάθε μηχανή για κάθε εργασία j ,αρκεί δηλαδή να υπολογίσουμε την ευρετική τιμή των υποψήψιων εργασιών προς προγραμματισμό στο $\mathbf{N}$ ως προς την αντίστοιχη μηχανή που πρέπει να δρομολογηθούν.Εφόσον λοιπόν στο JSP η διεργασία της j εργασίας δρομολογείται σε συγκεκριμένη μηχανή , ορίζουμε την τιμή T ως το χρονικό σημείο κατα το οποίο η j-οστή εργασία μπορεί να εκκινήσει την επεξεργασία της στη σχετική μηχανή ή διαφορετικά το χρονικό σημείο κατα το οποίο η μηχανή είναι διαθέσιμη για να δεχτεί την j-οστή εργασία.

  Όπως βλέπουμε ο κανόνας δίνει μεγαλύτερη προτεραιότητα στις εργασίες με τον μικρότερο χρόνο παράδοσης στη μεγαλύτερη διάρκεια ,ενώ προς τις τελευταίες επαναλήψεις της εκτέλεσης ,όπου θα έχουμε $T+p_j \geq d_j$,ευνόουνται οι εργασίες με τον μικρότερο ενωρίτερο χρόνο περάτωσης.

  Σε αυτή την περίπτωση λοιπόν η ευρετική πληροφορία για τους επιτρεπόμενους κόμβους στο $\mathbf{N}$ υπολογίζεται δυναμικά από την συνάρτηση ΕπέστρεψεΕυρετικήΠληροφορία() βάση των τιμών τους T και p_j.

4. *Στη συνέχεια καλείται η συνάρτηση ΕπέλεξεΑκμή() η οποία επιλέγει και επιστρέφει την ακμή που συνδέει τον ΤρέχωνΚόμβο με μια διεργασία στο $\mathbf{N}$ την οποία θα ακολουθήσει ο πράκτορας στο επόμενο βήμα του.Δέχεται ως όρισμα τις τιμές t_{ij} και n_{ij} και υπολογίζει τις πιθανότητες μετάβασης του πράκτορα βάση των κανόνων μετάβασης(εξ. 4.3 , εξ. 4.4). Οι πιθανότητες μετάβασης αποθηκέυονται σε διάνυσμα prob και η συνάρτηση επιλέγει την*

επόμενη ακμή βάση ενός μηχανισμού(roulette wheel selection,[41]).

Όταν ο πράκτορας συναντάει ποικίλους συνδέσμους (ακμές) για να επιλέξει, δημιουργεί έναν τροχό ρουλέτας έτσι ώστε όλοι οι επιτρεπόμενοι κόμβοι μοιράζονται τον τροχό με διαφορετικές αναλογίες σύμφωνα με την τιμή της πιθανότητάς τους prob. Όσο υψηλότερη η πιθανότητα ενός συνδέσμου τόσο μεγαλύτερη αναλογία αντιστοιχεί στον τροχό. Σε αυτό το σημείο, ο τροχός περιστρέφεται για να καθοριστεί ο σύνδεσμος που ενώνει την διεργασία Allowed{j}) με την διεργασίες του συνόλου **N** *που θα επιλέξει ο πράκτορας ως επόμενο κόμβο για τη διαδρομή του.Στον Αλγόριθμο 5.2 παρατίθεται ο ψευδοκώδικας του μηχανισμού τροχού ρουλέτας που υλοποιήσαμε:*

```
r = τυχαίος αριθμός, 0≤r≤ 1;
sum=0;
Για κάθε στοιχείο i στο N
{          sum = sum+prob(i);
Εάν r < sum {επέστρεψε i;}
}
```

Αλγόριθμος 5.2 :Μηχανισμός roulette wheel selection.

5. *Η συνάρτηση ΑνανέωσεΦερομόνη() κατόπιν ανανεώνει τον πίνακα phmone στο στοιχείο που αντιστοιχεί στην επιλεγμένη ακμή βάση του κανόνα τοπικής ανανέωσης(εξ 4.3).*

6. *Η συνάρτηση ΕπέστρεψεΚόμβο () δέχεται ως όρισμα την επιλεγμένη ακμή και επιστρέφει τον νεό ΤρέχωνΚόμβο από το σύνολο* **N.**

7. *Τέλος καλείται η συνάρτηση ΑνανέωσεΜνήμη() με όρισμα τον νεό ΤρέχωνΚόμβο και ανανέωνει ανάλογα την λίστα Allowed.Η ανανέωση της λίστας περιλαμβάνει τα εξής βήματα:*

- *Γίνεται swap του νέου κόμβου στην θέση Allowed{j+1} .Ο νέος ΤρέχωνΚόμβος ακολουθεί πλεόν τον προηγούμενο ΤρέχωνΚόμβο στην ακολουθία των διεργασιών.*
- *Αύξηση του δείκτη j κατα 1:Ενημερώνεται έτσι η λίστα tabu ->Allowed{2,j-1} με τον προηγούμενο ΤρέχωνΚόμβο .*
- *Προστίθεται στη λίστα Allowed η επόμενη διεργασία της εργασίας του νέου ΤρέχωνΚόμβου από τον πίνακα Graph:Ενημερώνεται έτσι το σύνολο* **N** *-> Allowed{j+1,end}.*

*Τα παραπάνω βήματα επαναλαμβάνονται ωσότου η λίστα Allowed είναι πλήρης,περιέχει δηλαδή μια ακολουθία από όλες τις διεργασίες του προβλήματος (μια πλήρη λύση) και ισχύει size(Allowed)={n x m} και size(**N**)= .*

*Η συνάρτηση **ΕκτέλεσεΠράκτορες()** επιστρέφει την λίστα Allowed ως μια πλήρη λύση του πράκτορα και εκτελείται για ολόκληρο των πληθυσμό των πρακτόρων.*

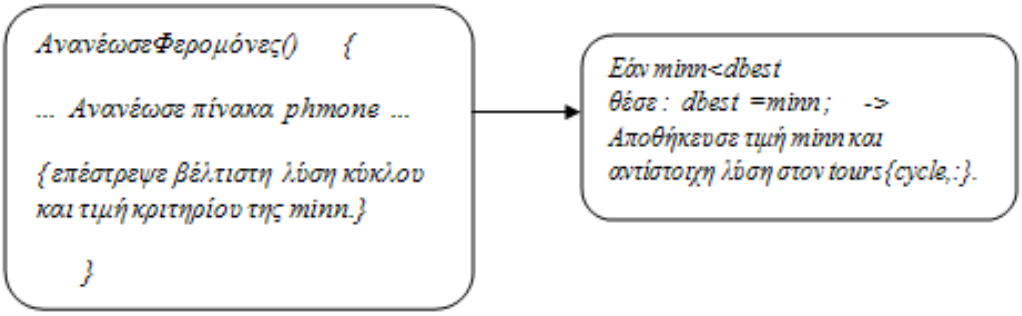
***ΑνανέωσεΦερομόνες():**Η συνάρτηση δέχεται ως όρισμα το σύνολο των λύσεων που επιστρέφει η **ΕκτέλεσεΠράκτορες()**και εκτελεί ολική ανανέωση της φερομόνης στον πίνακα phmone.Ο πίνακας ανανέωνεται στα στοιχεία του που αντιστοιχούν στην βέλτιστη λύση του συνόλου.Θυμίζουμε τον κανόνα ολικής ανανέωσης εξ. 4.2 :*

$$\tau_{ij}(t+n) \leftarrow (1-\alpha) \cdot \tau_{ij}(t) + \alpha \cdot \Delta \tau_{ij}(t,t+n) \quad , \ \forall (i,j) \in T^b$$

με $\Delta \tau_{ij}(t,t+n) = \frac{\quad}{\quad}$ *,* *,με* L^{bs} *το κόστος της βέλτιστης διαδρομής*

Στην περίπτωση λοιπόν πού στοχεύουμε στην ελαχιστοποίηση του C_{max} *το* L^{bs} *θα αντιστοιχεί στο λύση με το ελάχιστο makespan ,εάν στοχεύουμε στο* T_d *θα αντιστοιχεί στην λύση με το ελάχιστο* T_d*.Σε αυτό το σημείο είναι εύλογο να αναρωτηθεί καποιος για τον τρόπο υπολογσιμού των* C_{max} *,*T_d *μιας λύσης.Αν και δεν αναφέρεται στο διάγραμμα ροής θα παρουσιάσουμε τον γενικό αλγόριθμο στην επόμενη ενότητα.*

Σε αυτό το σημείο συγκρίνεται η τιμή minn της βέλτιστης λύσης του συστήματος στον τρέχων κύκλο με την ολικά βέλτιστη τιμή dbest της λύσης που έχει βρεθεί εως αυτό τον κύκλο.Εάν ισχύει minn< dbest ανανεώνεται η τιμή dbest και η αντίστοιχη λύση αποθηκέυεται σε πίνακα tours:



Ολοκληρώνοντας σε περίπτωση που ικανοποιήται το κριτήριο τερματισμού,ο αριθμός κύκλων maxit καθοριζόμενος από τον χρήστη , το σύστημα τερματίζει επιστρέφοντας την λύση dbest διαφορετικά επαναλαμβάνει την παραπάνω διαδικασία με τις ανανεωμένες τιμές.

Μετά τον τερματισμό του το σύστημα εκτυπώνει τα αντίστοιχα αποτελέσματα, το διάγραμμα Gantt της βέλτιστης λύσης dbest και το διάγραμμα σύγκλισης του συστήματος(πίνακας tours).

5.2.1 Κωδικοποίηση των λύσεων και υπολογισμός των κριτηρίων.

Κωδικοποίηση

Όπως έχουμε ήδη αναφέρει οι λύσεις κάθε πράκτορα αντιπροσωπέυουν ένα χρονοπρόγραμμα ,το οποίο περιγράφεται ως μια ακολουθία όλων των διεργασιών όλων των εργασιών(βλ. Ενότητα 4.2) και κάθε διεργασία στην ακολουθία περιγράφεται από έναν αριθμό $k \in [1, n\text{x}m]$ *.Δοσμένου λοιπόν ενός JSP μεγέθους (* n x m*) ,όπου* n *ο αριθμός των εργασιών και* m *ο αριθμός των μηχανών μια ακολουθία αναπαριστάται ως εξής (Bierwirth,1995):*

$$O=(O_1,O_2,\ldots,O_{n\text{x}m}).$$

,όπου $O_{n\text{x}m}=k$ *ο αριθμός της διεργασίας η οποία ανήκει στην - εργασία.Το χρονοπρόγραμμα τότε λαμβάνεται παίρνοντας με την σειρά, από τα αριστερά πρός τα δεξία όλες τις διεργασίες στο Ο και αναθέτωντας τους ,ενωρίτερους χρόνους έναρξης.Η διαδικασία αυτή περιγράφεται στην επόμενη ενότητα και είναι γνωστή ως μετάθεση μέσω επανάληψης(permutation by repetition),όπου αναλύουμε τον γενικό αλγόριθμο για τον υπολογισμό των κριτηρίων ενός χρονοπρογράμματος.Το πλεονέκτημα της μεθόδου είναι ότι δίνει πάντα εφικτά χρονοπρογράμματα.*

Ένα παράδειγμα μιας έγκυρης και πλήρους λύσης για ένα πρόβλημα 3x3 φαίνεται παρακάτω:

O =

1	7	2	4	5	3	6	8	9

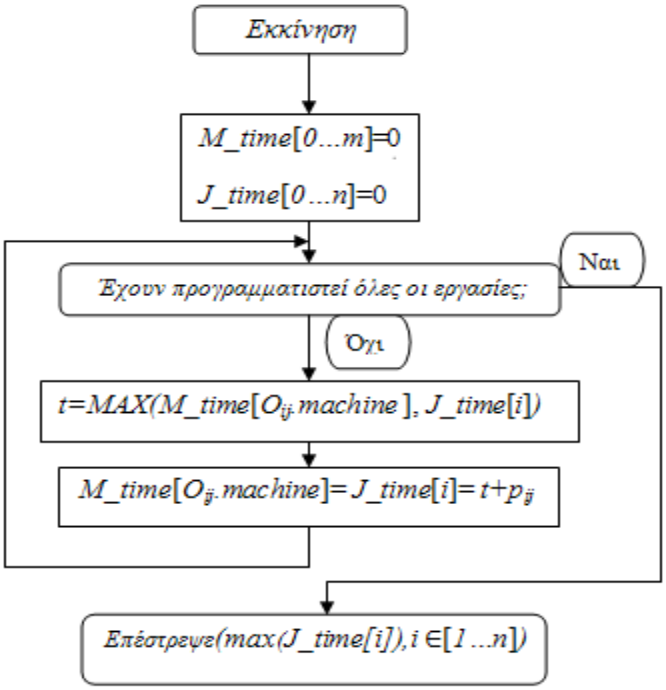
Στην παραπάνω λύση οι διεργασίες 1,2,3 ανήκουν στην 1η εργασία,οι διεργασίες 4,5,6 στη 2η εργασία και οι 7,8,9 στην 3η εργασία.Βλέπουμε στην ακολουθία των διεργασιών πως διατηρούνται οι περιορισμοί μεταξύ των εργασιών της ίδιας εργασίας,πχ για την 1η εργασία ,η 1η διεργασία προγραμματίζεται πριν την 2η ,η 2η πρίν την 3η ,για την 2η εργασία η 4η διεργασία προγραμματίζεται πριν την 5η , η 5η πριν την 6η κλπ.

Υπολογισμός των κριτηρίων

Αφού όλοι οι πράκτορες έχουν ολοκληρώσει από ένα tour πρέπει να υπολογίσουμε

τον makespan κάθε λύσης.Ο υπολογισμός του makespan βασίζεται στην γενική μέθοδο της μετάθεσης μέσω επανάληψης(Christian Bierwirth,2004), όπου αναθέτουμε ενωρίτερους χρόνους έναρξης(ή ενωρίτερους χρόνους περάτωσης) στις διεργασίες της λύσης.Για τις ανάγκες λοιπόν του προβλήματος υλοποιήσαμε τον Αλγόριθμο 5.3 ,όπου $m= O_{ij}.machine, \forall\ i \in [1,n]$, $j \in [1,m]$ *είναι η μηχανή στην οποία προγραμματίζεται η j διεργασία της i-οστης εργασίας(διαθέσιμη από τον πίνακα εισόδου M),* M_time *και* J_time *διανύσματα ,όπου στον* $M\_time(m)$ *κρατάμε τον ενωρίτερο χρόνο στον οποίο η μηχανή m είναι διαθέσιμη ,ενώ στον* $J\_time(i)$ *τον ενωρίτερο χρόνο περάτωσης της i-οστης εργασίας αντίστοιχα και* p_{ij} *ο χρόνος επεξεργασίας της(διαθέσιμος από τον πίνακα εισόδου T) .Λαμβάνοντας τις διεργασίες στην ακολουθία* O *από τα αριστερά πρός τα δεξιά, κάθε στιγμή μια διεργασία j έχει δύο άμεσους προκατόχους -της εργασίας και της μηχανής- άρα ο ενωρίτερος χρόνος περάτωσής της στο χρονοπράγραμμα θα είναι* $\{ MAX(M\_time(m)\ ,\ J\_time(i))\ +\ p_{ij} \}$*.Υπολογίζοντας επαναληπτικά τους ενωρίτερους χρόνους περάτωσης για το σύνολο των διεργασιών όλων των εργασιών βρίσκουμε την μέγιστη από αυτές -* $max(J\_time(i))$ *,δηλαδή τον μέγιστο από τους χρόνους περάτωσης των τελευταίων διεργασιών από κάθε εργασία ,που θα αντιστοιχεί στο* C_{max} *του χρονοπρογράμματος.*

Αλγόριθμος 5.3 :Διάγραμμα ροής για τον υπολογισμό του makespan ενός χρονοπρογράμματος.



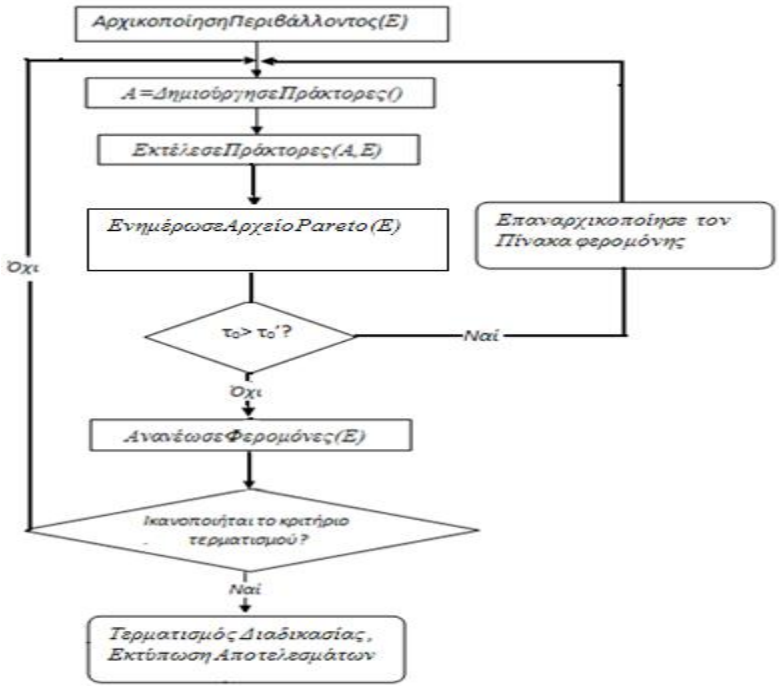
Μετά το πέρας του αλγορίθμου υπολογίζουμε της total tardiness του προγράμματος ως εξής:

,όπου $DDATE(i)$ *,* $i=1,...n$ *, διάνυσμα που περιέχει τον χρόνο παράδοσης(due date) της i-οστης εργασίας(διαθέσιμο από τον διάνυσμα εισόδου DDate).*

5.3 Ανάλυση και Σχεδίαση του Πολυκριτηριακού σύστηματος

Στα πλαίσια αυτής της διπλωματικής επιλύουμε το πρόβλημα του πολυκριτηριακού Job-Shop Scheduling,JSP ,το οποίο παρουσιάστηκε αναλυτικά στην Ενότητα 3.1.Θεωρούμε δύο αντικειμενικά κριτήρια ,ονομαστικά 1) το makespan,C_{max} *και 2) τον συνολικό χρόνο βραδύτερης περάτωσης,*T_d *τα οποία επιδιώκουμε να ελαχιστοποίησουμε ταυτόχρονα.*

Το πρόγραμμα είναι γραμμένο στο περιβάλλον της πλατφόρμας MATLAB και τα δεδομένα του προβλήματος, περιλαμβάνουν τα .txt αρχεία που αναπαριστούν ένα συγκεκριμένο στιγμιότυπο του Job-Shop προβλήματος. Τα δεδομένα περιλαμβάνουν ένα πίνακα M μεγέθους n (εργασίες) x m (μηχανές) με τις μηχανές στις οποίες πρέπει να δρομολογηθούν οι διεργασίες και ένα πίνακα T ίδιου μεγέθους με τους χρόνους επεξεργασίας των διεργασιών. Επιπλέον δεδομένα είναι ένα διάνυσμα DDate μεγέθους (1 x n) που περιλαμβάνει τους χρόνους παράδοσης των εργασιών.Παραθέτουμε και πάλι το διάγραμμα ροής του προτεινόμενου σύστηματος που παρουσιάσαμε στην Ενότητα 4.4:



ΑρχικοποίησηΠεριβάλλοντος():.*Το περιβάλλον αντιπροσωπεύεται όπως και στο μονοκριτηριακό σύστημα από έναν πίνακα Graph μεγέθους n(εργασίες) x m(μηχανες).*
Αρχικοποιούνται επίσης μέσα στη συνάρτηση και οι παράμετροι της αλγοριθμικής διαδικασίας ονομαστικά :η τιμή αρχικής φερομόνης τ_0, η τιμή q_0 , οι ρυθμοί εξάτμισης α και ρ ,το Pareto αρχείο P ως κενό και ο αριθμός των κύκλων maxit.

Η τιμή τ_0 αρχικοποιήται βάση της εξ. 4.7: τ_0= 1/N($F^1{}_\psi{}^h$. $F^2{}_\psi{}^h$) ,όπου $F^1{}_\psi{}^h$ και $F^2{}_\psi{}^h$ οι αρχικές εκτιμήσεις του κάθε κριτηρίου.Οι εκτιμήσεις έγιναν βάση του G&T αλγόριθμου.*

ΔημιούργησεΠράκτορες():*Η συνάρτηση καθορίζει τον αρχικό πληθυσμό A των πρακτόρων ,τιμή καθοριζόμενη από εμάς.*

ΕκτέλεσεΠράκτορες():*Η συνάρτηση περιλαμβάνει την λειτουργικότητα του πράκτορα(Εικόνα 5.3) και είναι υπέυθυνη για την κατασκεύη των λύσεων από τον πληθυσμό A των πρακτόρων στο περιβάλλον E. Η δομή της παραπάνω συνάρτησης είναι ίδια με του μονοκριτηριακού συστήματος.*

1. *Για την υλοποίηση της εσωτερικής μνήμης M δημιουργήσαμε μια μοναδική λίστα την οποία ονόμασαμε Allowed η οποία περιέχει τις διεργασίες που έχουν ηδη προγραμματιστεί(tabu), την τρέχουσα διεργασία στην οποία βρίσκεται ο πράκτορας(ΤρέχωνΚόμβος) και τις διεργασίες στις οποίες μπορεί να μεταβεί στην επόμενη επανάληψη της εκτέλεσης(**N**).*

2. *Η συνάρτηση ΕπέστρεψεΠροσάρτημένουςΚόμβους() δέχεται ως όρισμα τον ΤρέχωνΚόμβο , Allowed{j} ,και επιστρέφει το σύνολο **N** από την Allowed.*

3. *Κατόπιν καλούνται οι συναρτήσεις ΕπέστρεψεΦερομόνη() και ΕπέστρεψεΕυρετικήΠληροφορία().Η συνάρτηση ΕπέστρεψεΦερομόνη επιστρέφει τις τιμές φερομόνης τ_{ij} που σχετίζονται με τις ακμές που συνδεόυν τον ΤρέχωνΚόμβο με κάθε διεργασία του συνόλου **N**.Για το λόγο αυτό διατηρεί έναν πίνακα φερομόνης phmone μεγέθους {nxm (διεργασίες)+1} x {nxm (διεργασίες)}. Η {n x m +1} γραμμή αντιπροσωπεύει τις τιμές φερομόνης από τον dummy node πρός τους αρχικές διεργασίες(στήλες) των εργασιών.Κάθε στοιχείο λοιπόν του phmone περιέχει την τιμή φερομόνης που σχετίζεται με την μετάβαση από μια διεργασία σε μια άλλη.*

   Η συνάρτηση ΕπέστρεψεΕυρετικήΠληροφορία() επιστρέφει ανίστοιχα τις τιμές ευρετικής πληρόφορίας n_{ij}^1 και n_{ij}^2 για κάθε κριτήριο:

   - *Ο υπολογισμός της ευρετικής πληροφορία n_{ij}^1 των διεργασίων του συνόλου **N** θεωρώντας το C_{max} γίνεται όπως και πρίν βάση το κριτηρίου του μεγαλύτερου εναπομείναντος χρόνου κατεργασίας (Longest Remaining Processing Time,LRT)*

     Η τιμή της ευρετικής πληροφορίας των διεργασιών σε αυτή την περίπτωση είναι σταθερή και γνωστή a-priori καθόλη την διάρκεια του πειράματος,και την θέσαμε σαν έισοδο στο σύστημα μέσω ενός πίνακα vis μέγεθους ίδιου με τον πίνακα φερομόνης.

   - *Αντίστοιχα για τον υπολογισμό της ευρετικής πληροφορίας n_{ij}^2 για το*

T_d χρησιμοποιήσαμε τον MDD (Modified Due Date) .Η συνάρτηση ΕπέστρεψεΕυρετικήΠληροφορία() υπολογίζει δυναμικά την ευρετική πληροφορία για τους επιτρεπόμενους κόμβους στο **N** *βάση των τιμών τους T και p_j .*

4. *Στη συνέχεια καλείται η συνάρτηση ΕπέλεξεΑκμή() η οποία επιλέγει και επιστρέφει την ακμή που συνδέει τον ΤρέχωνΚόμβο με μια διεργασία στο* **N**.*Δέχεται ως όρισμα τις τιμές t_{ij} , n_{ij}^1 και n_{ij}^2 'υπολογίζει τις πιθανότητες μετάβασης του πράκτορα βάση των κανόνων μετάβασης(εξ. 4.8 , εξ. 4.9). Οι πιθανότητες μετάβασης αποθηκέυονται σε διάνυσμα prob και η συνάρτηση επιλέγει και πάλι την ακμή βάση ενός μηχανισμού roulette wheel selection,[41]).*

5. *Η συνάρτηση ΑνανέωσεΦερομόνη() κατόπιν ανανεώνει τον πίνακα phmone στο στοιχείο που αντιστοιχεί στην επιλεγμένη ακμή βάση του κανόνα τοπικής ανανέωσης(εξ 4.7).*

6. *Η συνάρτηση ΕπέστρεψεΚόμβο () δέχεται ως όρισμα την επιλεγμένη ακμή και επιστρέφει τον νεό ΤρέχωνΚόμβο από το σύνολο* **N.**

7. *Τέλος καλείται η συνάρτηση ΑνανέωσεΜνήμη() με όρισμα τον νεό ΤρέχωνΚόμβο και ανανέωνει όπως και στο μονοκριτηριακό σύστημα την λίστα Allowed.*

*Τα παραπάνω βήματα επαναλαμβάνονται ωσότου η λίστα Allowed είναι πλήρης,περιέχει δηλαδή μια πλήρη ακολουθία από όλες τις διεργασίες του προβλήματος , ισχύει δηλαδή size(Allowed)={n x m} και size(**N**)= .*

*Η συνάρτηση **ΕκτέλεσεΠράκτορες()** επιστρέφει την λίστα Allowed ως μια πλήρη λύση του πράκτορα και εκτελείται για ολόκληρο των πληθυσμό των πρακτόρων.*

***ΕνημέρωσεΑρχείοPareto():**Κατόπιν καλείται η συνάρτηση ΕνημέρωσεΑρχείοPareto(). Δέχεται ως είσοδο τον πληθυσμό των λύσεων που επιστρέφει η **ΕκτέλεσεΠράκτορες()** υπολογίζει τα C_{max} και T_d κάθε λύσης βάση του Αλγόριθμου 5.3 και ενημερώνει το Pareto αρχείο P.Γίνεται online αποθήκευση των λύσεων στο P: Για κάθε λύση s του πληθυσμού εάν κυριαρχεί μιας λύσης στο P την αντικαθιστά μαζί με τις τιμές των κριτηρίων της διαφορετικά αγνοείται.*

*Μετά τον τερματισμό της **ΕνημέρωσεΑρχείοPareto()** ελέγχεται εάν το αρχείο Pareto έχει βελτιωθεί.Υπολογίζεται η νέα τιμή τ_0' βάση της εξίσωσης 4.10 και αν ισχύει $\tau_0' < \tau_0$ επειδή το P ανανεώθηκε ,αρχικοποιήται με τ_0' ο πίνακας phmone και το σύστημα επαναλαμβάνει την διαδικασία καλώντας την συνάρτηση **ΔημιούργησεΠράκτορες()**,διαφορετικά καλείται η **ΑνανέωσεΦερομόνες().**Η **ΕνημέρωσεΑρχείοPareto()** επιστρέφει το αρχείο P.*

***ΑνανέωσεΦερομόνες():**Η συνάρτηση δέχεται ως όρισμα το αρχείο P και εκτελεί ολική ανανέωση της φερομόνης στον πίνακα phmone.Ο πίνακας ανανέωνεται στα στοιχεία που αντιστοιχούν στις μη κυριαρχούμενες λύσεις του P βάση της εξ. 4.5..*

Τέλος σε περίπτωση που ικανοποιήται το κριτήριο τερματισμού maxit, το σύστημα τερματίζει διαφορετικά επαναλαμβάνει την παραπάνω διαδικασία με τις ανανεωμένες τιμές.

Μετά τον τερματισμό του το σύστημα εκτυπώνει τα αντίστοιχα αποτελέσματα, δηλαδή το τελικό Pareto αρχείο P και πίνακα που περιλαμβάνει το μέγεθος του P ανα κύκλο, δηλαδή το πλήθος των μη κυριαρχούμενων λύσεων σε κάθε κύκλο της εκτέλεσης,,

Κεφάλαιο 6 Πειράματα

6.1 Εφαρμογές στο μονοκριτηριακό σύστημα

Στην 1η φάση των πειραμάτων δοκιμάσαμε το μονοκριτηριακό σύστημα σε εφτά κλασσικά προβλήματα χρονοπρογραμματισμού Ft06, Ft10 ,La01, La06, La10, La21 και Orb04 τα οποία αντλήθηκαν απο την OR-Library του J E Beasley,για καθένα από τα οποία το εκτελέσαμε 10 φορές με στόχο το ***makespan****.Τα στιγμιότυπα παρατίθονται στο Παράρτημα Α.Μετά από προκαθορισμένο αριθμό επαναλήψεων ανάλογα και με την "δυσκολία" του benchmark, το σύστημα τερματίζει,ενώ ιδιαίτερα σημαντικό ρόλο στα πειράματα έχει η ρύθμιση των παραμέτρων.Ύστερα απο πειραματική παρατήρηση και εκτέλεση του συστήματος αρκετές φορές για κάθε πρόβλημα, θέσαμε τους ρυθμούς εξάτμισης* $\alpha=\rho=0.1$ *ώστε να εμποδίσουμε την πρόωρη εξάτμισης της φερομόνης στα διάφορα μονοπάτια του γράφου- ο κύριος μηχανισμός που κατευθύνει την αναζήτηση του πράκτορα στον χώρο λύσεων- επίσης θέσαμε* $\beta=1$ *και* $,q_0=0.9$ *για τα πιό "εύκολα" στιγμιότυπα Ft06(6 x 6), La01(10 x 5), La06(15 x 5), La10(15 x 5), ενώ* $q_0=0.8$ *για τα σχετικά πιό δύσκολα Ft10(10 x 10), La21(15 x 10), Orb04(10 x 10).*

*Για την ελαχιστοποίηση του δεύτερου κριτηρίου -****της βραδύτερης περάτωσης****- θέσαμε* $q_0=0.8$ *και* $q_0=0.7$ *για τα εύκολα και δύσκολα προβλήματα αντίστοιχα ,ενώ οι υπόλοιπες παράμετροι παρέμενειναν ίδιες.Τέλος ο αριθμός των πρακτόρων τέθηκε ίσος με 40 σε όλα τα προβλήματα και για τα δύο κριτήρια.*

- *Στον Πίνακα 6.1 φαίνονται οι μέσοι όροι(Average) και τα βέλτιστα(ACS_Best) αποτελέσματα από την εκτέλεση του μονοκριτηριακού συστήματος για την ελαχιστοποίηση του makespan .Επίσης παραθέτουμε τις βέλτιστες θεωρητικές τιμές αυτών των προβλημάτων από κοινού με το σφάλμα των ACS_Best τιμών σε σύγκριση με τις θεωρητικές(Optimal value) και τον μέσο χρόνο σύγκλισης του συστήματος(secs) στην βέλτιστη τιμή του σε κάθε εκτέλεση.*

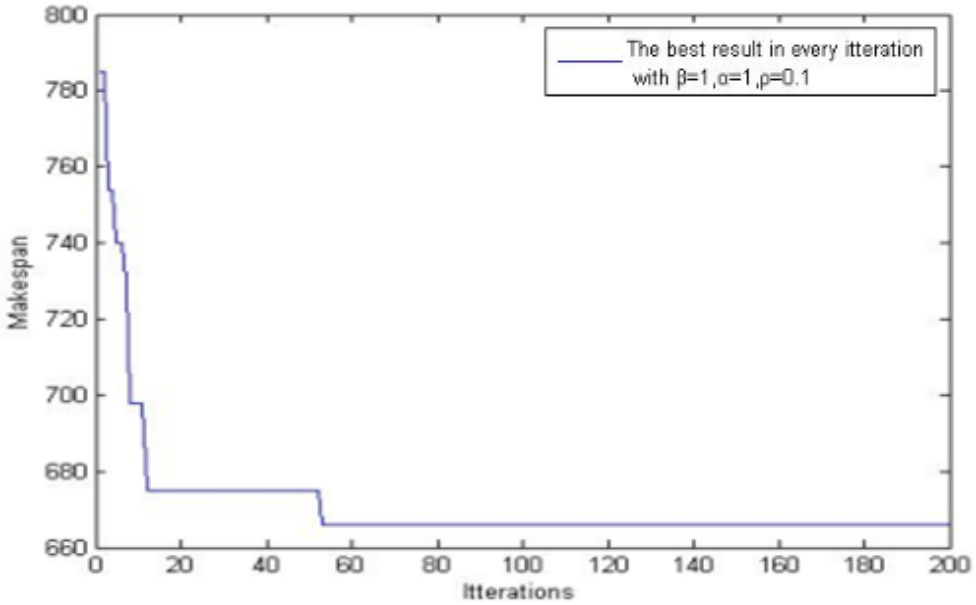
Πίνακας 6.1:Οι μέσες και οι βέλτιστες τιμές 7 προβλημάτων για 10 εκτελέσεις του μονοκριτηριακού συστήματος για την ελαχιστοποιηση του C_{max} ***.***

JSP name	*n*	*m*	*Average*	*ACS_Best*	*Optimal value*	*Error*	*Avg.Time (secs)*
Ft06	*6*	*6*	*56,6*	*55*	*55*	*0%*	*3.7171*
La01	*10*	*5*	*670*	*666*	*666*	*0%*	*10.8334*

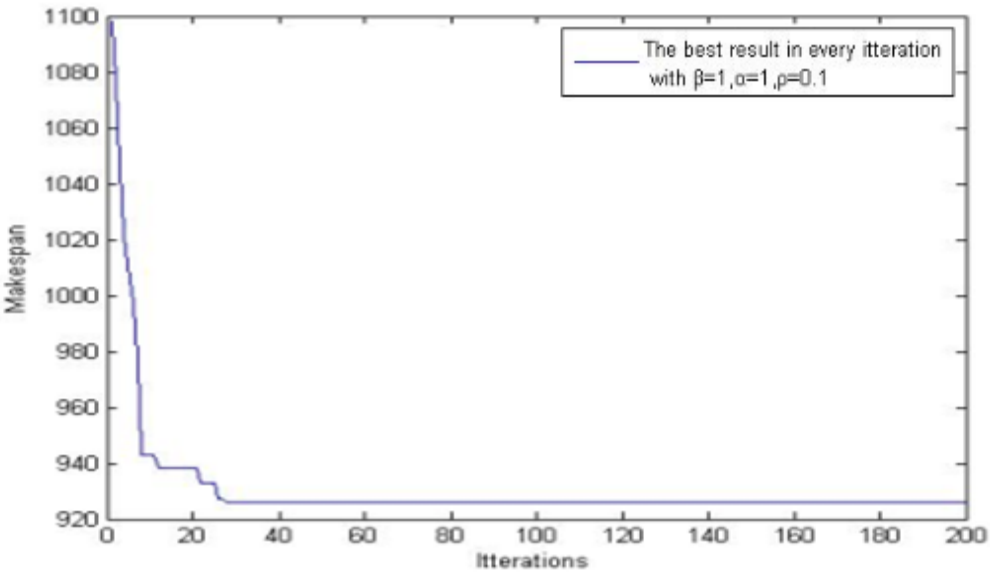
La06	*15*	*5*	*926*	*926*	*926*	*0%*	*21.8662*
La10	*15*	*5*	*958*	*958*	*958*	*0%*	*25.9344*
Ft10	*10*	*10*	*1096,6*	*1062*	*930*	*13,3%*	*94.7002*
La21	*15*	*10*	*1224,7*	*1183*	*1046*	*13,1%*	*170.8661*
Orb04	*10*	*10*	*1117*	*1084*	*1005*	*7,8%*	*87.8358*

Από τον παραπάνω πίνακα παρατηρούμε ότι το σύστημά μας δίνει ικανοποιητικά αποτελέσματα για τα παραπάνω προβλήματα και πλησιάζει την εκάστοτε λύση για ένα ικανοποιητικό αριθμό επαναλήψεων.Για τα Ft06, La01, La06 και La10 βρίσκει τις βέλτιστες τιμές σχεδόν σε κάθε εκτέλεση του, ωστόσο για τα πιο δύσκολα στιγμιότυπα Ft10, La21, Orb04 δεν καταφέρνει να βρεί τα βέλτιστα αποτελέσματα , αλλά τα προσεγγίζει σε ικανοποιητικό βαθμό.

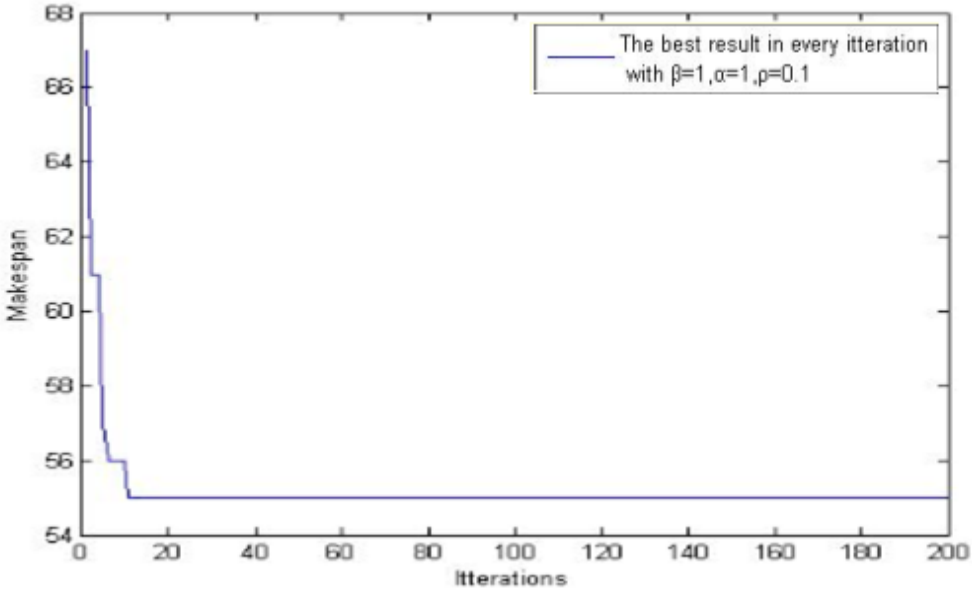
Οι Εικόνες 6.1 έως 6.3 δείχνουν την ταχύτητα σύγκλισης του για τα προβλήματα Ft06, La01, La06 σε 200 επαναλήψεις του αλγόριθμου.Βλέπουμε οτι συγκλίνει σχετικά γρήγορα-στις πρώτες 50 επαναλήψεις(κύκλους)- ενώ είναι αξιοσημείωτο ότι για β=0 το σύστημα δίνει ακομα καλύτερα αποτελέσματα σε κάποια προβλήματα ,όπως στα Ft06 και La06.



Εικόνα 6.1:*Η καλύτερη λύση σε κάθε επανάληψη για το La01 πρόβλημα.*



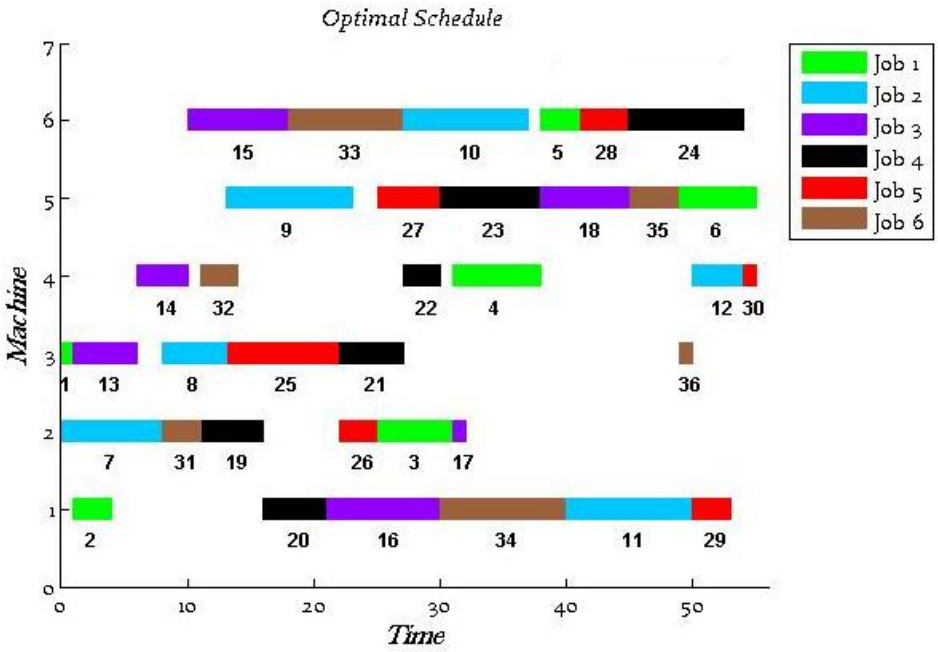
Εικόνα 6.2:*Η καλύτερη λύση σε κάθε επανάληψη για το La06 πρόβλημα.*



Εικόνα 6.3:*Η καλύτερη λύση σε κάθε επανάληψη για το Ft06 πρόβλημα.*

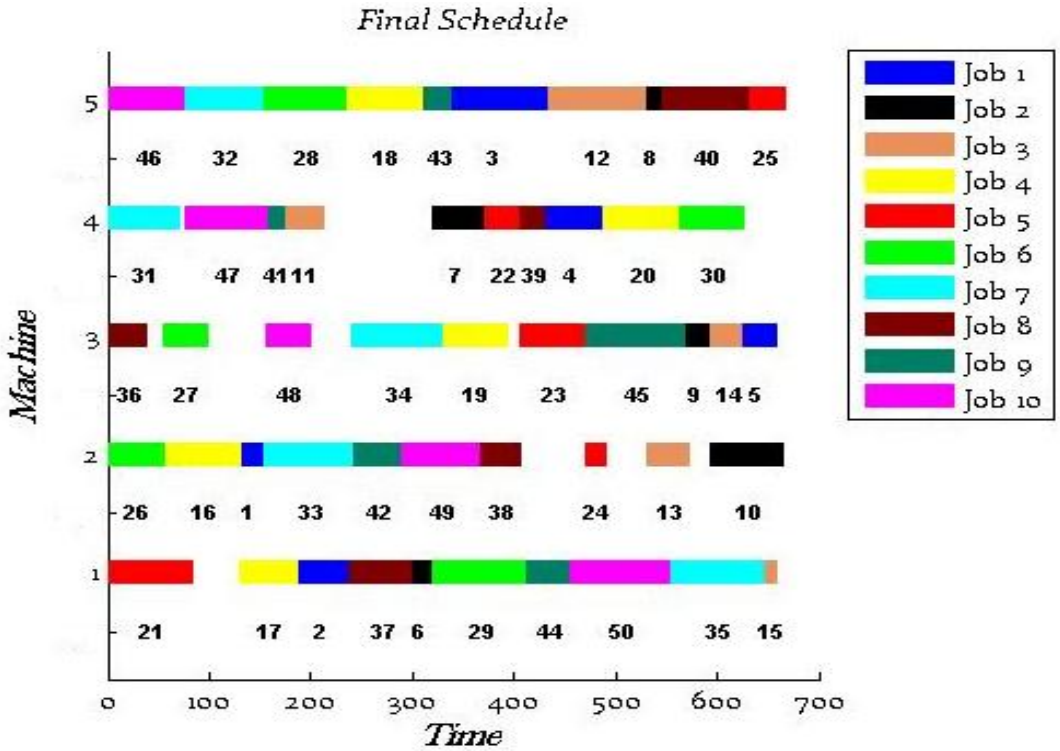
Ft06

Πιο συγκεκριμένα στο πρόβλημα Ft06 (6x6) βρέθηκε 0% σφάλμα από την θεωρητική τιμή του makespan που είναι 55, ενώ το σύστημα εκτελέστηκε για 200 επαναλήψεις(κύκλους). Η μέση τιμη Average αποκκλίνει ελάχιστα από την ACS_Best και το σύστημα συγκλίνει σχετικά γρήγορα.Παρακάτω παρατίθεται το διάγραμμα Gantt της υπολογιζόμενης λύσης (C_{max}=55):



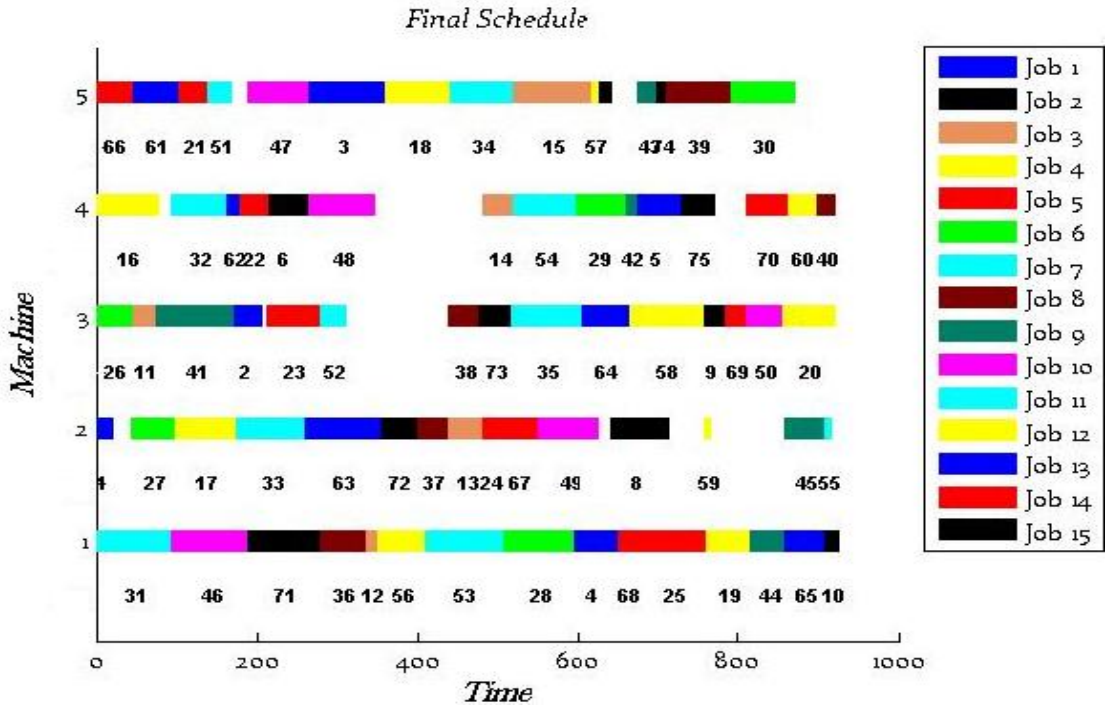
La01

Για το πρόβλημα La01 (10x5) βράθηκε 0% σφάλμα από την θεωρητική τιμή του makespan που είναι 666, για 200 επαναλήψεις του συστήματος.Η μέση τιμή Average έχει επίσης μικρή απόκκλιση από την ACS_Best και το σύστημα συγκλίνει σχεδόν πάντα στη βέλτιστη τιμή στις πρώτες 60 επαναλήψεις.Το διάγραμμα Gantt της βέλτιστα υπολογιζόμενης λύσης (Cmax=666) δίνεται παρακάτω:



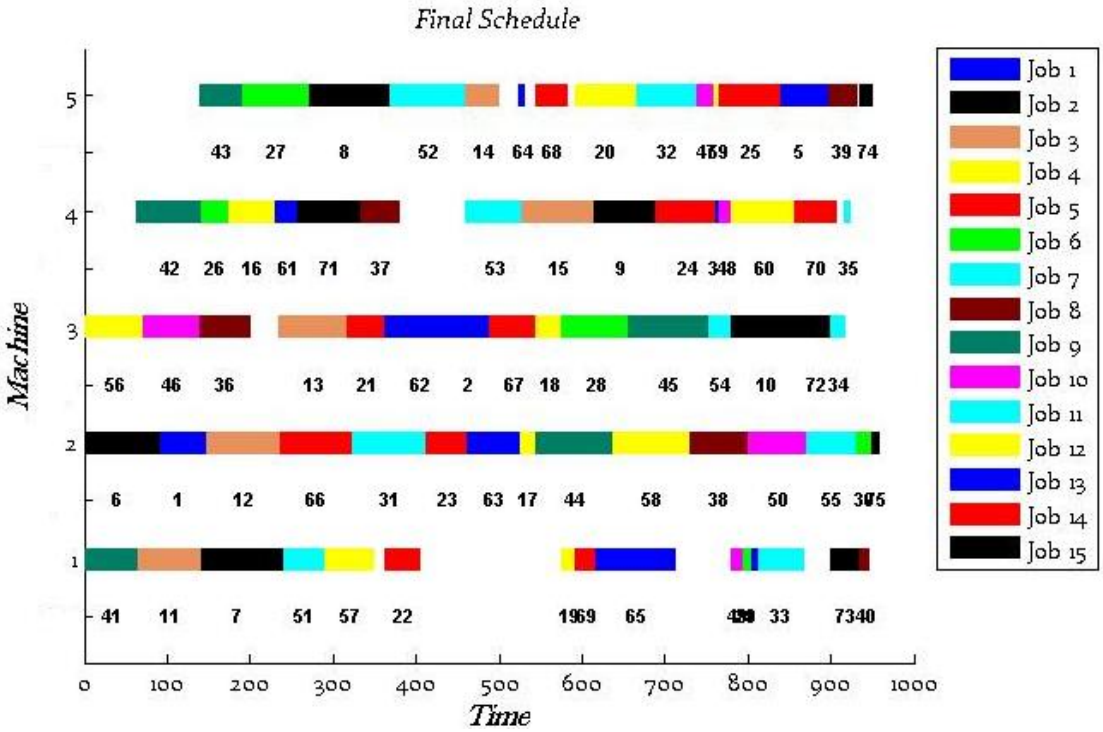
La06

Στο πρόβλημα La06 (15x5) η τιμή του makespan είχε σφάλμα 0% από την θεωρητική(Cmax=926) για 200 επαναλήψεις ,ενώ το σύστημα συγκλίνει σε κάθε εκτέλεση στο βέλτιστο makespan,δηλαδή ACS_Best= Average. Το διάγραμμα Gantt του χρονοπρογράμματος για Cmax=926 δίνεται παρακάτω:



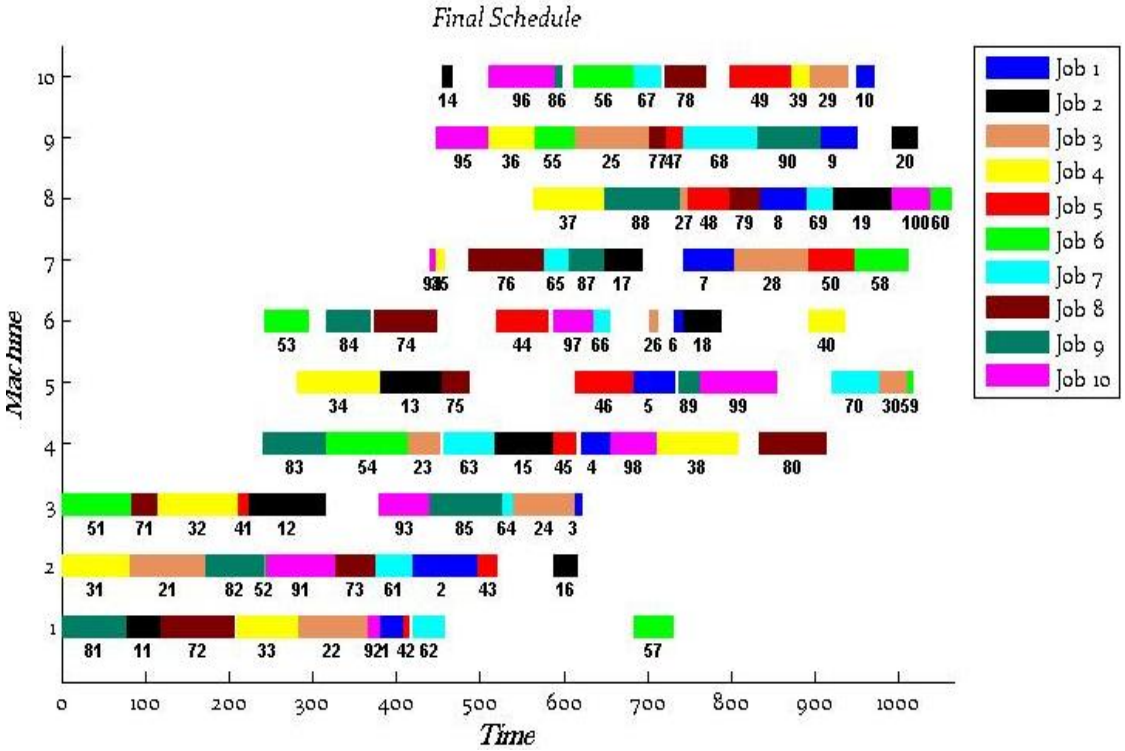
La10

Για το πρόβλημα La10 (10x5) βρέθηκε 0% σφάλμα από την θεωρητική τιμή του makespan που είναι 958, για 200 επαναλήψεις του αλγόριθμου.Η μέση τιμή Average έχει μηδενική απόκκλιση από την ACS_Best και το σύστημα συγκλίνει κάθε φορά στη βέλτιστη τιμή στις πρώτες 100 επαναλήψεις.Το διάγραμμα Gantt της βέλτιστης υπολογιζόμενης λύσης (Cmax=958) φαίνεται παρακάτω:



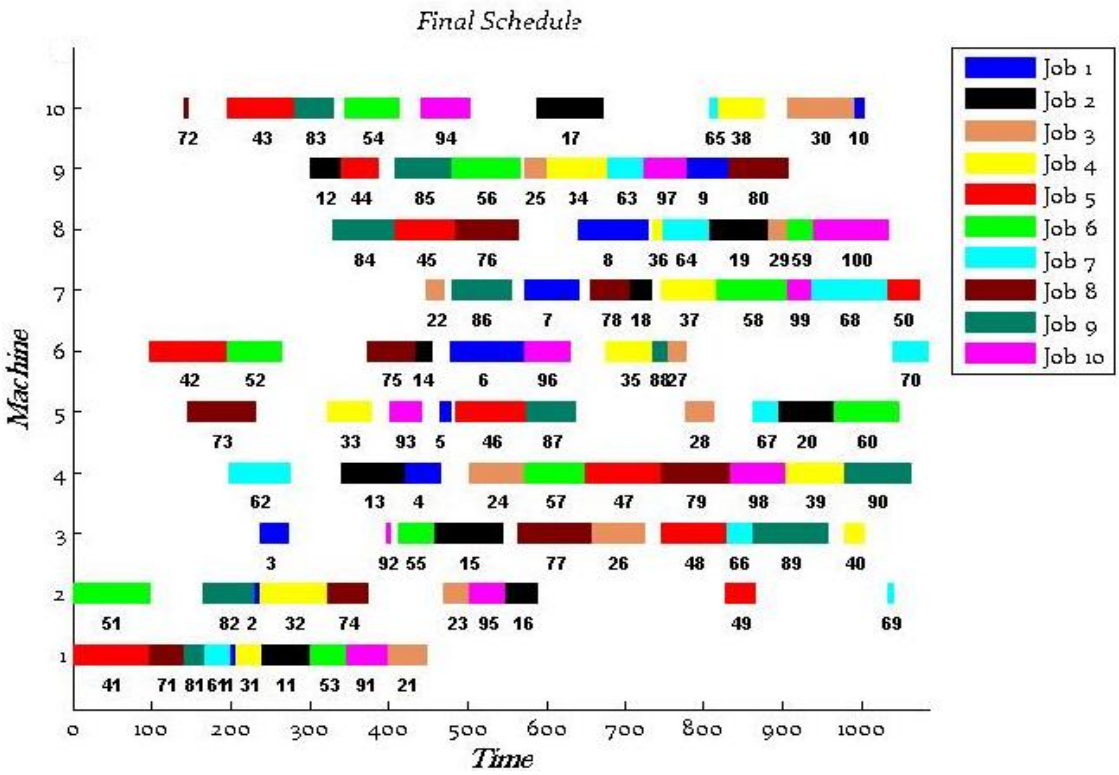
Ft10

Για το πρόβλημα ft10 (10x10) βρέθηκε 13,3% σφάλμα από την θεωρητική τιμή του makespan που είναι 937, για 400 επαναλήψεις του αλγόριθμου.Η μέση τιμή Average έχει αυξημένη απόκκλιση από την ACS_Best και το σύστημα συγκλίνει στη βέλτιστη τιμή του που είναι 1062, ύστερα από περίπου 250 επαναλήψεις.Το διάγραμμα Gantt της βέλτιστα υπολογιζόμενης λύσης (Cmax=1062) φαίνεται παρακάτω:



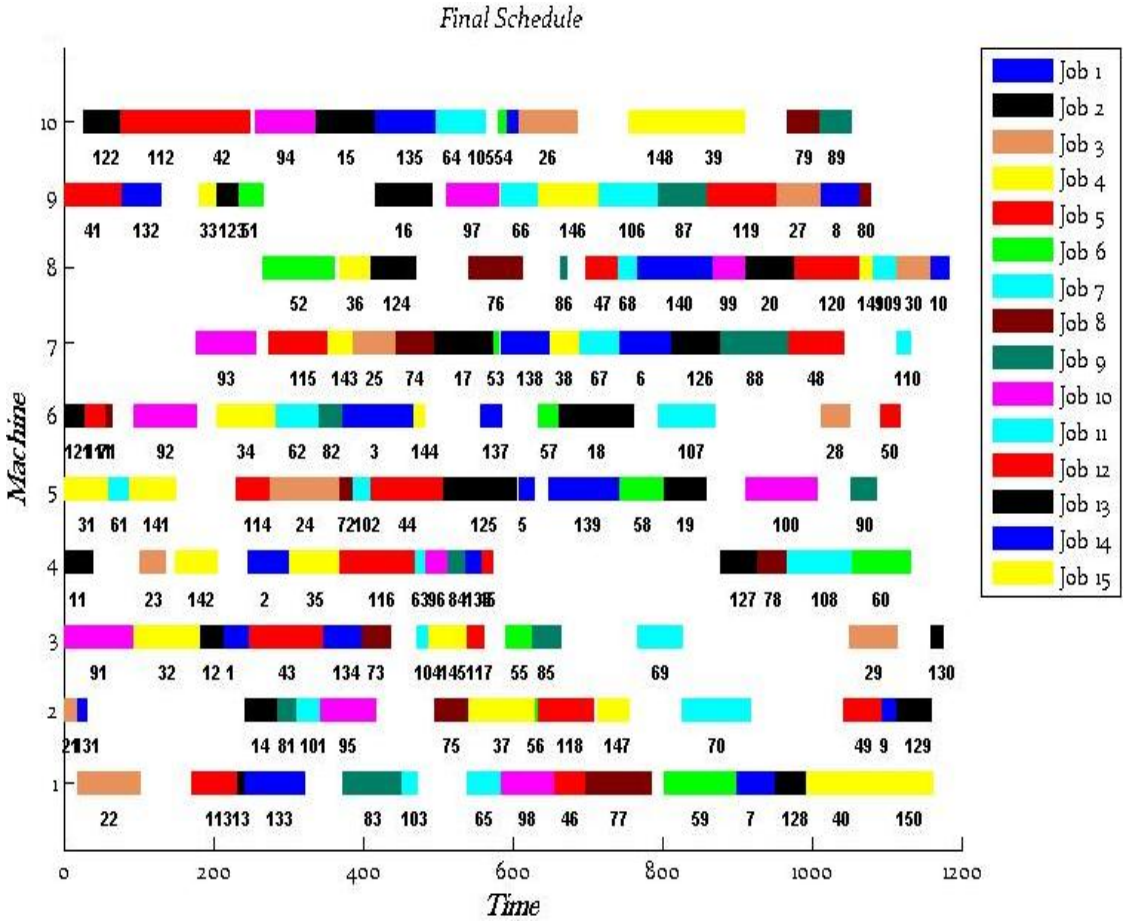
Orb04

Για το άλλο τετραγωνικό πρόβλημα Orb04 (10x10) βρέθηκε 7,8% σφάλμα από την θεωρητική τιμή του makespan που είναι 1005, για 400 επαναλήψεις του αλγόριθμου.Η μέση τιμή Average έχει επίσης αυξημένη απόκκλιση από την ACS_Best και το σύστημα συγκλίνει στη βέλτιστη τιμή του που είναι 1084, σε περίπου 350 επαναλήψεις.Παρατίθεται παρακάτω το διάγραμμα Gantt της βέλτιστα υπολογιζόμενης λύσης (Cmax=1084):



La21

Τέλος για το πρόβλημα La21(15x10) με σφάλμα που αγγίζει το 13,1% από την βέλτιστη θεωρητική τιμή που είναι 1046 ,το σύστημα υπολόγισε το παρακάτω χρονοπρόγραμμα με Cmax=1183 για 200 επαναλήψεις σε κάθε εκτέλεση.Το χρονοδιάγραμμα της βέλτιστης λύσης (Cmax=1183) δίδεται παρακάτω:



Παρατηρήσεις

Όπως σημειώθηκε παραπάνω προσεγγιστικές μεθόδοι όπως το μονοκριτηριακό σύστημά μας δεν εγγυούνται να λύσουν ένα Π.Σ.Β όπως το JSP βέλτιστα, παρά να προσεγγίσουν την βέλτιστη λύση σε ένα λογικό χρονικό διάστημα.Σύμφωνα λοιπόν με τα παραπάνω αποτελέσματα μπορούμε να εξάγουμε τα εξης συμπεράσματα:

1. *Ένα στιγμιότυπο του JSP είναι εύκολο να λυθεί όταν οι διαστάσεις του είναι ορθογώνιες(πχ. $n \geq 3m$).Για αυτά τα προβλήματα (La06 ,La10) το σύστημα υπολογίζει τις καλύτερες "γνωστές" τιμές σε κάθε εκτέλεση του,και επίσης θα ισχύει Average= Acs_Best.*
2. *Καθώς τα στιγμιότυπα γίνονται πιο τετραγωνικά ($n \rightarrow m$) και μεγαλύτερα (50→150) σε διαστάσεις βλέπουμε οτι γίνεται δυκολότερο να προσεγγίσουμε τις βέλτιστες τιμές .*

Σε σύγκριση με τα (La06 ,La10) μπορούμε να υπολογίσουμε την βέλτιστη τιμή , ωστόσο όχι σε κάθε έκτελεση του συστήματος(πχ. La01, $n=2m$). Καθώς τα στιγμιότυπα γίνονται μεγαλύτερα και πιο τετραγωνικά (πχ. La21 - $n=1.5m$, 150 διεργασίες) δεν είμαστε σε θέση να υπολογίσουμε τις βέλτιστες τιμές και

ταυτοχρόνως η Average τιμή αποκκλίνει σημαντικά από την Acs_Best.Τέλος τα πιο δύσκολα να λυθούν στιγμιότυπα είναι αυτά τετραγωνικών διαστάσεων, πχ. Ft10 , Orb04 (10x10). Δεν είναι δυνατό να βρεθεί το βέλτιστο makespan , οι τιμές Average ,Acs_Best διαφέρουν σε μεγάλο βαθμο και επίσης οι λύσεις των πρακτόρων σε κάθε κύκλο αποκκλίνουν σημαντικά σε σύγκριση με τα πιο ορθογώνια στιγμιότυπα.

Σύγκριση Αποτελεσμάτων

Η απόδοση του συστήματος συγκρίθηκε με τα αποτελέσματα ενός γενετικού αλγόριθμου και τριών PSO (Particle Swarm optimization) αλγόριθμων οι οποίοι έχουν εφαρμοστεί στο JSP.

Ο γενετικός αλγόριθμος του Tamer F. Abdelmaguid ερευνά την απόδοση ενός γενετικού αλγόριθμου στο JSP βάση 6 διαφορετικών σχεδιών απεικόνισης ενός χρονοπρογράμματος:

- *Στηριζόμενα στις διεργασίες ,OB*
- *Στηριζόμενα σε τυχαία ταξινόμηση προτιμήσεων,RK*
- *Στηριζόμενα στην λίστα προτιμήσεως,PL*
- *Στηριζόμενα στις μηχανές,MB*
- *Στηριζόμενα στις εργασίες ,JB*

Ο αλγόριθμος εκτελέστηκε 5 φορές για 300 επαναλήψεις (Generations) σε κάθε benchmark.Στον Πίνακα 6.2 παρουσιάζονται τα αποτελέσματα του αλγόριθμου για τα benchmarks που μας ενδιαφέρουν,παρατίθονται δηλ.οι βέλτιστη λύση Best και η μέση τιμή Avg. των βέλτιστων λύσεων από όλες τις εκτελέσεις.

Prob.	Size*	No. of Operations	Best known lower bound	GA Representation**											
				OB		RK		PL		PR		MB		JB	
				Best	Avg.	Best	Avg.	Best	Avg.	Best	Avg.	Best	Avg.	Best	Avg.
la01	5 x 10	50	666 (opt.)	666	688.6	666	682	675	701	671	684.6	666	666	700	720.6
la06	5 x 15	75	926 (opt.)	926	926	926	934.8	926	931.8	926	937	926	926	941	966.2
la10	5 x 15	75	958 (opt.)	958	958	958	958	958	967	958	958.2	958	958	987	1004.8
la21	10 x 15	150	1040	1224	1289.6	1230	1286.4	1353	1426.4	1189	1212.2	1105	1120	1285	1310.2
orb4	10 x 10	100	1005 (opt.)	1049	1110.2	1091	1163.4	1158	1252.4	1137	1158.8	1028	1040.6	1202	1234.2

Πίνακας 6.2:Αποτελέσματα του γενετικού αλγόριθμου του Tamer F. Abdelmaguid, 2010 σε 6 standar benchmarks, χρησιμοποιώντας 6 διαφορετικά σχέδια απεικόνισης ενός χρονοπρογράμματος παραγωγής.

Στον Πίνακα 6.3 παρουσιάζονται τα αποτελέσματα από την εφαρμογή ενος standar PSO και ενός βελτιωμένου IPSO(Improved PSO) αλγόριθμου σε δύο benchmarks, YAN Ping et al ,2011..Ο IMPSO αλγόριθμος χρησιμοποιεί νέες μεθόδους κωδικοποίησης και αποδικωποίησης των λύσεων , ενώ έχει ενσωματωθεί και μια ευρετική μέθοδος τοπικής αναζήτησης ώστε να βελτιώθεί περαιτέρω η διαδικασία αναζήτησης του ολικού βέλτιστου.Ο αλγόριθμος εκτελέστηκε 10 φορές σε κάθε benchmark ,χρησιμοποιεί πληθυσμό 30 σωματιδίων και τερματίζει ύστερα από υπολογισμό 30000 λύσεων(χρονοπρογραμμάτων).Ο Πίνακας περιλαμβάνει την βέλτιστη λύση Best που βρέθηκε και την μέση τιμή Average των βέλτιστων λύσεων από όλες τις εκτελέσεις.

Instance	*n*	*m*	Best known	PSO			IPSO		
				Best	worst	Average	Best	worst	Average
FT06	6	6	55	55	62	58	55	55	55
LA01	10	5	666	668	710	687.2	666	675	668.1

Πίνακας 6.3:Αποτελέσματα σε 2 benchmarks των standar PSO και IPSO αλγόριθμων,YAN Ping et al ,2011.

*Ο Πίνακας 6.4 περιλαμβάνει τα αποτελέσματα από την εφαρμογή ενος standar PSO και ενός υβριδικού PSO αλγόριθμου(Hybridised PSP,HPSO) στο JSP.Ο HPSO αλγόριθμος αναθέτει την εξερέυνηση του χώρου αναζήτησης στο σωματίδιο με την βέλτιστη στρατηγική διατήρησης και χρησιμοποιεί τον αλγόριθμο Taboo Search για να ενισχύσει την τοπική αναζήτηση.Ο αλγόριθμος εκτελέστηκε 10 φορές για κάθε benchmark,χρησιμοποιεί ένα πληθυσμό |O|*70% σωματιδίων και τερματίζει ύστερα από |O|*70% επαναλήψεις,όπου |O| ο αριθμός των διεργασιών του εκάστοτε προβλήματος.Παρατίθονται ομοίως τα αποτελέσματα της βέλτιστης λύσης Optimum που βρέθηκε και η μέση τιμή Average value των βέλτιστων λύσεων από όλες τις εκτελέσεις σε κάθε benchmark.*

Problem	Optimum Makespan	PSO			HPSO		
		Optimum	*Average valu*	*time/S*	*Optimum*	*Average valu*	*time/S*
FT10(10×10)	930	977	996.9	24.88	930	945.2	37.49
LA21(15×10)	1046	1254	1281.6	28.50	1078	1099.0	200.77

Πίνακας 6.4: Αποτελέσματα σε 2 benchmarks των standar PSO και HPSO αλγόριθμων, Xiaoyu Song et al ,2009.

Συγκρίνοντας τα αποτελέσματα του συστήματος μας (Πίνακας 6.1)με τον γενετικό ,για το la01 στιγμιότυπο βρίσκει την βέλτιστη τιμή του makespan(666) όπως και ο γένετικός, αλλά προσεγγίζει σε μεγαλύτερο βαθμό την βέλτιστη τιμή σε κάθε εκτέλεση του ,διατηρώντας μικρότερη Avg τιμή από όλες τις παραλλάγές του γενετικού ,εκτός της MB απεικόνισης.Υπερισχύει επίσης για το συγκεκριμένο στιγμιότυπο του standar PSO των YAN Ping et al ο οποίος δεν καταφέρνει να συγκλίνει στη βέλτιστη τιμή ,ενώ επιτυγχάνει ελαφρώς μεγαλύτερη Avg τιμή(670) από τον IMPSO(668.1).Για τα στιγμιότυπα la06 και la10 τα αποτελέσματα συγκρίση του γενετικού είναι παρόμοια με παραπάνω ,ενώ για το «δύσκολο» la21 στιγμιότυπο το σύστημά μας ξεπερνά ευθέως όλες τις παραλλαγές του γενετικού εκτός της MB απεικόνισης.Έπιπλέον έχει καλύτερη απόδοση από τον standar PSO των Xiaoyu Song et al,αλλά υστερεί έναντι του HPSO . Βλέπουμε ότι η ενσωμμάτωση μεθοδών τοπικής αναζήτησης όπως στην περίπτωση του HPSO αλγόριθμου βελτιώνει αισθητά την ποιότητα των λύσεων και βοηθάει τον αλγόριθμο να συγκλίνει σε σχεδόν βέλτιστες λύσεις και σε δύσκολα στιγμιότυπα όπως στην περίπτωση του διάσημου ft10.Ο HPSO καταφέρνει να βρεί την βέλτιστη τιμή(930) για το συγκεκριμένο πρόβλημα συγκλίνοντας κατα μέσο όρο σε μια τιμή 945 σε κάθε εκτέλεση ,υπερισχύοντας της απόδοσης του συτήματός μας(Best=1062,Avg=1096,6).Τέλος για το άλλο «δύσκολο» πρόβλημα Orb04 το προτεινόμενο σύστημα μας συγκρίθηκε με τον γενετικό και παρουσιάζει καλύτερα αποτελέσματα σε όλες τις περιπτώσεις εκτός των OB και MB απεικονίσεων.

- *Στον **Πίνακα 6.5** φαίνονται τα αποτελέσματα από την εκτέλεση του μονοκριτηριακού συστήματος για την ελαχιστοποίηση της T_d.*

JSP name	***n***	***m***	***Average***	***ACS_Best***	***G&T Algorithm***
Ft06	*6*	*6*	*85,3*	*68*	*118*
La01	*10*	*5*	*2158,9*	*2068*	*3146*
La06	*15*	*5*	*5577,7*	*5366*	*6369*
La10	*15*	*5*	*5657,3*	*5417*	*6394*
Ft10	*10*	*10*	*3078,8*	*2842*	*3338*
La21	*15*	*10*			

Orb04	10	10	3747,1	3259	4424

Πίνακας 6.5:Οι μέσες και οι βέλτιστες τιμές 7 προβλημάτων για 10 εκτελέσεις του συστήματος για την ελαχιστοποίηση του T_d .

Για την εξαγωγή των χρόνων παράδοσης(due dates) των εργασιών χρησιμοποιήσαμε την TWK(total work content,Baker (1984)) μέθοδο,μια χρήσιμη μέθοδο ανάθεσης χρόνων παράδοσης που επιτυγχάνει την καλύτερη απόδοση σε προβλήματα που σχετίζονται με την ελαχιστοποίηση της βραδύτητας.Η TWK είναι μια στατική μέθοδος ανάθεσης που λαμβάνει υπόψη τα χαρακτηριστικά κάθε εργασίας(χρόνους επεξεργασίας) και περιορίζει στον ίδιο βάθμο τους χρόνους παράδοσης όλων των εργασιών.Ο χρόνος παράδοσης της i-οστης εργασιας θα είναι:

$$d_i = r_i + k\,x \qquad (6.5)$$

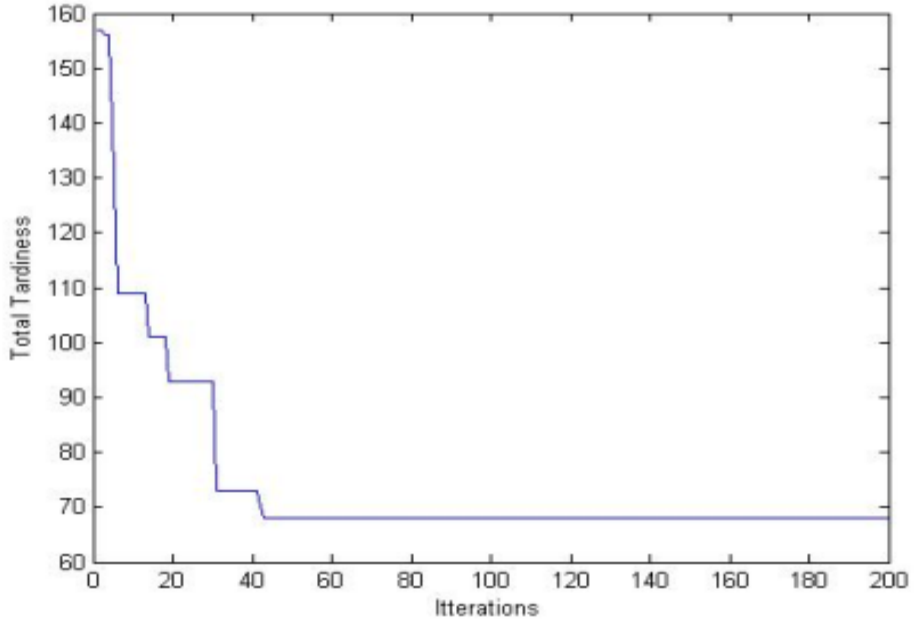
,όπου r_i και d_i οι χρόνοι άφιξης και παράδοσης της εργασίας i στο σύστημα , p_{ij} ο χρόνος επεξεργασίας της διεργασίας O_j και k συντελεστης που καθορίζει το πόσο περιοριστικός θα είναι ο χρόνος παράδοσης.Όσο μεγαλύτερη η τιμή του k τόσο πιό χαλαρός ο χρόνος παράδοσης.Επίσης θεωρώντας ότι όλες οι εργασίες σε ένα job shop μπορούν να αρχίσουν την επεξεργασία τους την χρονική στιγμή 0 ,οι χρόνοι έναρξης r_i των εργασιών θα είναι μηδέν.Για την τιμή του k ανάμεσα στις επιλογές για περιοριστικούς ,μέτριους και χαλαρούς χρόνους παράδοσης(k= 1, 1.5 και 2 αντίστοιχα) επιλέξαμε k=1,άρα ο χρόνος παράδοσης γίνεται : $d_i =$,δηλαδή θα είναι ο συνολικός χρόνος επεξεργασίας της εργασίας.

Επιπλέον εφαρμόσαμε σαν συγκριτικό μέτρο για το σύστημά μας τον $G\&T$ αλγόριθμο(εν. 5.2) σε κάθε benchmark.

Συγκεκριμένα για κάθε στιγμιότυπο θα έχουμε:

Ft06

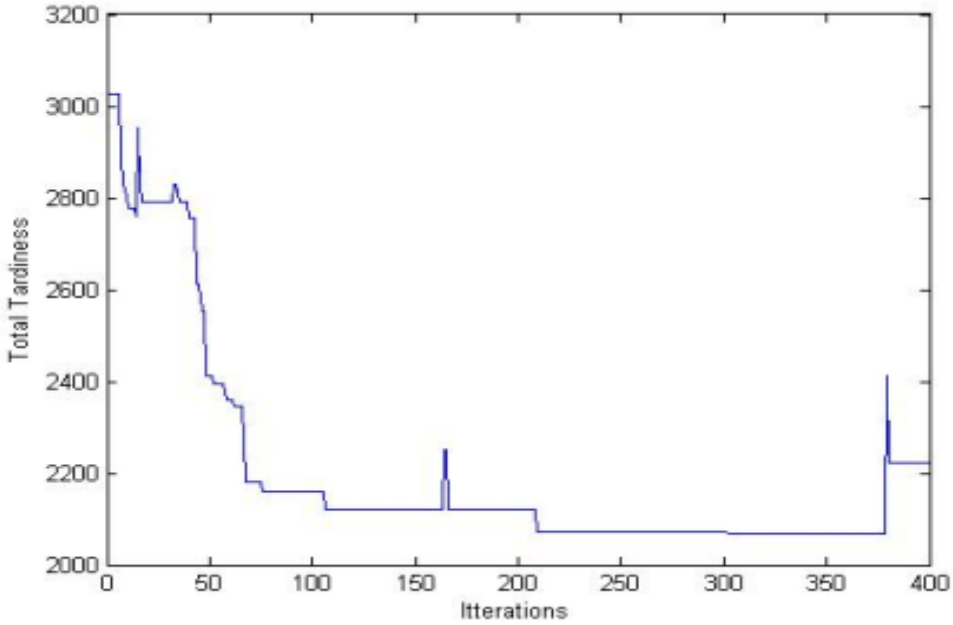
Για το ft06 πρόβλημα το σύστημα μας υπολογίζει βέλτιστη τιμή T_d =68 ,αρκετά χαμηλότερη από τον $G\&T$ αλγόριθμο που είναι 118, για 200 επαναλήψεις.Το σύστημα συγκλίνει γρήγορα κάθε φορά(Εικόνα {a}) , ωστόσο τα αποτελέσματα διαφέρουν σε κάθε εκτέλεση παρουσιάζοντας σημαντική απόκκλιση.



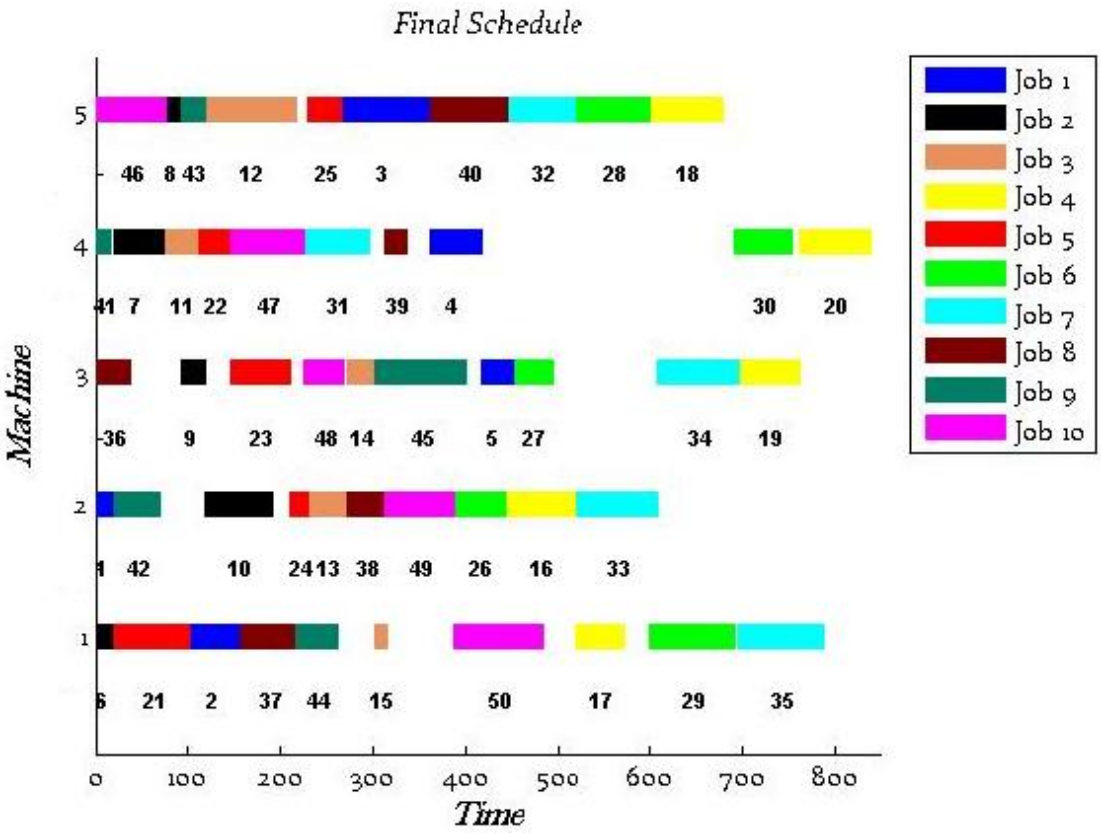
(α)

La01

Για το La01 πρόβλημα υπολογίζει βέλτιστη τιμή T_d =2068 σε 400 επαναλήψεις βελτιώνοντας κατα πολύ το αποτέλεσμα του G&T αλγόριθμου(3146).Η ταχύτητα σύγκλισης και το διάγραμμα Gantt της βέλτιστης λύσης φαίνονται παρακάτω:



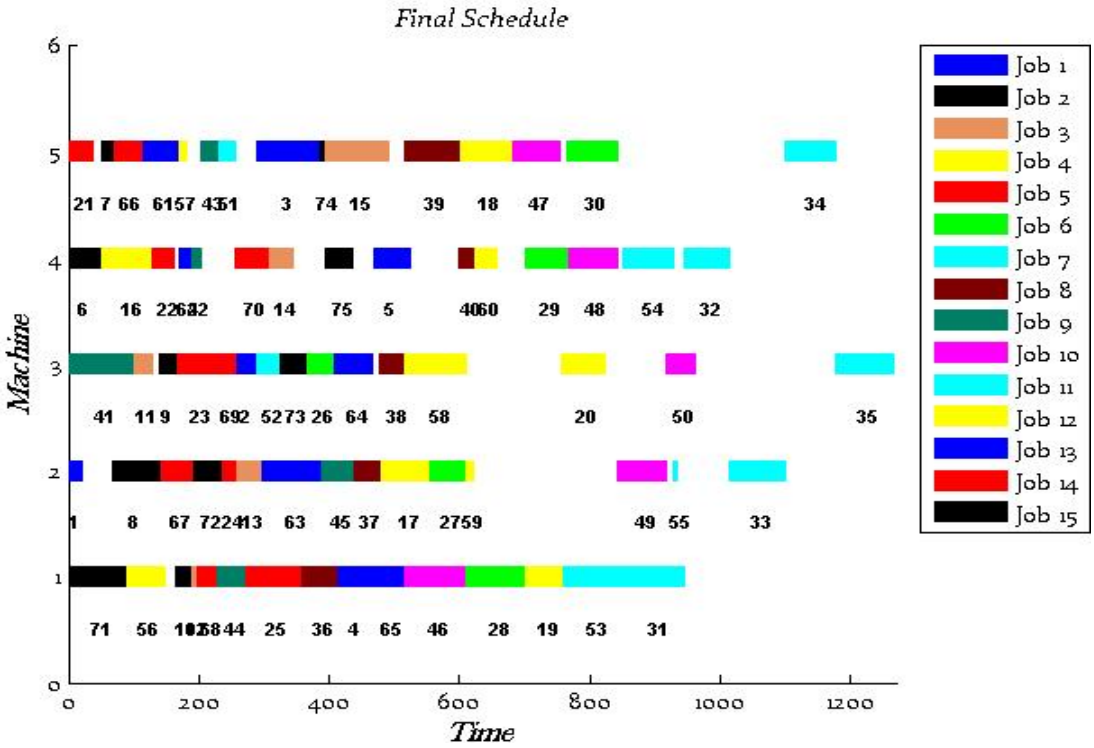
(α)



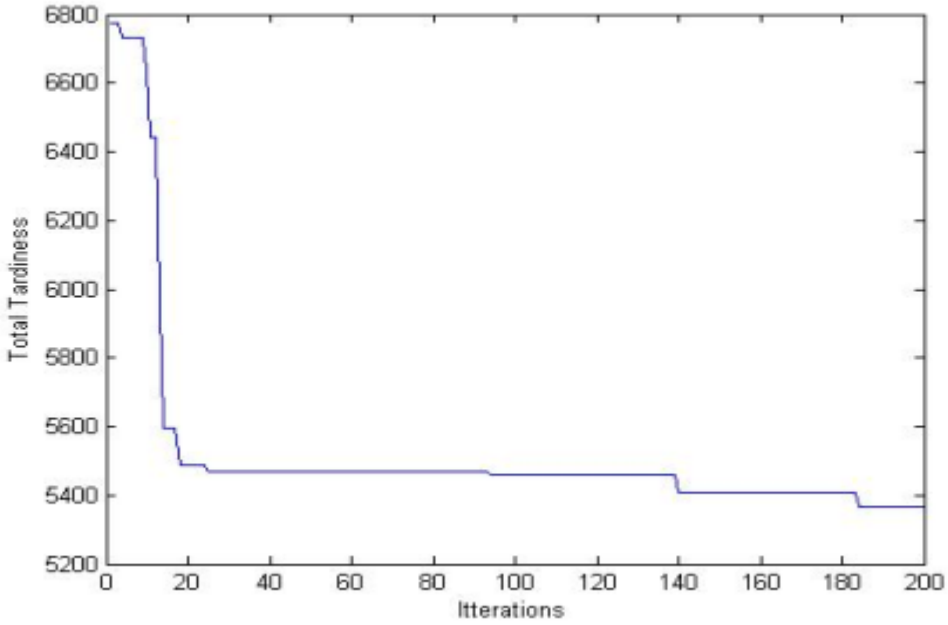
(β)

La06

Για το La06 πρόβλημα υπολογίζει βέλτιστη τιμή T_d =5366 σε 400 επαναλήψεις αισθητά χαμηλότερη από τον G&T αλγόριθμο.(T_d =6369). Η ταχύτητα σύγκλισης και το διάγραμμα Gantt του χρονοπρογράμματος δίδονται παρακάτω:



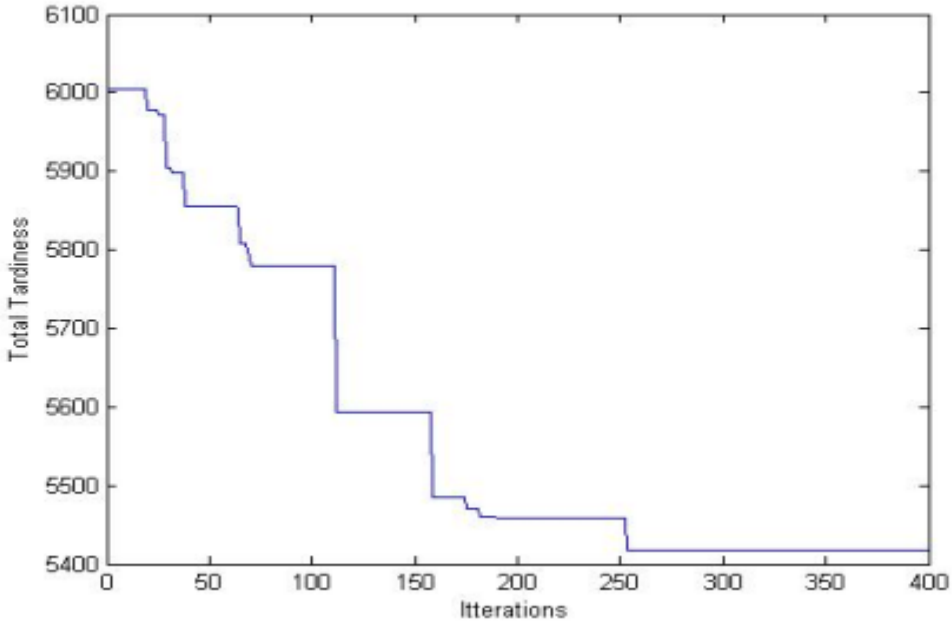
(α)



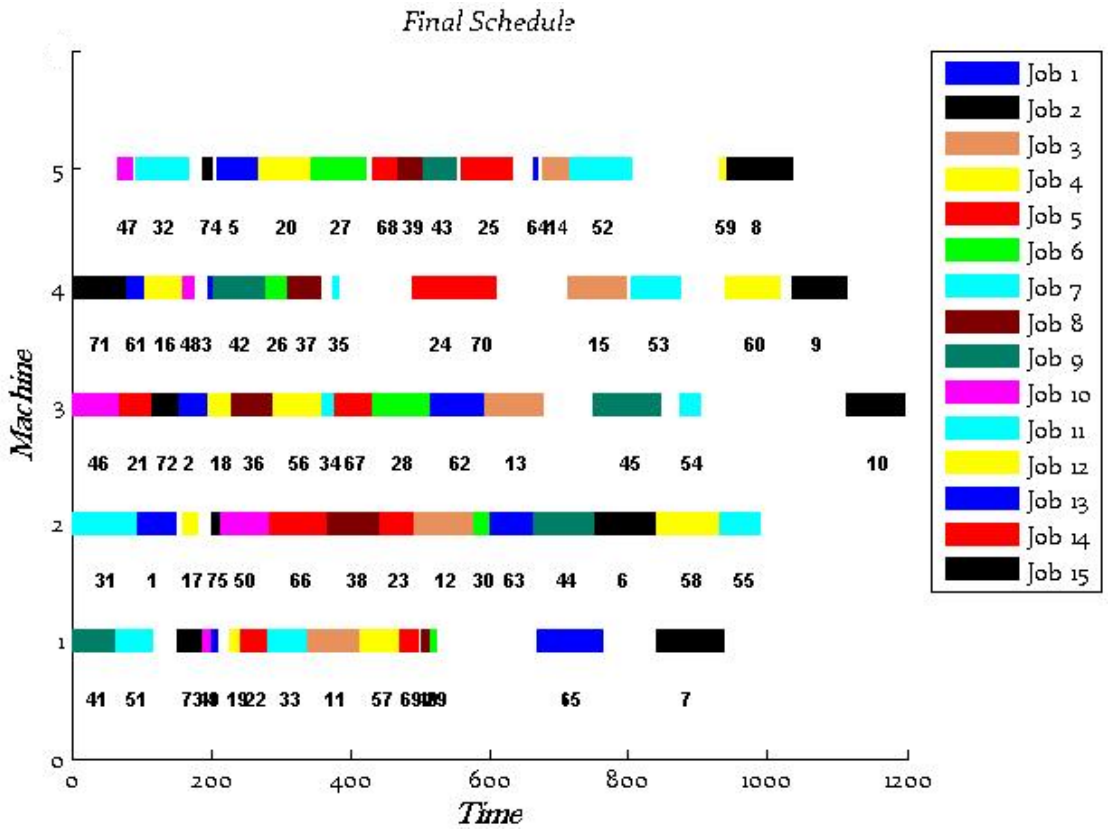
(β)

La10

Για το La10 πρόβλημα το σύστημα μας υπολογίζει βέλτιστη τιμή T_d =5417 σε 400 επαναλήψεις αρκετά βελτιωμενη σε σχέση με τον ευρετικό $G\&T$ αλγόριθμο.(T_d =6394). Η ταχύτητα σύγκλισης και το διάγραμμα Gantt του βέλτιστου προγράμματος δίδονται παρακάτω:



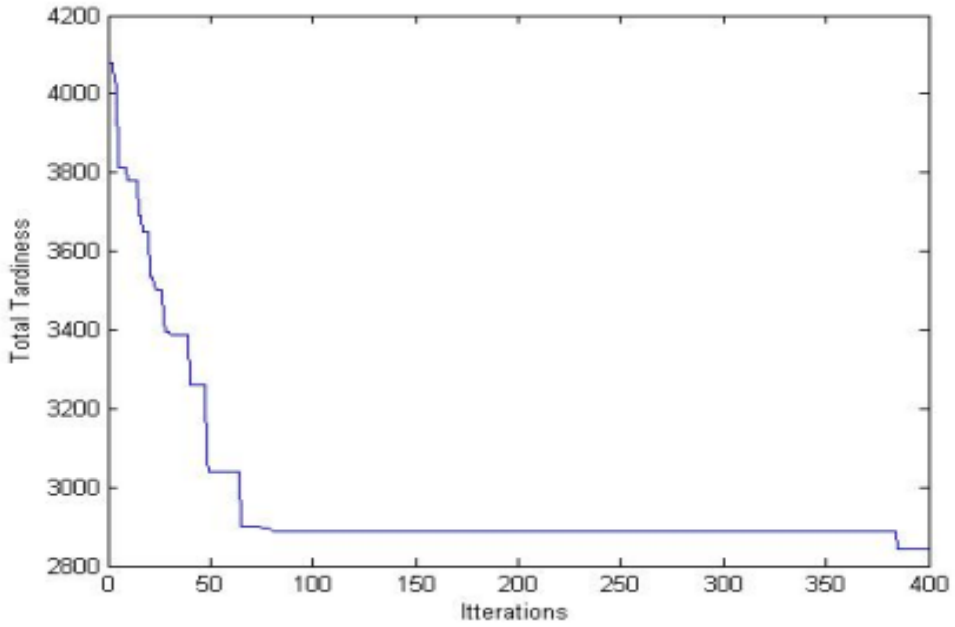
(α)



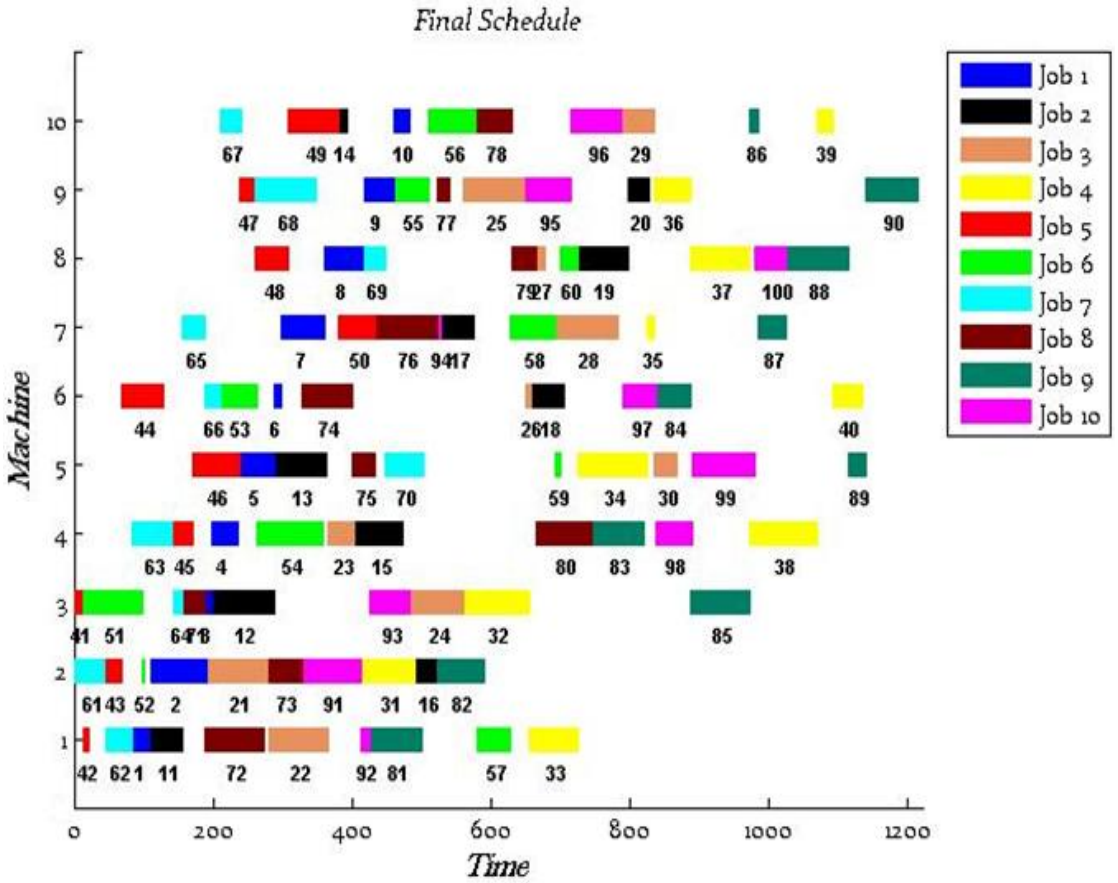
(β)

Ft10

Για το τετραγωνικό πρόβλημα Ft10 το σύστημα υπολογίζει βέλτιστη τιμή T_d =2842 σε 400 επαναλήψεις και είναι το μοναδικό πρόβλημα που το σύστημα δεν κατάφερε να βελτιωσει αρκετά την T_d σε σχέση με τον G&T αλγόριθμο.(T_d =3338). Η ταχύτητα σύγκλισης και το διάγραμμα Gantt του βέλτιστου προγράμματος δίδονται παρακάτω:



(α)

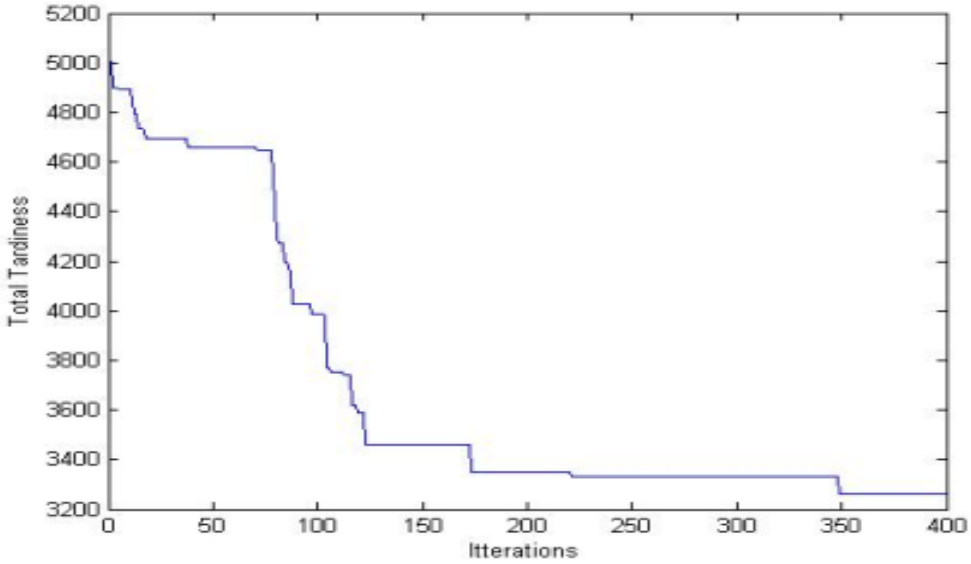


(β)

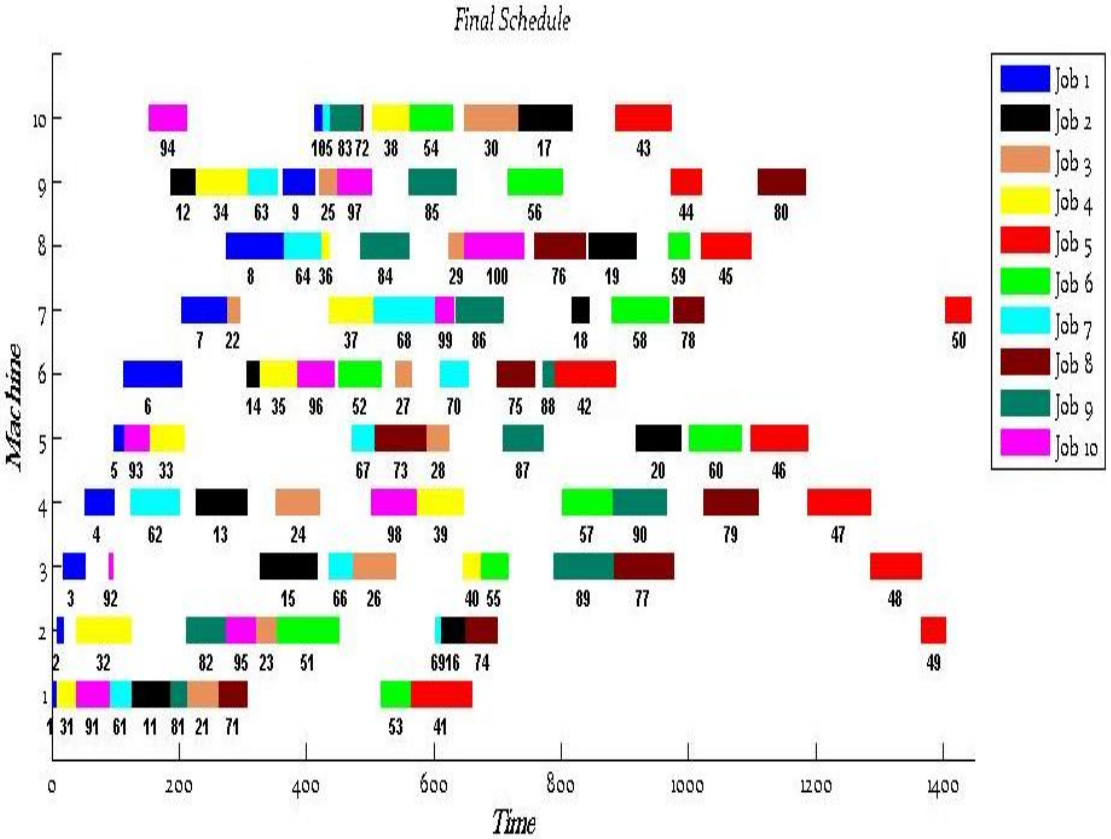
Orb04

Τέλος για το Orb04 παρατηρήθηκε η πιο θεαματική βελτιωση(T_d =3259) σε σχέση με

το αποτέλεσμα του G&T αλγόριθμου(T_d =4424),και ταυτόχρονα η μεγαλύτερη απόκκλιση μεταξύ των Average και ACS_Best τιμών. Η ταχύτητα σύγκλισης και το διάγραμμα Gantt του βέλτιστου προγράμματος φαίνονται παρακάτω:



(α)



(β)

Παρατηρήσεις-Συγκρίσεις αποτελεσμάτων

Στην βιβλιογραφία υπάρχουν ελάχιστες μελέτες που ασχολούνται με την ελαχιστοποίηση της T_d ,κατεπέκταση δεν έχουν υπολογιστεί συγκεκριμένες βέλτιστες θεωρητικές τιμές για τα στιγμιότυπα ,ώστε να συγκρίνουμε τα αποτελέσματα μας.Ύστερα απο εφαρμογή του αλγόριθμου των Giffler and Thompson(GT algorithm) ,ενός απλού αλγόριθμου για την παραγωγή ενεργών χρονοπρογραμμάτων ,με χρήση του EDD κανόνα(βλ ενότητα 5.2), συγκρίναμε τα αποτελέσματα του αλγόριθμου με αυτά του σύστήματος μας αναμένοντας αισθητή βελτιωσή τους ,καθώς το σύστημά μας χρησιμοποιεί πιο εξειδικευμένες τεχνικές εξερεύνησης του χώρου λύσεων,σε αντίθεση του στερούμενου διορατικότητας G&T αλγορίθμου που απλά προσθέτει διεργασίες στο πρόγραμμα βάση ενος άπληστου κριτηρίου(EDD rule).Οι προσδοκίες μας επαληθεύτηκαν καθώς το σύστημα βελτιώνει δραματικά την τιμή της Td σε όλα τα στιγμιότυπα με εξαίρεση το Ft10.

Γενικώς παρατηρήθηκαν σημαντικές αποκκλίσεις των αποτελεσμάτων μεταξύ των διαφόρων εκτελέσεων του σύστήματος σε όλα σχεδόν τα προβλήματα ,χωρίς να καταφέρνει να συγκλίνει στο ίδιο τοπικό βέλτιστο για παραπάνω από δύο εκτελέσεις(runs).Το γεγονός αυτό φαίνεται από την διαφορά των Average,ACS_Best τιμών, μπορεί να οφείλεται στους αυστηρούς χρόνους παράδοσης των εργασιών και σε μερική ανεπάρκεια της ευρετικής πληροφορίας.Ο MDD κανόνας θεωρείται ιδιαίτερα αποδοτικός σε προβλήματα που περιλαμβάνουν χρόνους παράδοσης,ωστόσο καθώς αυξανονται οι διαστάσεις και η πολυπλοκότητα του προβλήματος προσεκτική εφαρμογή δύο ή και περρισοτέρων κανόνων ανάθεσης στο ίδιο πρόβλημα μπορεί να δώσει καλύτερα αποτελέσματα (Joc Cing Tay & Nhu Binh Ho).

6.2 Εφαρμογές στο πολυκριτηριακό σύστημα

Στη 2η φάση των πειραμάτων εφαρμόσαμε το πολυκριτηριακό σύστημα που μοντελοποιήσαμε ,αναλύσαμε και σχεδιάσαμε στις Ενότητες 4.4 ,5.3.Για να εκτιμήσουμε την απόδοση του ως προς την ποιότητα των λύσεων ,δοκιμάσαμε πάλι το προτεινόμενο σύστημα στα εφτά κλασσικά προβλήματα των Fisher & Thompson() και Lawrence και ένα των Applegate & Cook(Orb04) που παρουσιάστηκαν στη Ενότητα 3.4.Το σύστημα εκτελέστηκε δέκα φορές σε κάθε πρόβλημα.Ως γνωστόν τα Ft10,La21 και Orb04 αποτελούν δύσκολες υπολογιστικές προκλήσεις για αρκετό καιρό.Στον Πίνακα 6.3 φαίνονται τα στατιτστικά της απόδοσης του συστήματος στα πλαίσια του makespan(C_{max}) και της συνολικής βραδύτερης περάτωσης(T_d). Ο Πίνακας περιλαμβάνει τις καλύτερες τιμές για κάθε αντικειμενική στο βέλτιστο Pareto μέτωπο, καθώς και την αξιολόγησή του βάση τριών ποιοτικών μέτρων:

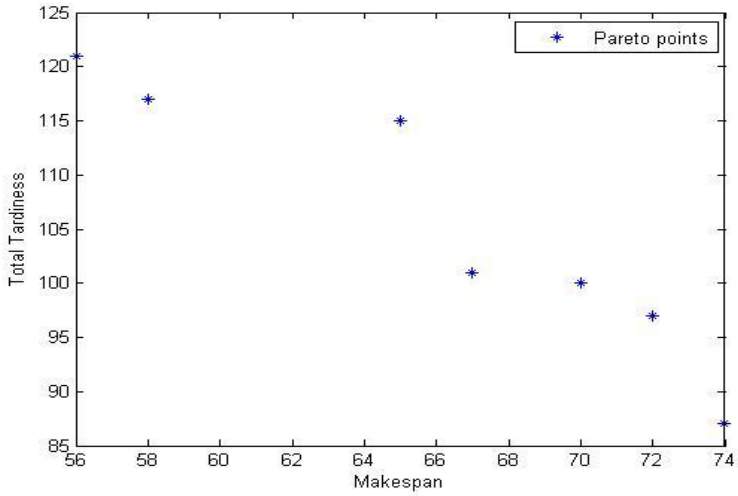
1. ***ONVG**: Το πλήθος των μη-κυριαρχούμενων λύσεων στο P.*
2. ***Spacing metric**:Το μέτρο αυτο ορίστηκε στη Ev. 5.2.2 , Εξ. 5.15 και υπολογίζει την μέση απόκκλιση των αποστάσεων ανάμεσα στις λύσεις του P.*
3. ***Diversification Metric(DM),** Mirzaei et al(2010):Υπολογίζει το εύρος του συνόλου λύσεων στο P, και ορίζεται ως ,όπου*

,η ευκλείδια απόσταση ανάμεσα στις μη-κυριαρχούμενες λύσεις και .Μεγαλύτερες τιμές του μέτρου δείχνουν πιό «spread» σύνολο.

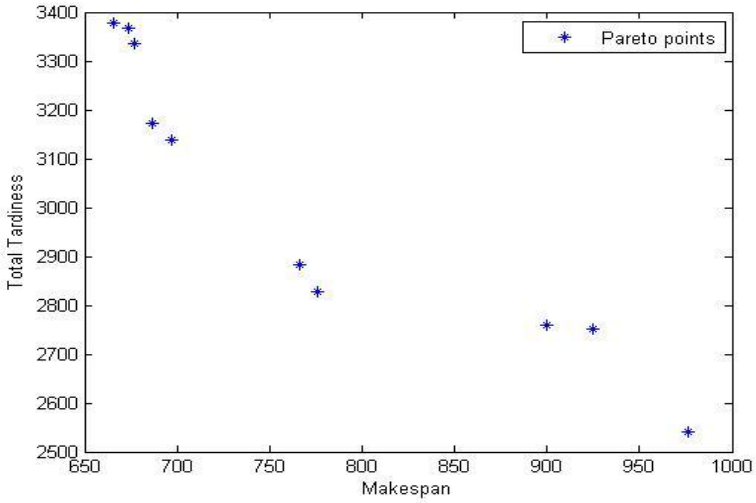
Τα χρονοπρογράμματα των ακραίων μη-κυριαρχούμενων λύσεων του Πίνακα 6.3 παρατίθονται στο Παράρτημα Β.

Οι παράμετροι τέθηκαν ίσες σε όλα τα προβλήματα,δηλαδή β=1,q0=0.7,και ρ=0.1 και αριθμός των μυρμηγκιών τέθηκε ίσος με m= 40 .

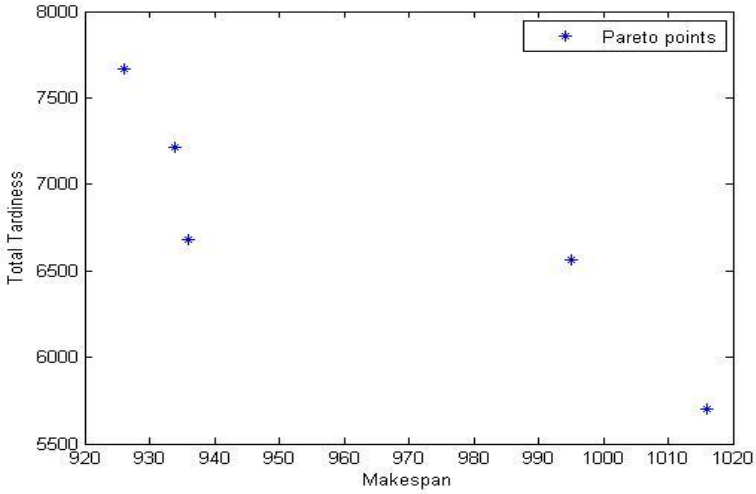
Για να δείξουμε την σύγκλιση και διαφοροποίηση των λύσεων ,οι μη κυριαρχούμενες λύσεις του τελικού Pareto συνόλου σε κάθε πρόβλημα παρατίθονται στις Εικόνες 6.4,α) εώς 6.4,στ).Οι τελικές λύσεις είναι καλά κατανεμημένες και προσεγγίζουνε σε ικανοποιητικό βαθμό τις βέλτιστες τιμές για κάθε κριτήριο που υπολογίσαμε στα πλαίσια του single-objective συστήματος.Συγκεκριμένα για τα πιό μικρά στιγμιότυπα ft06 la01,la06,la10 το σύστημα συγκλίνει σε ένα ικανοποιητικό αριθμό μη – κυριαρχούμενων λύσεων,όπως φαινεται στις Εικόνες 6.5a) εως 6.5στ) που συχνά αποτελούν και τα κατώτερα όρια για τα δύο κριτήρια.Άρα το σύστημα είναι ικανό στον να βρίσκει ακραίες λύσεις ,κάτι που δικαιολογεί και η τιμή του Space(S) μέτρου για κάθε πρόβλημα(Πίνακας 6.3).Μικρότερες τιμές για το S σημαίνει μεγαλύτερη διαφοροποίηση των λύσεων στο Pareto μέτωπο.



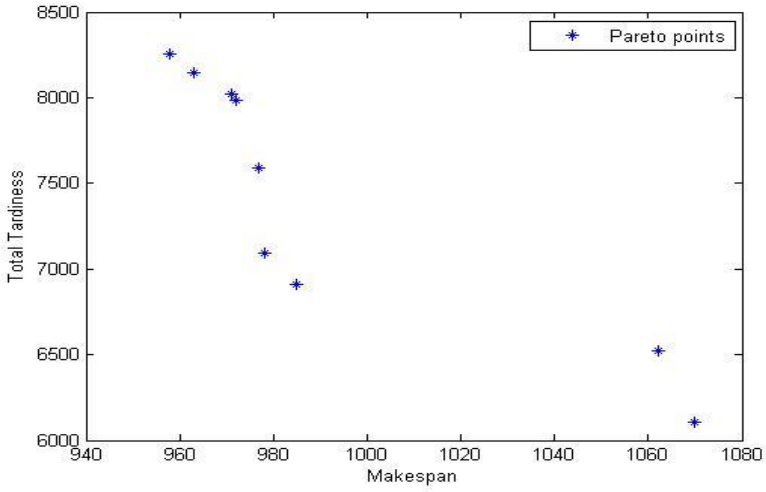
(α) *Τελικό Pareto αρχείο* P *για το* ft06 *στιγμιότυπο*



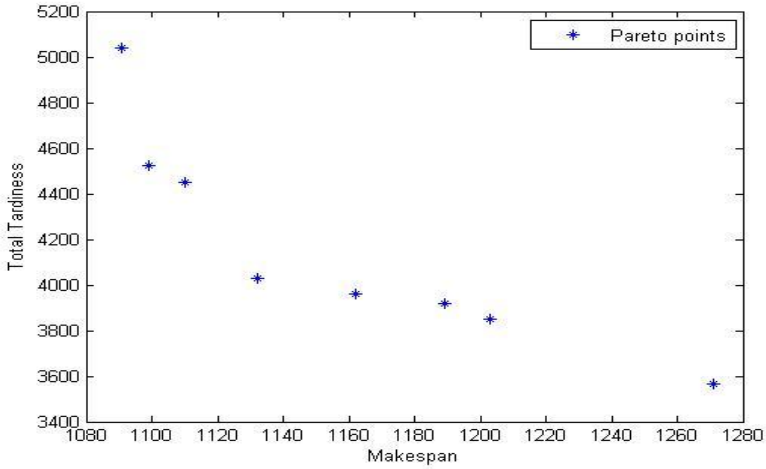
(β) *Τελικό Pareto αρχείο* P *για το* la01 *στιγμιότυπο* .



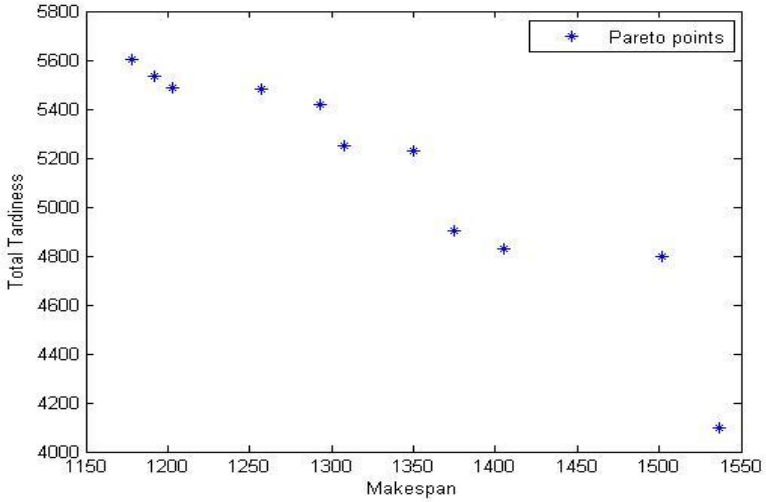
(γ) *Τελικό Pareto αρχείο* P *για το* la06 *στιγμιότυπο* .



(δ) *Τελικό Pareto αρχείο* P *για το* la10 *στιγμιότυπο* .



(ε) *Τελικό Pareto αρχείο* P *για το* ft10 *στιγμιότυπο* .

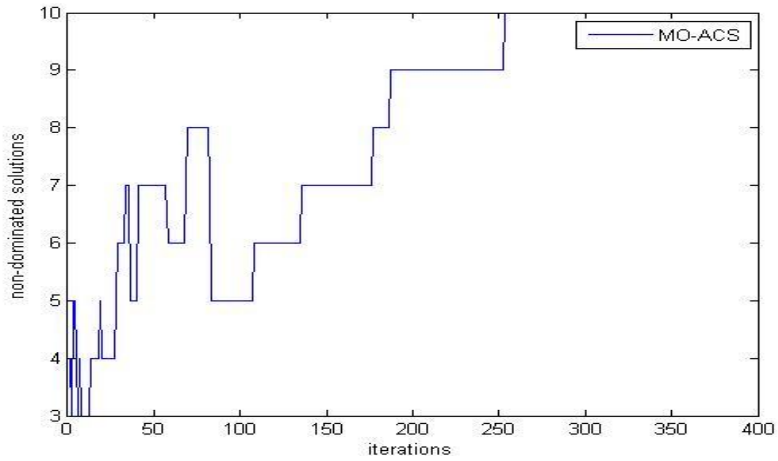


(στ) *Τελικό Pareto αρχείο* P *για το* orb04 *στιγμιότυπο* .

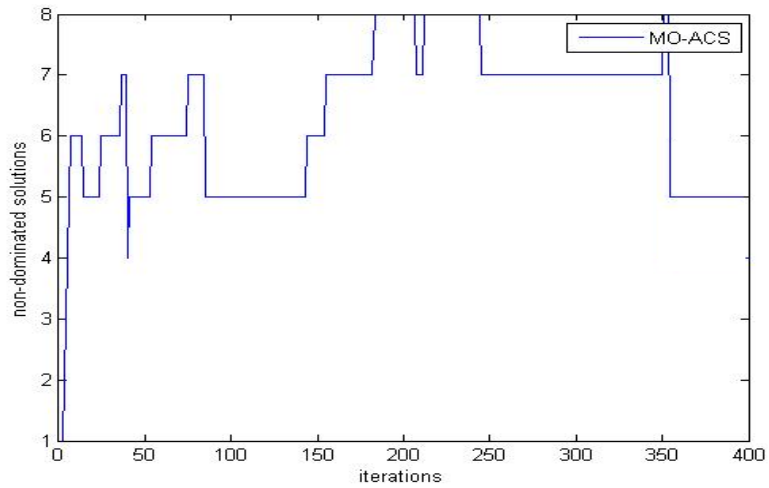
Εικόνα 6.4 :Τελικό Pareto-βέλτιστο μέτωπο P για 7 στιγμιότυπα του JSP.

JSP name	***Best makespan***	***Best tardiness***	***Diversity(DM)***	***Spacing(S)***	***ONVG***
Ft06	*(**56**,121)*	*(74,**87**)*	*28.2489*	*0.3986*	*7*
La01	*(**666**,3377)*	*(976,**2541**)*	*115.4123*	*0.7590*	*10*
La06	*(**926**,7664)*	*(1016,**5701**)*	*98.2955*	*0.5820*	*5*
La10	*(**958**,8255)*	*(1070,**6109**)*	*135.3144*	*0.7011*	*9*
Ft10	*(**1091**,5038)*	*(1271,**3564**)*	*132.1212*	*0.8878*	*8*
La21					
Orb04	*(**1178**,5601)*	*(1537,**4097**)*	*160.9845*	*0.8426*	*11*

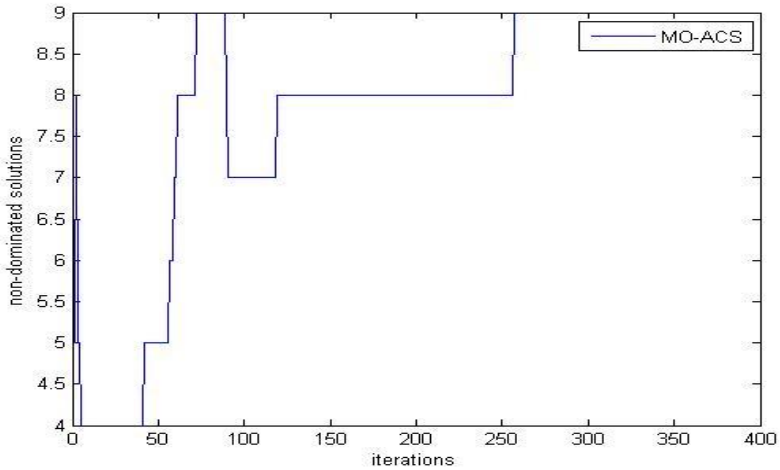
Πίνακας 6.3:Αξιολόγηση του τελικού Pareto-βέλτιστου μετώπού για κάθε benchmark.



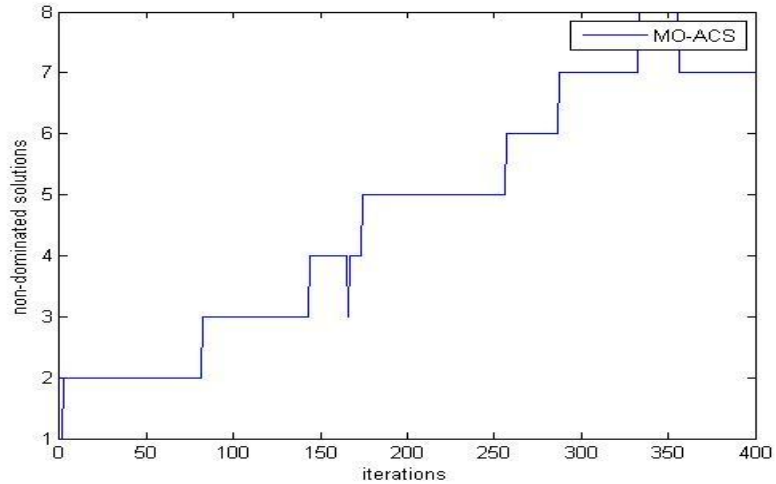
(α)la01



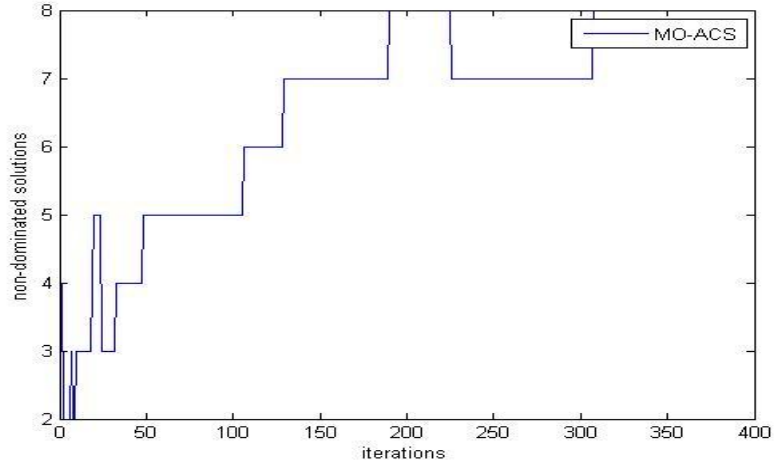
(β)la06



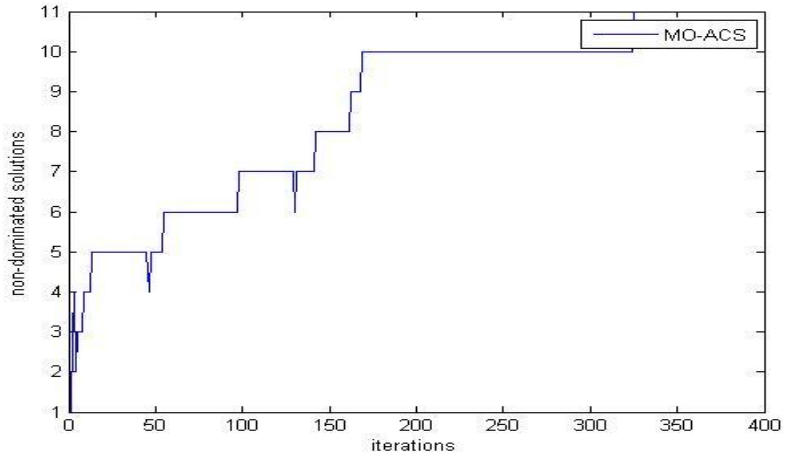
(γ)la10



(δ)ft06



(ε)ft10



(στ)orb04

Εικόνα 6.5 :Μη-κυριαρχούμενες λύσεις σε κάθε κύκλο.

Συνοψίζοντας το προτεινόμενο σύστημα είναι σε θέση να παράγει έναν ικανοποιητικό αριθμό απο consistent σχεδόν βέλτιστες λύσεις οι οποίες είναι καλά κατανεμημένες και συγκλίνουσες.Με βάση την σχετικά μέτρια απόδοση του single –objective συστήματος για κάθε κριτήριο παράγει λύσεις που προσεγγίζουν σε καλό βαθμό το βέλτιστο Pareto μέτωπο ,ενώ ειδικά τα αποτελέσματα για την ελαχιστοποίηση της T_d δείχνουν ότι αν και το σύστημα δεν καταφέρνει τις βέλτιστες τιμες της μονοκριτηριακής εκδοχής παρουσιάζει μεγαλύτερη σταθερότητα και μικρότερες αποκκλίσεις για την ελάχιστη τιμή της T_d σε κάθε εκτέλεσή του,ενώ για το makespan επιτυγχάνει τις βέλτιστες τιμές για τα εύκολα στιγμιότυπα (ft06,la01,l06,la10) και τις πλησιάζει σε ικανοποιητικό βαθμό για τα πιο πολύπλοκα προβλήματα (ft10,la21,orb04).Η καλή κατανομή και διασπορά των μη-κυριαρχούμενων λύσεων δικαιολογεί την εύρεση ακραίων τιμών όπως αναφέραμε παραπάνω και παρέχει στον λήπτη αποφάσεων μια ευρεία γκάμα επιλογών.

Ft06	*(58,117)*	*(**56,**121)*	*(65,115)*	*(70,100)*	*(72,97)*	*(74,**87**)*	*(67,101)*	

La01	*(697,3139)*	*(976,**2541**)*	*(674,3368)*	*(687,3173)*	*(776,2827)*	*(900,2758)*	*(677,3336)*	*(**666**,3377)*	*(925,2752)*	*(766*
La06	*(995,6561)*	*(936,6682)*	*(1016,**5701**)*	*(934,7216)*	*(**926**,7664)*					
La10	*(971,8024)*	*(985,6911)*	*(977,7590)*	*(1079,**6109**)*	*(1062.6526)*	*(963,8148)*	*(978,7091)*	*(972,7981)*	*(**958**,8255)*	
Ft10	*(**1091**,5038)*	*(1110,4451)*	*(1132,4029)*	*(1162,3963)*	*(1189,3917)*	*(1271,**3564**)*	*(1203,3850)*	*(1099,4256)*		
Ft20										
Orb04	*(**1178**,5601)*	*(1203,5487)*	*(1350,5230)*	*(1375,4902)*	*(1192,5532)*	*(1405,4827)*	*(1257,5483)*	*(1537,**4097**)*	*(1293,5421)*	*(1502*

Πίνακας 6.4:Επιλεγμένα μέτωπα Pareto για κάθε benchmark.

Στον Πίνακα 6.4 παραθέτουμε επιλεγμένα Pareto σύνολα για κάθε benchmark.Με bold φαίνονται τα Pareto σημεία που έχουν ακραίες τιμές για κάθε κριτήριο.

Κεφάλαιο 7 Συμπεράσματα- Μελλοντική Έρευνα

Συμπεράσματα

Ο βασικός στόχος της 1ης φάσης των περιραμάτων είναι η σχεδίαση και εφαρμογή ενός μονοκριτηριακού συστήματος χρονοπρογραμματισμού ευφυών πρακτόρων σε 7 δημοφιλή προβλήματα χρονοπρογραμματισμού παραγωγής της διεθνούς βιβλιογραφίας.Στο προτεινόμενο σύστημα , ένα σύνολο συνεργαζόμενων πρακτόρων (μυρμήγκια) συνεργάζονται για την εύρεση καλών λύσεων για το πρόβλημα χρονοπρογραμματισμού παραγωγής κατα παραγγελία(JSP).Οι πράκτορες συνεργάζονται χρησιμοποιώντας μια έμμεση μορφή επικοινωνίας η οποία κατευθύνεται από την φερομόνη που εναποθέτουν στις ακμές του κατασκευαστικού γράφου ,καθώς κατασκευάζουν τις λύσεις. Για δεδομένα benchmarks ,επιλέξαμε ως αντικειμενικά κριτήρια τον χρόνο ολοκλήρωσης όλων των εργασιών(C_{max}) και τον συνολικό χρόνο βραδύτερης περάτωσης(T_d).,επιδιώκοντας την ελαχιστοποίηση τους με χρήση του μονοκριτηριακού συστήματος.Υιοθετήσαμε τους κανόνες ανανέωσης φερομόνης και μετάβασης του αλγόριθμου ACS(Ant Colony System).Το σύστημα υλοποιήθηκε με την βοήθεια των μαθηματικών εργαλείων της MATLAB.

Αναλύωντας τα πειραματικά αποτελέσματα συμπεραίνουμε ότι το μονοκριτηριακό σύτημα είναι ικανό να παράγει προγράμματα που προσεγγίζουν σε καλό βαθμό τις βέλτιστες τιμές για κάθε κριτήριο και με μικρό υπολογιστικό κόστος, λαμβάνοντας υπόψη την πολυπλοκότητα του κάθε προβλήματος.Η απόδοση του στα πλαίσια της ελαχιστοποίησης του makespan παρουσιάζει μεγαλύτερα σφάλματα καθώς αυξάνεται ο αριθμός των μηχανών και των εργασιών ,τα οποία ωστόσο είναι αποδεκτά συγκρίση του μικρού χρόνου εκτέλεσης.Κατα αντιστοιχία για την ελαχιστοποίηση του T_d καταφέρνει να τον βελτιστοποιήσει σε ικανοποιητικά επίπεδα, ώστοσο οι βέλτιστες λύσεις των πρακτόρων σε κάθε εκτέλεση παρουσιάζουν μεγαλύτερη αστάθεια σε σύγκριση με το makespan.Συγκρίναμε τα αποτελέσματα του σύστήματος με αυτά ενός ευρετικού αλγορίθμου(G&T algorithm) ,όπου η προσθήκη των διεργασιών γίνεται ευρετικά σε κάθε βήμα και άρα εγγυάται μόνο τοπικά βέλτιστες κινήσεις,αναδεικνύωντας κατα συνέπεια την συνδρομή του μοντέλου φερομόνης στην εύρεση καλών λύσεων.

Είναι ενδεικτικό επίσης ,ανεξάρτητα του κριτηρίου που ελαχιστοποιούμε ,ότι το σύστημα συγκλίνει γρήγορα, ωστόσο η διαδικασίας αναζήτησης νέων λύσεων από τον πληθυσμό των πρακτόρων σε κάθε κύκλο παραμένει ενεργή, δεν μπαίνει δηλαδή σε κατάσταση κορεσμού(stagnation),όπου όλα οι πράκτορες ακολουθούν την ίδια διαδρομή στον γράφο.Αντίθετα συνεχίζουν την αναζήτηση νέων ,πιθανώς καλύτερων

λύσεων ακόμα και αν το σύστημα έχει συγκλίνει ήδη στη βέλτιστη τιμή.

Συνοψίζοντας σε αυτή την φάση των πειραμάτων επικεντρωθήκαμε περισσότερο στην μελέτη της απόδοσης του μονοκριτηριακού συστήματος στο JSP ,παρά στην διαμόρφωση ενός αλγόριθμου που θα συγκλίνει στις βέλτιστες λύσεις, μια δραστηριότητα που απαιτεί προσεκτική μελέτη της ευρετικής πληροφορίας καθώς και εφαρμογή μεθόδων τοπικής αναζήτησης και κατεπέκταση προσεκτικό ορισμό των συναρτήσεων γειτονιάς στα πλαίσια του JSP.

Στη 2η φάση των πειραμάτων εφαρμόσαμε έναν Pareto-based πολυκριτηριακό ACO σύστημα πρακτόρων για την λύση του JSP με δύο κριτήρια.Εξετάσαμε λεπτομερώς όλες τις πτυχές τις πολυκριτηριακής βελτιστοποίησης από την σκοπιά της οικογενείας των ACO μεθευρετικών αλγόριθμων και στην ενότητα 5.4 σχεδιάσαμε ένα σύστημα που να εκμεταλλεύεται όλα τα απαραίτητα στοιχεία μιας πολυκριτηριακής μεθόδου για την επίτευξη των δύο βασικών στόχων της ολικής βελτιστοποίησης: α) την εύρεση ενός σύνολου λύσεων που να προσεγγίζουν στον μέγιστο βαθμό το βέλτιστο Pareto-μέτωπο και β) την εύρεση ενός συνόλου καλά κατανεμημένων(diverse) λύσεων εντός του χώρου αναζήτησης.Ο στόχος μας λοιπόν ήταν να βρεθεί ένα σύνολο άριστων Pareto λύσεων από το οποίο να μπορούν να συλλεχθούν λύσεις του προβλήματος που θα επιλέξει ο αποφασίζων, με βάση την αντικειμενική συνάρτηση που χρειάζεται να ικανοποιηθεί κάθε φορά σε διαφορετικό βαθμό.Έτσι γίνεται ανταλλαγή (trade-off) ποσοτήτων μεταξύ των αντικειμενικών συναρτήσεων εντός του συνόλου Pareto.Τα αποτελέσματα είναι ενθαρρυντικά,καθώς οι στόχοι που θέσαμε επιτεύχθηκαν σε μεγάλο βαθμό,δίνοντας περιθώρια για περαιτέρω ανάπτυξη και βελτίωση του συστήματος.Συνοψίζοντας οι ιδέες που παρουσιάστηκαν στον συγκεκριμένο πολυκριτηριακό σύστημα μπορούν να επεκταθούν εύκολα και σε διαφορετικές αντικειμενικές συναρτήσεις για το JSP ή ακόμη και σε διαφορετικά πολυκριτηριακά προβλήματα ,ακολουθώντας το ίδιο μοτίβο ,τη χρήση δηλαδή διαφορετικών ελκυστικοτήτων για καθε αντικειμενική, αποθηκεύοντας την πληροφορία για τις καλές λύσεις σε ένα μοναδικό πίνακα φερομόνης.Κατά αυτόν τον τρόπο ένα πολυκριτηριακό σύστημα χρονοπρογραμματισμού ευφυών πρακτόρων συνδυάζει φυσικά τις καλύτερες λύσεις που έχουν βρεθεί εώς τώρα,προσπαθώντας να ανακαλύψει νέες Pareto-βέλτιστες λύσεις.

Μελλοντική Έρευνα

Ολοκληρώνουμε αυτή την διπλωματική προτείνοντας κάποιες πιθανές κατευθύνσεις για μελλοντική έρευνα.Γενικά μπορούμε να τις ταξινομήσουμε σε δύο κατηγορίες: περαιτέρω βελτίωση του μονοκριτηριακού συστήματος για την λύση του JSP και επέκταση του για την επίλυση άλλων προβλημάτων χρονοπρογραμματισμού με μηχανές.

1. *Βελτίωση του μονοκριτηριακού συστήματος με δοκιμή των υπάρχοντων μοντέλων φερομόνης(Merkle & Middendorf, 2000,2001) ή και ανάπτυξη νέων ,καθώς και εισαγωγή μεθόδων τοπικής αναζήτησης για την ενίσχυση της αναζήτησης προς καλύτερες λύσεις.Η ανάπτυξη αποδοτικών δομών γειτονικών συναρτήσεων καθώς και ο υβριδισμός του με άλλους ευρετικούς αλγορίθμους(GA,TS) για την αποφυγή του κορεσμού και της παγίδευσης του αλγόριθμου σε κάποιο τοπικό ελάχιστο (Eswaramurthy & Tamilarasi,2009), μπορεί να βελτιώσει αισθητά την ποιότητα των λύσεων.*
2. *Εφαρμογή των παραπάνω βελτιώσεων στο πολυκριτηριακό μοντέλο σε*

συνδυασμό με διάφορες παραλλαγές των αλγοριθμικών του συστατικών,πχ.χρήση πολλαπλών αποικιών μυρμηγκιών ,fixed βαρη για καθε κριτήριο ,offline-elite αποθήκευση των μη-κυριαρχούμενων λύσεων,κλπ.

3. *Εφαρμογή του σε άλλα είδη προβλημάτων χρονοπρογραμματισμου, όπου διαφορετικές αναπαραστάσεις του προβλήματος(μοντέλο φερομόνης) και καινούριες δομές γειτονικών συναρτήσεων μπορούν να κατασκευαστούν.*

Βιβλιογραφία

1. *M. Dorigo & K. Socha, An Introduction to Ant Colony Optimization. T. F. Gonzalez (Ed.), Approximation Algorithms and Metaheuristics, CRC Press, 2007.*

2. *FOO SIANG LYN,HYBRIDISED ANT COLONY OPTIMISATION FOR JOB SHOP PROBLEM , Msc THESIS Of ENGINEERING, DEPARTMENT OF INDUSTRIAL AND SYSTEMS ENGINEERING NATIONAL, UNIVERSITY OF SINGAPORE 2005.*

3. *Elisa Valentina Onet & Ecaterina Vladu, Nature inspired algorithms and artificial intelligence, Department of Computer Science, University of Oradea, Faculty of Electrical Engineering and Information Technology.*

4. *Christopher L. Elliott , Multiagent Systems Agents:Different Meanings to Different People,University of British Columbia ,Department of Mechanical Engineering*

5. *Βλαχαβάς-Κεφαλάς-Βασιλειάδης-Κόκκορας-Σακελλαρίου,ΤΕΧΝΗΤΗ ΝΟΗΜΟΣΥΝΗ , Γ΄ Έκδοση.*

6. *ΔΕΛΙΑΣ ΠΑΥΛΟΣ, Εργασία Στο Μάθημα: «ΦΥΣΙΚΗ & ΤΕΧΝΗΤΗ ΝΟΗΜΟΣΥΝΗ».*

7. *http://www.public.asu.edu/~gfaineko/pub/diploma_thesis.pdf .*

8. *http://artemis.cslab.ntua.gr/Dienst/UI/1.0/Display/artemis.ntua.ece/DT2005-0175 .*

9. *Bodea, Badea and Purnus (2010),"Complex project scheduling using multi-agent methods: a Case Study for Research Projects" ", Management & Marketing, No. 3*

10. *Günther Zäpfel, Roland Braune, Michael Bögl ,Metaheuristic Search Concepts: A Tutorial with Applications to Production and Logistics, Springer , 2010.*

11. *Fatos Xhafa & Ajith Abraham ,Metaheuristics for Scheduling in Industrial and Manufacturing Applications, Springer , 2008.*

12. *T. Weise, Global Optimization Algorithms– Theory and Application, 2nd ed. Thomas Weise 2009.*

13. *Nikolaos F. Matsatsinis & Pavlos Delias ,Implementing an Agent-based Decision Support System for Task Allocation: a Multi-criteria Approach, Decision Support Systems Laboratory, Technical University of Crete.*

14. *Fujita, S., Lesser, V.: Centralized Task Distribution in the Presence of*

Uncertainty and Time Deadlines, ICMAS-96, (1996)

15. *Djamila Quelhadj ,A MULTI-AGENT SYSTEM FOR THE INTERGATED DYNAMIC SCHEDULING OF STEEL PRODUCTION, Thesis submitted to The University of Nottingham for the degree of Doctor of Philosophy, School of Computer Science & Information Technology, August 2003.*

16. *Jose M Vidal ,Fundamentals of Multiagent Systems with NetLogo Examples March 1, 2010.*

17. *Anant Singh Jain-Sheik Meeran "A state-of-the-art review of job-shop scheduling techniques" ,1998, Department of Applied Physics ,Electronic and Mechanical Engineering University of Dundee, Scotland ,UK, DD1 4HN.*

18. *Metta, Haritha, "ADAPTIVE, MULTI-OBJECTIVE JOB SHOP SCHEDULING USING GENETIC ALGORITHMS" (2008).Masters Theses. Paper 518.*

19. *Pinedo, Michael L. 2005. Planning and Scheduling in Manufacturing and Services, Springer, New York.*

20. *Michael Pinedo, Scheduling Theory, Algorithms, and Systems,Prentice Hall, 2002.*

21. *Colorni, A., Dorigo, M., Maniezzo, V. and Trubian, M. 1993: Ant system for Job-Shop scheduling. Belgian Journal of Operations Research, Statistics and Computer Science 34,39-54.*

22. *ΔΕΛΙΑΣ ΠΑΥΛΟΣ, Εργασία Στο Μάθημα: «ΦΥΣΙΚΗ & ΤΕΧΝΗΤΗ ΝΟΗΜΟΣΥΝΗ».*

23. *Minimizing total tardiness on a single machine using ant colony optimization. Andreas Bauer, Bernd Bullnheimer, Richard F Hartl, Christine Strauss in Central European Journal (2000).*

24. *Jun, Zhang, Xiaomin, Hu, Tan, X., Zhong, J. H. and Huang, Q., Implementation of an Ant Colony Optimization technique for job shop scheduling problem, Transactions of the Institute of Measurement and Control, Vol. 8, No. 1, pp.93-108, 2006.*

25. *Rob J. M. Vaessens, Emile H. L. Aarts, Jan Karel Lenstra: Job Shop Scheduling by Local Search. INFORMS Journal on Computing 8(3): 302-317 (1996).*

26. *Anders Bjerg Pedersen,Analysis of Taboo Search in JSSP, May 29, 2009.*

27. *E. Nowicki & C. Smutnicki: A Fast Taboo Search Algorithm for the Job Shop Problem, in Management Science, Vol. 42, No. 6, p797-813 (1996).*

28. *S. Iredi, D. Merkle, and M. Middendorf. Bi-criterion optimization with multi colony ant algorithms. In E. Zitzler, K. Deb, L. Thiele, C. C. Coello, and D. Corne, editors, First International Conference on Evolutionary Multi-Criterion*

Optimization, (EMO'01), volume 1993 of Lecture Notes in Computer Science, pages 359–372. Springer Verlag, Berlin, Germany, 2001.

29. *Lopez-Ibanez,M. (2004). Multi-objective ant colony optimization, Master's thesis, Darmstadt University of Technology.*

30. *Baran, B. & Schaerer, M. (2003). A multiobjective ant colony system for vehicle routing problem with time windows, Proceedings of the 21st IASTED International Conference, pp. 97–102.*

31. *Jaqueline S. Angelo and Helio J.C. Barbosa (2011). Ant Colony Algorithms for Multiobjective Optimization, Ant Colony Optimization - Methods and Applications, Avi Ostfeld (Ed.), ISBN: 978-953-307-157-2, InTech, Available from:* http://www.intechopen.com/articles/show/title/ant-colony-algorithms-for-multiobjective-optimization

32. *http://itia.ntua.gr/getfile/201/8/documents/2004WRSOMulticriteriaOptSM.pdf*

33. *Grosan, C., Oltean, M., Dumitrescu, D. Performance Metrics for Multiobjective Optimization Evolutionary Algorithms, In Proceedings of Conference on Applied and Industrial Mathematics (CAIM), Oradea, 2003.*

34. *Veldhuizen, D.A.V.: Multiobjective Evolutionary Algorithms: Classifications,Analyses, and New Innovations, PhD Thesis, Department of Electrical and Computer Engineering, Air Force Institute of Technology, Wright-Patterson AFB, USA (1999).*

35. *J. Knowles and D. Corne. On Metrics for Comparing Nondominated Sets. In Proceedings of the 2002 Congress on Evolutionary Computation (CEC'2002), volume 1, pages 711--716.*

36. *R. Tavakkoli-Moghaddam, A. Rahimi-Vahed and A. Hossein Mirzaei, Solving A Multi-objective No-wait Flow Shop Problem By A Hybrid Multi-objective Immune Algorithm.*

37. *Sha, D.Y., Liu, C.-H.: Using data mining for due date assignment in a dynamic job shop environment. Int. Adv. Manuf. Technol. 25, 1164–1174 (2005).*

38. *Haipeng Zhang, Mitsuo Gen, Shigeru Fujimura "HYBRIDIZING ANT COLONY OPTIMIZATION WITH DISPATCHING HEURISTICS FOR JOB-SHOP SCHEDULING PROBLEM" Proceedings of the fifth Asia Pacific Industrial Engineering and Management Systems Conference 2004, pp.27.6.1-27.6.9(2004).*

39. *M. Dorigo & L.M. Gambardella, Ant Colony System: A Cooperative Learning Approach to the Traveling Salesman Problem. IEEE Transactions on Evolutionary Computation, 1(1):53-66,1997.*

40. *MAHANIM BINTI OMAR,MODIFIED MULTI-STEP CROSSOVER FUSION (MSXF) IN SOLVING SOME DETERMINISTIC JOBSHOP SCHEDULING*

PROBLEM (JSSP), UNIVERSITI SAINS MALAYSIA 2008.

41. *http://en.wikipedia.org/wiki/Fitness_proportionate_selection.*

42. *Merkle, D., & Middendorf, M. (2000). An ant algorithm with a new pheromone evaluation rule for total tardiness problems. In: Proceedings of the EvoWorkshops 2000, lecture notes in computer science 1803 (pp. 287–296). Berlin: Springer.*

43. *Angus, D. & Woodward, C. (2009). Multiple objective ant colony optimisation, Swarm Intelligence 3(1): 69–85.*

44. *Angus, D. (2007). Crowding population-based ant colony optimisation for themulti-objective travelling salesman problem.In 2007 IEEE symposium on computational intelligence in multi-criteria decision-making (MCDM 2007) (pp. 333–340). New York: IEEE Press.*

45. *Bierwirth, C., A Generalized Permutation Approach to Job Shop Scheduling with Genetic Algorithms, Department of Economics, University of Bremen, Germany.*

Παράρτημα Α

Κάθε στιγμιότυπο περιλαμβάνει μια γραμμή περιγραφής του,μια γραμμή με τον αριθμό των εργασιών και μηχανών και ακολούθως από το σημείο αυτό,κάθε γραμμή αντιστοιχεί σε μία εργασία και περιλαμβάνει ένα ζεύγος αριθμών (τον αριθμό της μηχανής και τον χρόνο επεξεργασίας) για κάθε διεργασία της εργασίας.Οι μηχανές αριθμούνται αρχίζοντας από το μηδέν.

- ft06 and ft10 are from
  H. Fisher, G.L. Thompson (1963),
  Probabilistic learning combinations of local job-shop scheduling rules,
  J.F. Muth, G.L. Thompson (eds.), Industrial Scheduling,
  Prentice Hall, Englewood Cliffs, New Jersey, 225-251.

```
++++++++++++++++++++++++++

instance ft06

++++++++++++++++++++++++++++
Fisher and Thompson 6x6 instance, alternate name (mt06)
6 6
2  1  0  3  1  6  3  7  5  3  4  6
1  8  2  5  4 10  5 10  0 10  3  4
2  5  3  4  5  8  0  9  1  1  4  7
1  5  0  5  2  5  3  3  4  8  5  9
2  9  1  3  4  5  5  4  0  3  3  1
1  3  3  3  5  9  0 10  4  4  2  1


++++++++++++++++++++++++++++

instance ft10

++++++++++++++++++++++++++++
Fisher and Thompson 10x10 instance, alternate name (mt10)
10 10
0 29 1 78 2  9 3 36 4 49 5 11 6 62 7 56 8 44 9 21
0 43 2 90 4 75 9 11 3 69 1 28 6 46 5 46 7 72 8 30
1 91 0 85 3 39 2 74 8 90 5 10 7 12 6 89 9 45 4 33
1 81 2 95 0 71 4 99 6  9 8 52 7 85 3 98 9 22 5 43
2 14 0  6 1 22 5 61 3 26 4 69 8 21 7 49 9 72 6 53
2 84 1  2 5 52 3 95 8 48 9 72 0 47 6 65 4  6 7 25
1 46 0 37 3 61 2 13 6 32 5 21 9 32 8 89 7 30 4 55
2 31 0 86 1 46 5 74 4 32 6 88 8 19 9 48 7 36 3 79
0 76 1 69 3 76 5 51 2 85 9 11 6 40 7 89 4 26 8 74
1 85 0 13 2 61 6  7 8 64 9 76 5 47 3 52 4 90 7 45
```

- la01-la06-la10-la21 are from
  S. Lawrence (1984),
  Resource constrained project scheduling: an experimental investigation of heuristic scheduling techniques (Supplement),

Graduate School of Industrial Administration, Carnegie-Mellon University, Pittsburgh, Pennsylvania.

```
++++++++++++++++++++++++++++

instance la01

++++++++++++++++++++++++++++
Lawrence 10x5 instance (Table 3, instance 1); also called (setf1) or
(F1)
10 5
1 21 0 53 4 95 3 55 2 34
0 21 3 52 4 16 2 26 1 71
3 39 4 98 1 42 2 31 0 12
1 77 0 55 4 79 2 66 3 77
0 83 3 34 2 64 1 19 4 37
1 54 2 43 4 79 0 92 3 62
3 69 4 77 1 87 2 87 0 93
2 38 0 60 1 41 3 24 4 83
3 17 1 49 4 25 0 44 2 98
4 77 3 79 2 43 1 75 0 96


++++++++++++++++++++++++++++

instance la06

++++++++++++++++++++++++++++
Lawrence 15x5 instance (Table 4, instance 1); also called (setg1) or
(G1)
15 5
1 21 2 34 4 95 0 53 3 55
3 52 4 16 1 71 2 26 0 21
2 31 0 12 1 42 3 39 4 98
3 77 1 77 4 79 0 55 2 66
4 37 3 34 2 64 1 19 0 83
2 43 1 54 0 92 3 62 4 79
0 93 3 69 1 87 4 77 2 87
0 60 1 41 2 38 4 83 3 24
2 98 3 17 4 25 0 44 1 49
0 96 4 77 3 79 1 75 2 43
4 28 2 35 0 95 3 76 1  7
0 61 4 10 2 95 1  9 3 35
4 59 3 16 1 91 2 59 0 46
4 43 1 52 0 28 2 27 3 50
0 87 1 45 2 39 4  9 3 41


++++++++++++++++++++++++++++

instance la10

++++++++++++++++++++++++++++
Lawrence 15x5 instance (Table 4, instance 5); also called (setg5) or
(G5)
15 5
1 58 2 44 3  5 0  9 4 58
1 89 0 97 4 96 3 77 2 84
0 77 1 87 2 81 4 39 3 85
```

```
3 57 1 21 2 31 0 15 4 73
2 48 0 40 1 49 3 70 4 71
3 34 4 82 2 80 0 10 1 22
1 91 4 75 0 55 2 17 3  7
2 62 3 47 1 72 4 35 0 11
0 64 3 75 4 50 1 90 2 94
2 67 4 20 3 15 0 12 1 71
0 52 4 93 3 68 2 29 1 57
2 70 0 58 1 93 4  7 3 77
3 27 2 82 1 63 4  6 0 95
1 87 2 56 4 36 0 26 3 48
3 76 2 36 0 36 4 15 1  8
```

++++++++++++++++++++++++++++

instance la21

++++++++++++++++++++++++++++

```
Lawrence 15x10 instance (Table 7, instance 1); also called (setb1)
or (B1)
15 10
2 34 3 55 5 95 9 16 4 21 6 71 0 53 8 52 1 21 7 26
3 39 2 31 0 12 1 42 9 79 8 77 6 77 5 98 4 55 7 66
1 19 0 83 3 34 4 92 6 54 9 79 8 62 5 37 2 64 7 43
4 60 2 87 8 24 5 77 3 69 7 38 1 87 6 41 9 83 0 93
8 79 9 77 2 98 4 96 3 17 0 44 7 43 6 75 1 49 5 25
8 35 7 95 6  9 9 10 2 35 1  7 5 28 4 61 0 95 3 76
4 28 5 59 3 16 9 43 0 46 8 50 6 52 7 27 2 59 1 91
5  9 4 20 2 39 6 54 1 45 7 71 0 87 3 41 9 43 8 14
1 28 5 33 0 78 3 26 2 37 7  8 8 66 6 89 9 42 4 33
2 94 5 84 6 78 9 81 1 74 3 27 8 69 0 69 7 45 4 96
1 31 4 24 0 20 2 17 9 25 8 81 5 76 3 87 7 32 6 18
5 28 9 97 0 58 4 45 6 76 3 99 2 23 1 72 8 90 7 86
5 27 9 48 8 27 7 62 4 98 6 67 3 48 0 42 1 46 2 17
1 12 8 50 0 80 2 50 9 80 3 19 5 28 6 63 4 94 7 98
4 61 3 55 6 37 5 14 2 50 8 79 1 41 9 72 7 18 0 75
```

- orb04 is from
  D. Applegate, W. Cook (1991),
  A computational study of the job-shop scheduling instance,
  ORSA Journal on Computing 3, 149-156.it was generated in Bonn
  in 1986).

++++++++++++++++++++++++++++

instance orb04

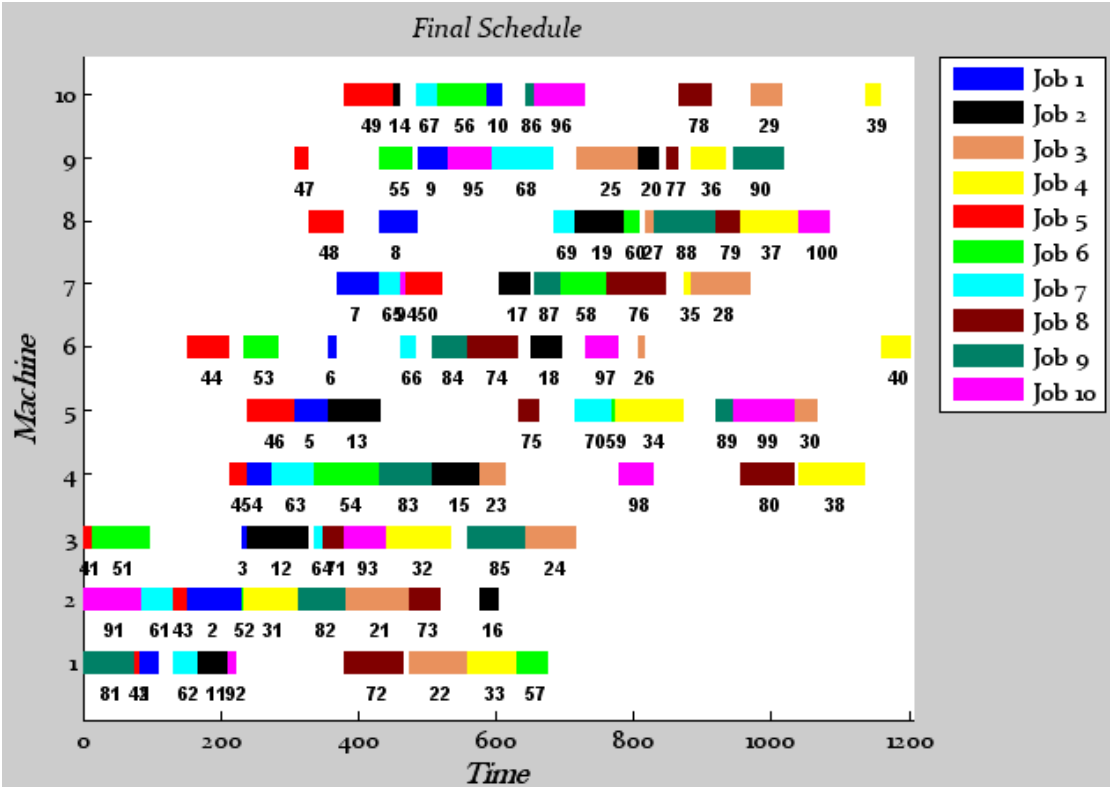
++++++++++++++++++++++++++++

```
deadly 10x10 instance from Bruce Shepherd (BRS1)
10 10
0  8 1 10 2 35 3 44 4 15 5 92 6 70 7 89 8 50 9 12
0 63 8 39 3 80 5 22 2 88 1 39 9 85 6 27 7 74 4 69
0 52 6 22 1 33 3 68 8 27 2 68 5 25 4 34 7 24 9 84
0 31 1 85 4 55 8 80 5 58 7 11 6 69 9 56 3 73 2 25
0 97 5 98 9 87 8 47 7 77 4 90 3 98 2 80 1 39 6 40
1 97 5 68 0 44 9 67 2 44 8 85 3 78 6 90 7 33 4 81
0 34 3 76 8 48 7 61 9 11 2 36 4 33 6 98 1  7 5 44
0 44 9  5 4 85 1 51 5 58 7 79 2 95 6 48 3 86 8 73
```

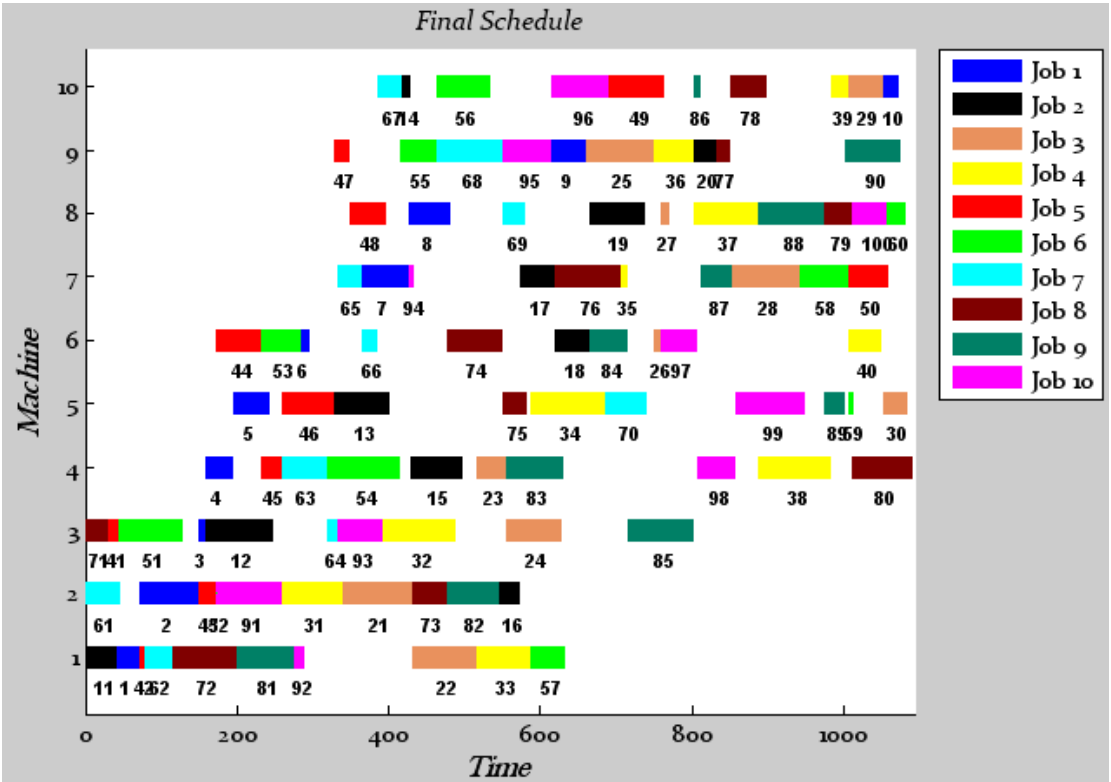
```
0 24 1 63 9 48 7 77 8 73 6 74 4 63 5 17 2 93 3 84
0 51 2  5 4 40 9 60 1 46 5 58 8 54 3 72 6 29 7 94
```

Παράρτημα Β

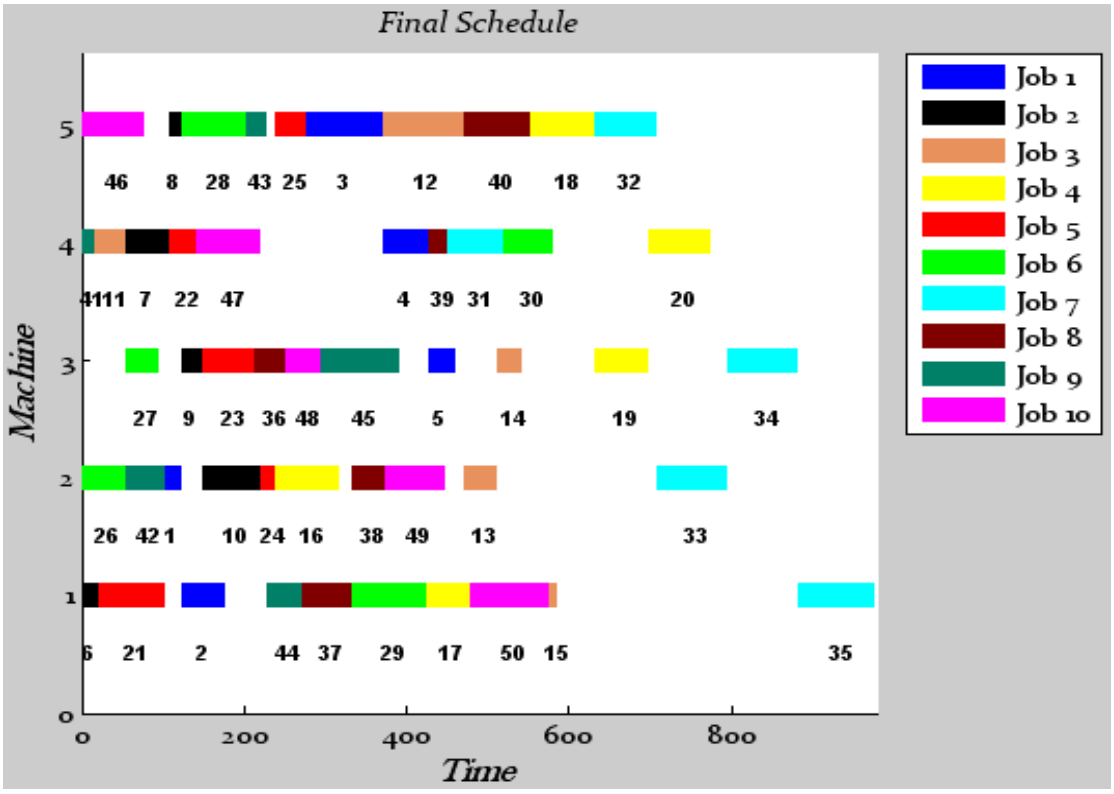
Ft10 :$C_{max=}$1251 ,$T_{d=}$3564



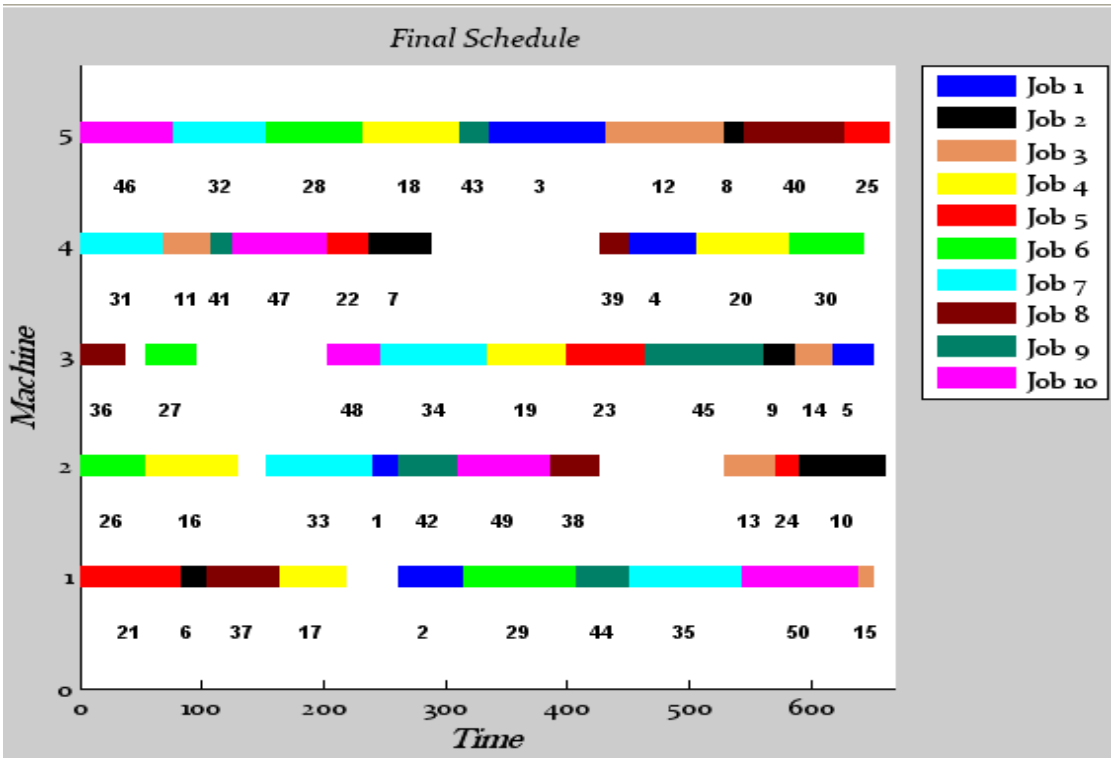
Ft10 :$C_{max=}$ 1091 ,$T_{d=}$5038



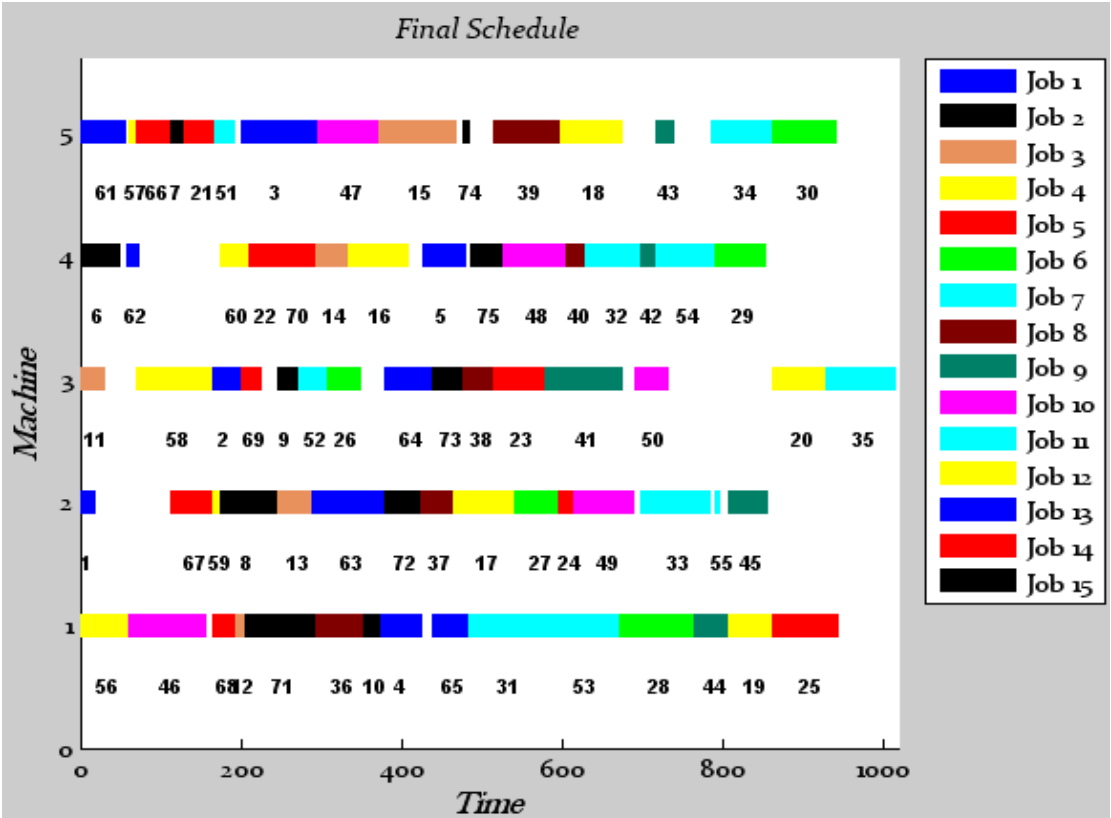
La01: $C_{max=}$ 976 ,$T_{d=}$2541



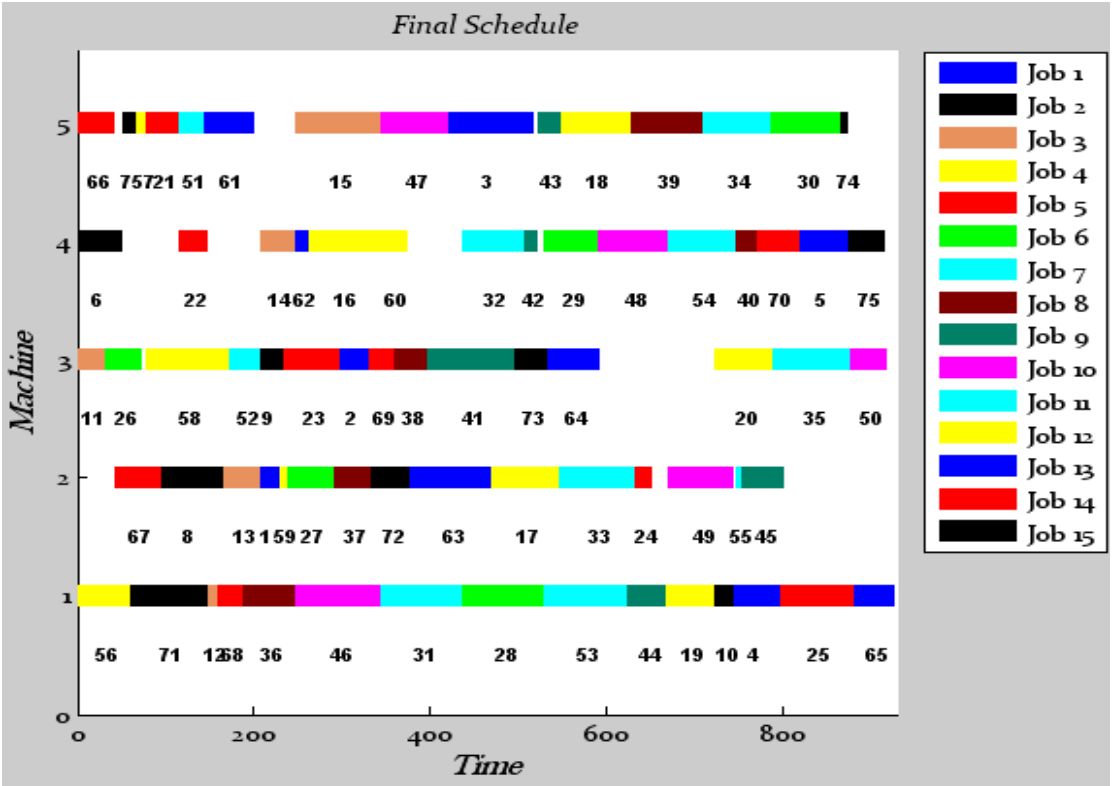
La01: $C_{max=}$666 ,$T_{d=}$3377



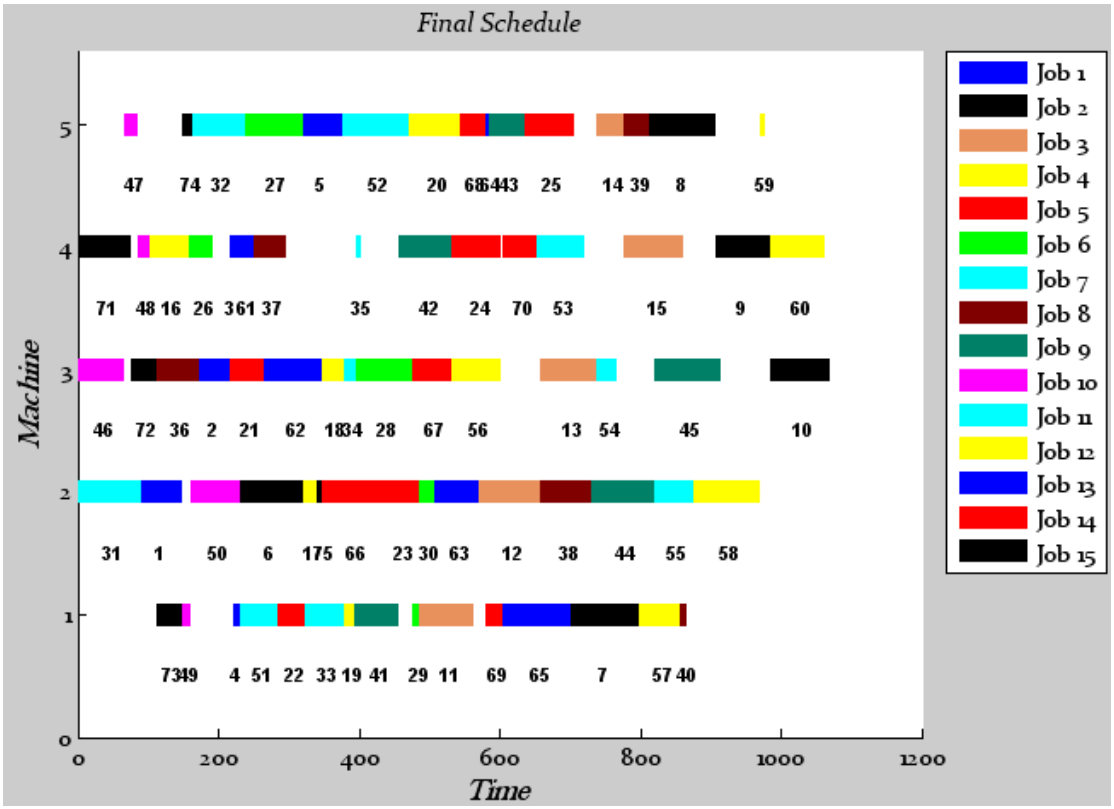
La06 :$C_{max=}$ 1016 ,$T_{d=}$5701



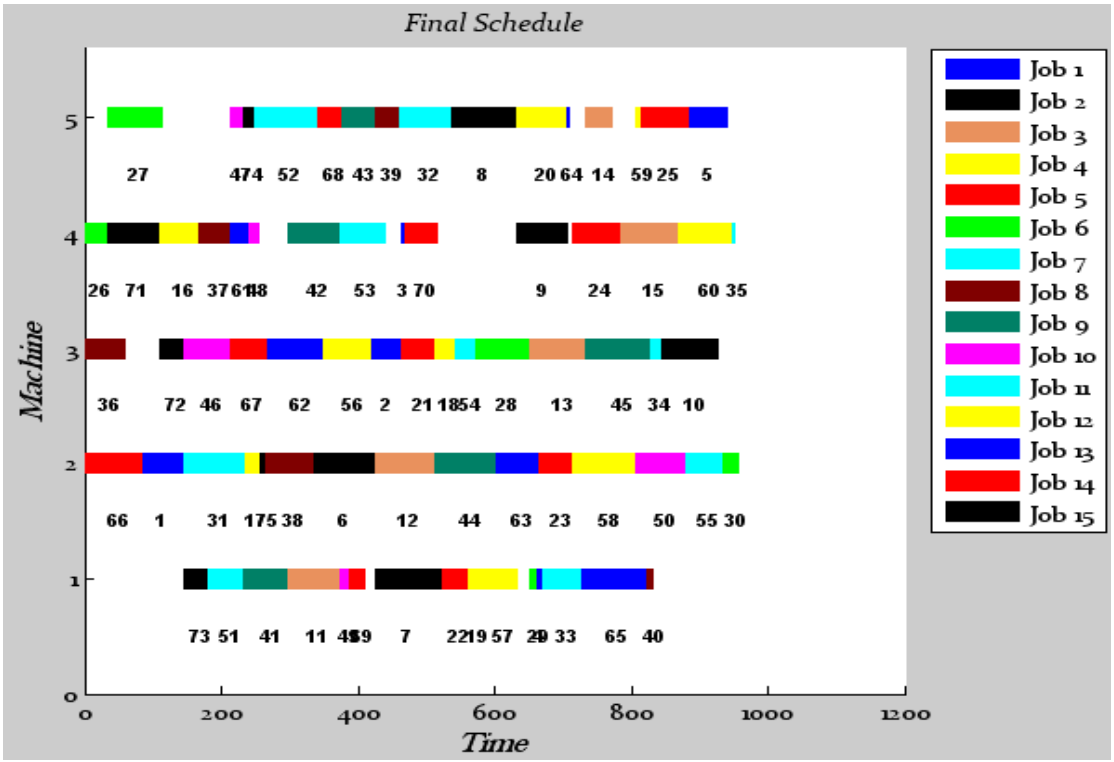
La06 :$C_{max=}$ 926 ,$T_{d=}$7664



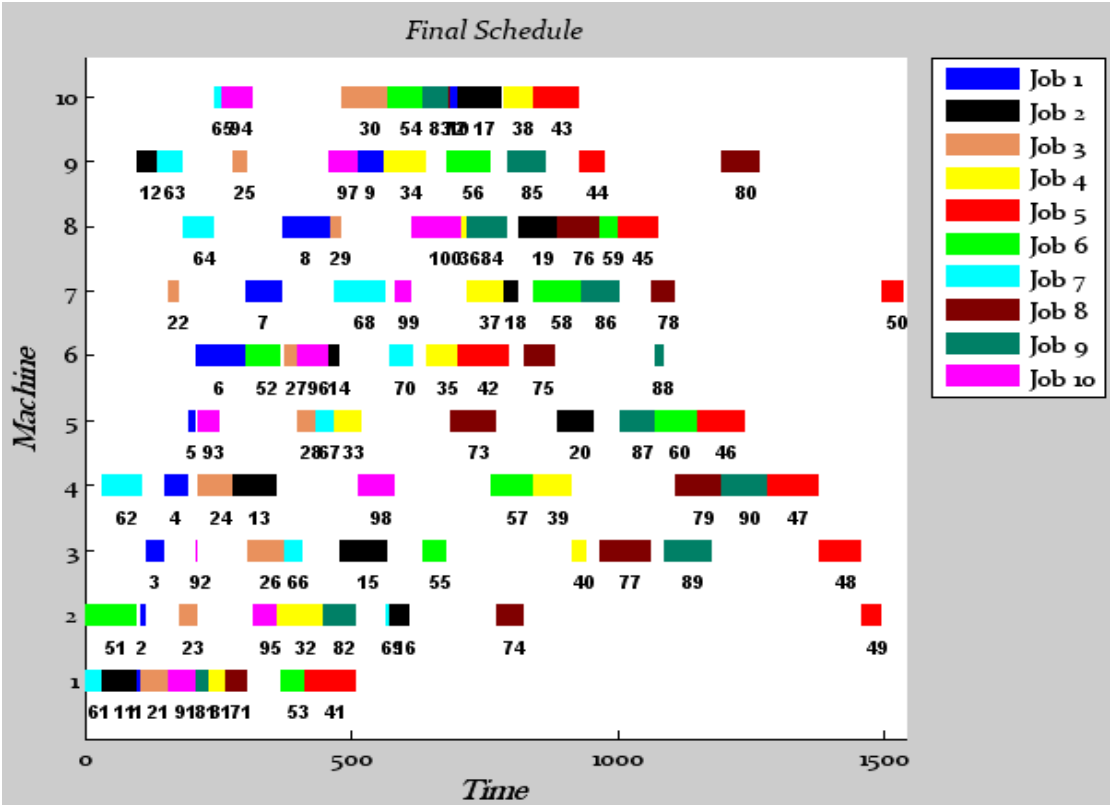
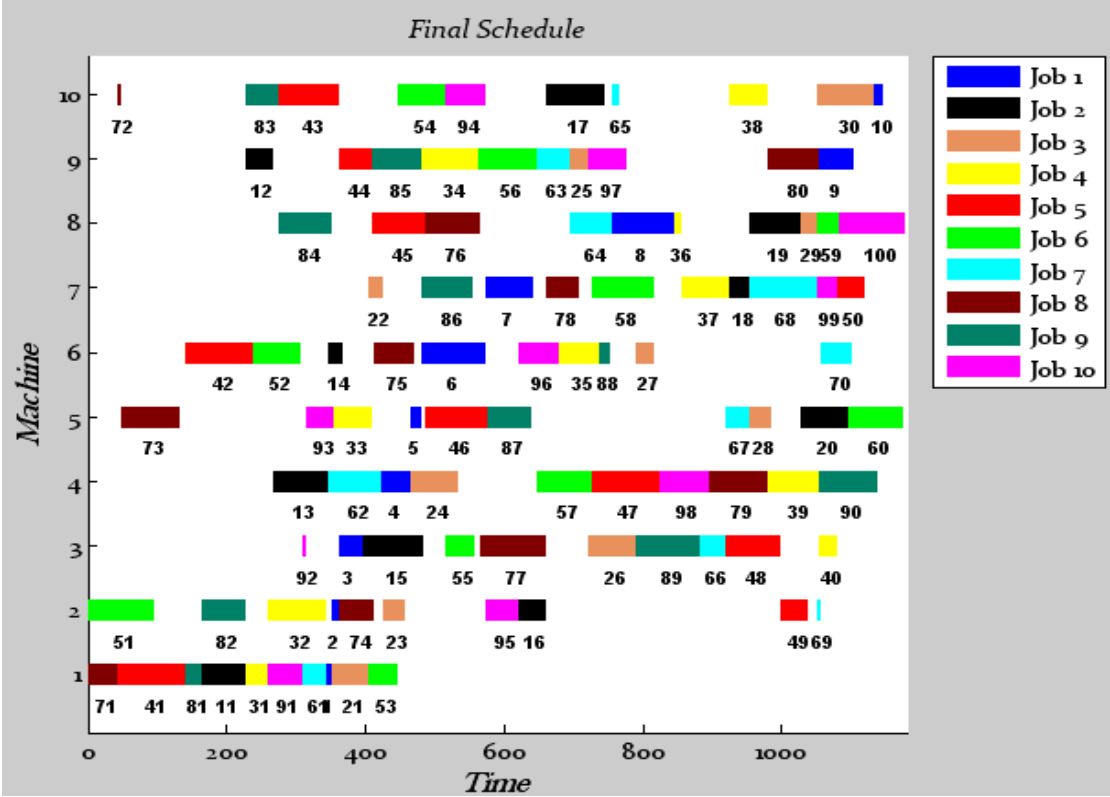
La10 :C_{max}=1070 ,T_d=6109



La10 :C_{max}= 958 ,T_d=8255



Orb04:$C_{max=}$ 1537,$T_{d=}$4097



Orb04:$C_{max=}$ 1178,$T_{d=}$5601