ΠΟΛΥΤΕΧΝΕΙΟ ΚΡΗΤΗΣ
ΤΜΗΜΑ ΜΗΧΑΝΙΚΩΝ ΠΑΡΑΓΩΓΗΣ ΚΑΙ ΔΙΟΙΚΗΣΗΣ
ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

ΣΥΓΚΡΙΤΙΚΗ ΜΕΛΕΤΗ ΜΕΘΟΔΩΝ ΑΚΕΡΑΙΟΥ ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΥ ΓΙΑ ΤΟΝ ΠΡΟΓΡΑΜΜΑΤΙΣΜΟ ΚΑΤΑΣΤΗΜΑΤΩΝ ΡΟΗΣ

ΕΠΙΜΕΛΕΙΑ: ΠΟΔΑΡΑ ΓΕΩΡΓΙΑ
ΕΠΙΒΛΕΠΩΝ ΚΑΘΗΓΗΤΗΣ : ΚΟΥΙΚΟΓΛΟΥ ΒΑΣΙΛΗΣ
ΣΕΠΤΕΜΒΡΙΟΣ 2005

Στους γονείς μου

ΠΕΡΙΕΧΟΜΕΝΑ

Ευχαριστίες

Είναι γεγονός πως πριν 5 χρόνια τέτοιο καιρό δεν σκεφτόμουν την στιγμή που θα γράφω αυτές τις γραμμές, αλλά τώρα είναι αναπόφευκτο να μην γυρίσω τον χρόνο πίσω. Με την ολοκλήρωση αυτής εδώ της διπλωματικής εργασίας κλείνει ένας κύκλος σπουδών 5 χρόνων γεμάτα με όνειρα, φιλοδοξίες, απόκτηση γνώσεων, ώρες μελέτης και αγωνίας κατά την διάρκεια των εξεταστικών περιόδων αλλά και πολλών όμορφων και ευχάριστων στιγμών ξενοιασιάς και ανεμελιάς. Και ενώ, ξεκινώντας την πορεία του στην πανεπιστημιακή κοινωνία κανείς φοιτητής δεν σκέφτεται το τέλος, στο τέλος αυτής της πορείας όλοι αναπολούν την αρχή.

Μετά από όλο αυτό το διάστημα, μπορώ να πω με σιγουριά πως η εκπόνηση της διπλωματικής εργασίας είναι από τα πιο ενδιαφέροντα κομμάτια στην διάρκεια των φοιτητικών χρόνων. Η ολοκλήρωση μιας τέτοιας εργασίας δίνει την δυνατότητα στον κάθε φοιτητή να επιλέξει το θέμα που του προκαλεί μεγαλύτερο ενδιαφέρον, να δουλέψει και να αποκτήσει περισσότερη γνώση πάνω σε αυτό. Ο φοιτητής έρχεται αντιμέτωπος με τον εαυτό του, δοκιμάζει τις δυνάμεις του, τις αντοχές του, τις γνώσεις του και τελικά με την παράδοση και ολοκλήρωση της, ανταμείβεται πάνω από όλα ηθικά.

Το αποτέλεσμα της πορείας μου σε αυτό το ταξίδι γνώσης, είναι η διπλωματική εργασία που κρατάτε αυτή την στιγμή στα χέρια σας. Ένα ταξίδι που ξεκίνησε στις 19 Φεβρουαρίου του 2005, όταν έφυγα από τα Χανιά για την Βαρκελώνη με το πρόγραμμα της Ευρωπαϊκής Ένωσης “Leonardo Da Vinci”, για να πραγματοποιήσω μια τρίμηνη πρακτική άσκηση στο ερευνητικό ινστιτούτο “Institute of Industrial Engineering and Control” του Πολυτεχνείου της Καταλονίας (Universidad Politechnica de Catalunya-UPC). Εκείνη την μέρα δεν μπορούσα να φανταστώ την κατάληξη της τρίμηνης παραμονής μου εκεί και ούτε φυσικά την γνώση και εμπειρία που θα αποκτούσα ως εργαζόμενη σε ένα περιβάλλον τόσο άγνωστο και διαφορετικό από αυτό που είχα συνηθίσει, όσο συνάμα και συναρπαστικό και ενδιαφέρον. Με την επιστροφή μου στα Χανιά, ήρθα σε επαφή με τον κ.Κουϊκόγλου Βασίλη, ώστε να μου δοθεί η δυνατότητα να παρουσιάσω την δουλειά που έκανα και τις γνώσεις που απέκτησα από αυτή μου την εμπειρία.

Με την ολοκλήρωση αυτής της διπλωματικής, οφείλω να ευχαριστήσω κάποιους ανθρώπους που συνέβαλαν και με βοήθησαν ουσιαστικά όλο αυτό το διάστημα, τόσο πρακτικά όσο και ηθικά.

Πρώτα από όλους, θα ήθελα να ευχαριστήσω τον επιβλέποντα καθηγητή αυτής της διπλωματικής εργασίας κ. Βασίλη Κουϊκόγλου για την πολύτιμη βοήθεια και καθοδήγηση του κατά την διάρκεια διεκπεραίωσης αυτής της εργασίας. Επίσης, ένα μεγάλο ευχαριστώ στους καθηγητές Albert Subia Coromina και Amaia Lusa Garcia, που με βοήθησαν και με συμβούλεψαν κατά την παραμονή μου και εργασία μου στο "Institute of Industrial Engineering and Control" στην Βαρκελώνη, όπου και πραγματοποιήθηκε το πρακτικό μέρος αυτής της εργασίας.

Φυσικά, ευχαριστώ τους φίλους μου για την ψυχική υποστήριξη και βοήθεια τους όλο αυτό τον καιρό.

Τέλος, ένα μεγάλο ευχαριστώ στους γονείς μου, Ζαφείρη και Αργυρούλα, και στην αδερφή μου Κασσιανή, για την ηθική και υλική υποστήριξη τους τα 5 χρόνια που σπούδαζα στο Πολυτεχνείο Κρήτης και πάνω από όλα για την υπομονή τους και την αγάπη τους.

ΠΡΟΛΟΓΟΣ

Η εργασία αυτή είναι μια μελέτη πάνω στα μοντέλα μικτού ακέραιου γραμμικού προγραμματισμού για την επίλυση του προβλήματος προγραμματισμού εργασιών σε γραμμές παραγωγής ή όπως αναφέρονται στην βιβλιογραφία, καταστήματα ροής. Αρχικά γίνεται μια αναφορά στο πρόβλημα με το οποίο ασχοληθήκαμε και σε σχετικές έρευνες που έχουν γίνει πάνω στο ίδιο θέμα. Στην συνέχεια παρουσιάζονται βασικές έννοιες σχετικές με το πρόβλημα και αναλύονται κάποιες από τις μεθόδους επίλυσης του που έχουν αναπτυχθεί. Τέλος, παρουσιάζονται 8 μαθηματικά μοντέλα, που συγκρίνονται στο πειραματικό κομμάτι της εργασίας και στην συνέχεια καταλήγουμε σε βασικά συμπεράσματα.

Ελπίζω, αυτή η εργασία να είναι το ίδιο ενδιαφέρουσα για τον αναγνώστη όσο ενδιαφέρουσα ήταν για μένα η πορεία της διεκπεραίωσης της.

1. Εισαγωγή στο πρόβλημα

Σε αυτή την διπλωματική εργασία, ερευνούμε την απόδοση 2 οικογενειών μοντέλων μικτού-ακέραιου γραμμικού προγραμματισμού (MILP-mixed integer linear programming) για την επίλυση του απλού προβλήματος καταστημάτων ροής, στο οποίο οι εργασίες έχουν κοινή διάταξη στις μηχανές (regular permutation flow-shop problem) και η αντικειμενική συνάρτηση είναι η ελαχιστοποίηση του χρόνου περάτωσης των εργασιών. Κοινή διάταξη σημαίνει ότι οι εργασίες εκτελούνται με την ίδια σειρά σε όλες τις μηχανές. Αν και αυτό δεν οδηγεί σε βέλτιστα αποτελέσματα, είναι μια παραδοχή που διευκολύνει την μοντελοποίηση και την επίλυση προβλημάτων βελτιστοποίησης.

Βασιζόμενοι στο άρθρο των Stafford, Tseng and Gupta, θα συγκρίνουμε 8 MILP μοντέλα με βάση τον υπολογιστικό χρόνο που απαιτεί το καθένα από αυτά για την εύρεση βέλτιστης λύσης, και σε περίπτωση που δεν μπορούμε να βγάλουμε συμπέρασμα από εκεί, με βάση την τιμή της αντικειμενικής συνάρτησης. Στόχος της έρευνας μας είναι να συγκρίνουμε τα μοντέλα χρησιμοποιώντας διαφορετικό λογισμικό από αυτό που αναφέρεται στο άρθρο και να εφαρμόσουμε κάποιες αλλαγές στα μοντέλα, βασιζόμενοι στις σημειώσεις του καθηγητή Albert Subia Coromina, του Πολυτεχνείου της Καταλονίας.

Οι οικογένειες των μοντέλων που συγκρίνουμε είναι, η οικογένεια Wagner με 3 μοντέλα προγραμματισμού και η Manne που αποτελείται από 5 μοντέλα. Για την επίλυση των προβλημάτων χρησιμοποιήθηκε το λογισμικό CPLEX σε συνδυασμό με την γλώσσα ILOG OPL, την οποία χρησιμοποιήσαμε για την κωδικοποίηση των μοντέλων και στην συνέχεια την εκτέλεση τους.

Λόγω των εφαρμογών του προβλήματος προγραμματισμού καταστημάτων ροής, τόσο στην βιομηχανία όσο και στην οικονομία, πολλές έρευνες έχουν γίνει πάνω σε αυτό με διαφορετικές υποθέσεις, διαφορετικές αντικειμενικές συναρτήσεις και χρησιμοποιώντας διαφορετικές τεχνικές βελτιστοποίησης. Με το συγκεκριμένο θέμα ασχολήθηκαν πολλοί ερευνητές κατά καιρούς, με κυριότερο τον Johnson και το άρθρο του σχετικά με την επίλυση του προβλήματος καταστημάτων ροής (flow-shop problem) πριν 50 χρόνια περίπου. Η έρευνά του αποτέλεσε έναυσμα για πολλούς ερευνητές τα

επόμενα χρόνια, με αποτέλεσμα να έχουν βρεθεί διάφορες τεχνικές επίλυσης του προβλήματος. Επίσης, διάφορες αναφορές και υποθέσεις έχουν γίνει για το πρόβλημα όπου οι εργασίες έχουν κοινή διάταξη στις μηχανές (permutation flow-shop problem) από τον Gupta.

Μαθηματικά μοντέλα προγραμματισμού έχουν προταθεί κατά καιρούς και αξιολογηθεί για την εύρεση βέλτιστης λύσης για αυτό το πρόβλημα. Ο Wagner (1959) εισήγαγε ένα μαθηματικό μοντέλο ακέραιου γραμμικού προγραμματισμού για την επίλυση του «permutation» προβλήματος καταστημάτων ροής με 3 μηχανές. Στην συνέχεια οι Bowman (1959) και Manne (1960) παρουσίασαν 2 ανταγωνιστικά μοντέλα για την επίλυση του προβλήματος καταστημάτων εργασιών. Το 1974 ο Baker, έχοντας ως βάση το μοντέλο του Wagner, το επέκτεινε για Μ μηχανές και δημιούργησε ένα μαθηματικό μοντέλο 0-1 MILP. Ο Stafford, το 1983 και στην συνέχεια το 1989, πρόσθεσε ένα σύνολο περιορισμών στο μοντέλο του Wagner ώστε να κρατήσει την πρώτη εργασία στην ακολουθία, στην γραμμή μηδέν του διαγράμματος Gantt. Έδειξε επίσης, ότι χωρίς αυτούς τους περιορισμούς το μοντέλο του Wagner δίνει μη υπαρκτές λύσεις.

Όπως είναι φυσικό, η βιβλιογραφία αναφέρει μοντέλα MILP για την επίλυση του προβλήματος καταστημάτων ροής, βασιζόμενα στις 2 οικογένειες μοντέλων, στην Wagner και στην Manne. Στην οικογένεια Wagner περιλαμβάνεται το μοντέλο WST που είναι αποτέλεσμα εργασιών των Stafford καιTseng το 2002, το μοντέλο Wilson από την έρευνα του Wilson το 1989 και τέλος το μοντέλο TS2 των Tseng και Stafford το 2001. Στην οικογένεια Manne υπάρχουν 5 μοντέλα που είναι χωρισμένα σε 2 μικρότερες κατηγορίες. Στην πρώτη κατηγορία ανήκουν τα μοντέλα SGST(PI) και SGST(pi) που αναφέρθηκαν από τους Stafford και Tseng το 1990 και στην δεύτερη κατηγορία ανήκουν τα μοντέλα LYsub(PI), LYsub(pi) και LYeq(PI) που έχουν προκύψει με τις τροποποιήσεις των Liao και You (1992). Επίσης, έχουν πραγματοποιηθεί συγκριτικές μελέτες των μοντέλων αυτών ως προς την πολυπλοκότητα τους, τις απαιτήσεις τους σε υπολογιστικό χρόνο ή και για τα δύο. Σε μια συγκριτική μελέτη ως προς την πολυπλοκότητα των μοντέλων που αναφέραμε, ο Pan κατέληξε ότι το Manne μοντέλο υπερέχει των υπολοίπων και ακολουθούν τα Wagner, Wilson, Bowman και Morten and Pentico. Επίσης ο Wilson σύγκρινε το μοντέλο του με του Stafford και κατέληξε ότι το

δικό του είναι καλύτερο για μικρού μεγέθους προβλήματα και του Stafford για μεγαλύτερα. Ακόμη, οι Stafford καιTseng σε πειράματα σύγκρισης του υπολογιστικού χρόνου κατέληξαν στο συμπέρασμα ότι το μοντέλο WST είναι καλύτερο από το SGST.

Παρόλο που το πρόβλημα προγραμματισμού καταστημάτων ροής είναι ιδιαίτερα γνωστό, στην βιβλιογραφία υπάρχουν λίγες και ανακριβείς συγκριτικές μελέτες των μοντέλων MILP προγραμματισμού του προβλήματος. Αυτό συμβαίνει διότι, οι ερευνητές χρησιμοποιούν διαφορετικές καταστάσεις των προβλημάτων για να καταλήξουν σε συμπεράσματα, το μέγεθος των προβλημάτων είναι ιδιαίτερα περιορισμένο και τέλος όλα τα μοντέλα MILP δεν χρησιμοποιούνται για την επίλυση του ίδιου προβλήματος κάτω από τις ίδιες συνθήκες.

2.Περιγραφή Καταστημάτων Ροής

2.1. Γενικά για τα προβλήματα προγραμματισμού

Ένα πρόβλημα προγραμματισμού (scheduling) εργασιών περιγράφεται από τις εξής πληροφορίες:

i. από τις εργασίες και τις διαδικασίες που θα πραγματοποιηθούν
ii. από τον αριθμό και το είδος των μηχανών που θα αποτελούν το σύστημα παραγωγής
iii. από τους κανόνες που προσδιορίζουν τον τρόπο ανάθεσης των εργασιών
iv. από τα κριτήρια αξιολόγησης του προγράμματος

Τα προβλήματα ταξινόμησης διαφέρουν ως προς τον αριθμό των εργασιών που θα εκτελεστούν, τον τρόπο με τον οποίο φτάνουν οι εργασίες στα καταστήματα και την σειρά με την οποία εμφανίζονται οι μηχανές στην διαδικασία κατεργασίας κάθε εργασίας. Ο τρόπος άφιξης των εργασιών καθορίζει αν ένα πρόβλημα είναι στατικό ή δυναμικό. Στα στατικά προβλήματα ένας ακριβής αριθμός εργασιών φτάνει ταυτόχρονα σε ένα κατάστημα το οποίο είναι άεργο και άμεσα διαθέσιμο για εργασία. Σε ένα δυναμικό πρόβλημα, το κατάστημα είναι μια συνεχής διαδικασία, όπου οι χρόνοι αφίξεων διαφέρουν.

Για την δήλωση ενός προβλήματος προγραμματισμού χρησιμοποιείται ο συμβολισμός A/B/C/D όπου κάθε μια από τις τέσσερις παραμέτρους δηλώνει τα εξής :

A την διαδικασία άφιξης των εργασιών.

B τον αριθμό των μηχανών στο κατάστημα

C τον τύπο ροής των εργασιών στο κατάστημα (στην περίπτωση των προβλημάτων καταστημάτων ροής, είναι το γράμμα F).

D το κριτήριο με το οποίο αξιολογείται το πρόβλημα

Η σειρά με την οποία εμφανίζεται η αρίθμηση των μηχανών στην κατεργασία κάθε εργασίας καθορίζει αν ένα πρόβλημα είναι πρόβλημα καταστημάτων ροής. Σε αυτή την περίπτωση όλες οι εργασίες ακολουθούν το ίδιο μονοπάτι από την μία μηχανή στην άλλη. Αυτό σημαίνει ότι υπάρχει αρίθμηση των μηχανών τέτοια ώστε ο δείκτης της

μηχανής για την διαδικασία y να είναι μεγαλύτερος από αυτόν της μηχανής για την διαδικασία χ της ίδιας εργασίας ,όταν η χ προηγείται της y.

2.2. Καταστήματα εργασιών καθορισμένης ροής (flow shops)

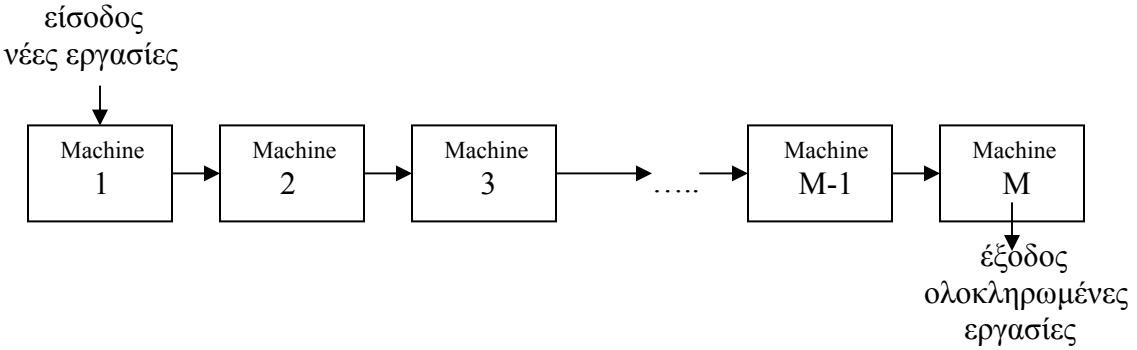
Το απλό πρόβλημα των καταστημάτων ροής (***pure flowshop problem***) αποτελείται από δύο βασικά στοιχεία : (α) από μία ομάδα Μ μηχανών και (β) από ένα σύνολο Ν εργασιών που θα πραγματοποιηθούν σε αυτές τις μηχανές. Λέγοντας εργασία, εννοούμε μια ομάδα κατεργασιών οι οποίες έχουν μια συγκεκριμένη δομή. Συγκεκριμένα, κάθε κατεργασία μετά την πρώτη έχει μία ακριβώς προηγούμενη και κάθε κατεργασία πριν από την τελευταία έχει μία επόμενη. Για αυτό τον λόγο, απαιτείται οι κατεργασίες να εκτελεστούν με συγκεκριμένη σειρά ώστε να θεωρηθεί μια εργασία ολοκληρωμένη.

Οι συνθήκες που χαρακτηρίζουν τα προβλήματα καταστημάτων ροής είναι όμοιες με αυτές του βασικού μοντέλου μίας μηχανής.

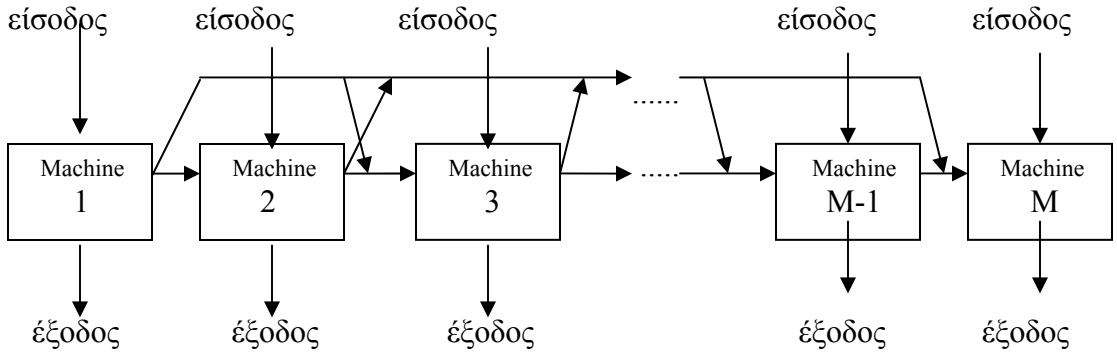
- ✓ Ένα σύνολο Ν εργασιών με πολλαπλές κατεργασίες είναι διαθέσιμο για κατεργασία την χρονική στιγμή μηδέν.(κάθε εργασία απαιτεί Μ κατεργασίες και κάθε κατεργασία απαιτεί διαφορετική μηχανή).
- ✓ Οι χρόνοι εκκίνησης των κατεργασιών είναι ανεξάρτητοι της ακολουθίας και συμπεριλαμβάνονται στον χρόνο περάτωσής της.
- ✓ Οι *δείκτες (descriptors*) των εργασιών είναι γνωστοί εκ των προτέρων
- ✓ Μ διαφορετικές μηχανές είναι συνεχώς διαθέσιμες
- ✓ Οι κατεργασίες δεν είναι μικρής προτεραιότητας, δηλαδή δεν διακόπτονται για να πραγματοποιηθεί κάποια άλλη κατεργασία

Το κατάστημα περιέχει Μ διαφορετικές μηχανές και κάθε εργασία απαιτεί από Μ κατεργασίες, κάθε μια από τις οποίες εκτελείται σε διαφορετική μηχανή. Το κατάστημα ροής χαρακτηρίζεται από την ροή εργασιών που είναι μονής διεύθυνσης. Με άλλα λόγια, ένα κατάστημα ροής περιλαμβάνει μία φυσική ακολουθία των μηχανών : είναι δυνατό να αριθμήσουμε τις μηχανές με τέτοιο τρόπο ώστε εάν η j κατεργασία οποιαδήποτε εργασίας προηγείται τις k κατεργασίας, τότε η μηχανή που απαιτείται από

την j κατεργασία έχει μικρότερο αριθμό από αυτή που απαιτείται από την k. Οι μηχανές είναι αριθμημένες ως 1,2,3....,M και οι κατεργασίες της εργασίας i αντίστοιχα ως (i, 1), (i, 2), (i, 3),……(i, M). Κάθε εργασία μπορεί να θεωρηθεί ότι αποτελείται από ακριβώς M κατεργασίες και σε περίπτωση που υπάρχουν λιγότερες, οι αντίστοιχοι χρόνοι διαδικασίας θεωρούνται ίσοι με το μηδέν. Στην συνέχεια παραθέτουμε στο σχήμα 2.1 την ροή των εργασιών στο απλό πρόβλημα των καταστημάτων ροής, όπου όλες οι εργασίες απαιτούν μία κατεργασία σε κάθε μηχανή και στο σχήμα 2.2 την ροή των εργασιών σε ένα πιο γενικό πρόβλημα καταστημάτων ροής. Σε αυτή την περίπτωση πρέπει να σημειώσουμε ότι είναι πιθανόν οι εργασίες να απαιτούν λιγότερες από M κατεργασίες, ότι οι κατεργασίες μπορεί να μην απαιτούν συνεχόμενες μηχανές στην αριθμημένη ακολουθία και τέλος ότι η αρχική και τελική κατεργασία μπορεί να μην αντιστοιχεί πάντα στην πρώτη και στην M μηχανή.



Σχήμα 2.1 : pure flow shop



Σχήμα 2.2: general flow shop

Στόχος είναι ο προσδιορισμός της σειράς και του χρόνου περάτωσης των εργασιών στις μηχανές του καταστήματος ροής, βάσει κάποιων υποθέσεων που έχουν γίνει. Το πρόβλημα καταστημάτων ροής με κριτήριο τον χρόνο περάτωσης των εργασιών στην διαδικασία ή το μέγιστο χρόνο περάτωσης των εργασιών εκφράζεται ως $n/m/F/c_{max}$ ή όμοια $F// c_{max}$, όπου δηλώνεται ένα πρόβλημα καταστημάτων ροής με n εργασίες και m μηχανές. Στην περίπτωση μίας μηχανής, υπάρχει ένα προς ένα σχέση ανάμεσα στην ακολουθία των εργασιών και στον συνδυασμό των δεικτών τους 1,2,....,n. Για την εύρεση μιας ιδανικής ακολουθίας είναι απαραίτητο να εξετάσουμε ,στην πιο απλή περίπτωση, κάθε μία από τις ακολουθίες που ανταποκρίνεται σε n! διαφορετικούς συνδυασμούς. Στο πρόβλημα καταστημάτων ροής υπάρχουν n! πιθανές διαφορετικές ακολουθίες εργασιών για κάθε μηχανή και έτσι $(n!)^m$ διαφορετικά προγράμματα να εξεταστούν. Όμως, οι περισσότεροι ερευνητές έχουν επικεντρώσει τις έρευνες τους, στην ανάπτυξη προγραμμάτων για καταστήματα ροής με κοινή διάταξη των εργασιών στις μηχανές (***permutation flow-shop problem***). Αυτά τα προβλήματα μπορούν θεωρηθούν ως κλασσικά προβλήματα καταστημάτων ροής με την παραδοχή ότι οι εργασίες πρέπει να επεξεργάζονται με την ίδια ακολουθία από κάθε μηχανή. Αποτέλεσμα αυτού είναι η μείωση του χώρου των πιθανών λύσεων σε n!.

Εκτός από το κλασσικό πρόβλημα καταστημάτων ροής υπάρχουν και κάποιες παραλλαγές του όπως, τα παραλειπόμενα καταστήματα ροής (skip shops), όπου κάποιες εργασίες πιθανόν να παραλείψουν κάποιες από τις μηχανές, τα καταστήματα ροής με επανεισόδους (reentrant flow shops), όπου υπάρχουν μηχανές από τις οποίες οι εργασίες μπορεί να περάσουν περισσότερες από μια φορές, τα σύνθετα καταστήματα ροής (compound flow shops), στα οποία οι μηχανές σε σειρά μπορούν να αντικατασταθούν από ένα σύνολο μηχανών που μπορεί να είναι, είτε παράλληλες μηχανές είτε μια στοίβα που ακολουθείται από παράλληλες μηχανές. Τέλος, τα καταστήματα ροής πεπερασμένων ουρών (finite queue) όπου πριν από κάθε μηχανή, όχι μόνο στην πρώτη, υπάρχει περιορισμένος χώρος αποθήκευσης–αναμονής. Ιδιαίτερη περίπτωση είναι αυτή όπου υπάρχει αποθηκευτικός χώρος μόνο στην πρώτη μηχανή.

2.3 Προβλήματα καταστημάτων εργασιών καθορισμένης ροής

Τα κυριότερα προβλήματα των καταστημάτων ροής είναι τα εξής:

1. Το κλασσικό $N/2/F_{max}$ πρόβλημα όπου αναζητείται η ακολουθία N εργασιών που διέρχονται από 2 μηχανές, ενδιάμεσα των οποίων υπάρχει αποθηκευτικός χώρος άπειρης χωρητικότητας, η οποία ελαχιστοποιεί το μέγιστο χρόνο ροής Fmax.
2. Το $N/M/F_{max}/B_i \neq 0, \infty$ όπου N εργασίες πρέπει να διέλθουν από M μηχανές με δεδομένο ότι υπάρχουν μεταξύ των μηχανών αποθηκευτικοί χώροι με πεπερασμένη χωρητικότητα.
3. Το $N/M/F_{max}/ B_i=0$ όπου και πάλι έχουμε N εργασίες που διέρχονται από M μηχανές με την διαφορά ότι δεν υπάρχουν αποθήκες ανάμεσα στις μηχανές.

2.4 Προγράμματα Διάταξης

Το πρόβλημα καταστημάτων ροής, είναι ένα πρόβλημα στο οποίο κάποιες φορές είναι αρκετή η θεώρηση ενός προγράμματος με κοινή διάταξη των εργασιών και σε γενικές γραμμές υπάρχει μια συγκεκριμένη διάταξη για τις εργασίες στις αρχικές και τελικές μηχανές. Στόχος κάθε φορά είναι η ελαχιστοποίηση ενός κοινού μέτρου απόδοσης. Ένα μέτρο απόδοσης που περιγράφει το κόστος λειτουργίας, ονομάζεται κοινό (regular) αν είναι αύξουσα συνάρτηση των χρόνων περάτωσης των εργασιών. Με απλά λόγια, όσο καθυστερούν οι εργασίες τόσο αυξάνεται το κόστος. Τα μέτρα απόδοσης είναι κοινά στις περισσότερες περιπτώσεις. Υπάρχουν δύο θεωρήματα για τα ομαλά προβλήματα προγραμματισμού καταστημάτων ροής.

Θεώρημα 1

Όταν προγραμματίζουμε ένα N/M/F πρόβλημα με όλες τις εργασίες ταυτόχρονα διαθέσιμες, για να ελαχιστοποιήσουμε οποιοδήποτε κοινό μέτρο απόδοσης πρέπει να σκεφτούμε μόνο προγράμματα στα οποία έχουμε την ίδια σειρά των εργασιών στις δύο πρώτες μηχανές.

Βέβαια το παραπάνω θεώρημα δεν αποκλείει το ενδεχόμενο να υπάρχουν καλά προγράμματα με διαφορετική διάταξη των δύο πρώτων μηχανών. Απλά, είναι αρκετό

κάποιος να θεωρήσει μόνο τα σχέδια με την συγκεκριμένη διάταξη των πρώτων μηχανών, μιας και για οποιοδήποτε άλλη διάταξη υπάρχει η αντίστοιχη και εύκολα πραγματοποιήσιμη διάταξη αυτού του τύπου που είναι εξίσου καλή και πιθανόν καλύτερη.

Θεώρημα 2

Όταν προγραμματίζουμε ένα $N/M/F/F_{max}$ πρόβλημα με όλες τις εργασίες ταυτόχρονα διαθέσιμες, θεωρούμε προγράμματα στα οποία υπάρχει η ίδια ακολουθία εργασιών στις δύο πρώτες και στις δύο τελευταίες μηχανές του καταστήματος.

2.5 Μέτρα απόδοσης

Η ποικιλία των κριτηρίων που προέρχονται από την θεωρία του προγραμματισμού αντανακλά μόνο μερικά την ποικιλία των διαφόρων καταστάσεων που προκύπτουν σε πραγματικά προβλήματα προγραμματισμού. Παρόλα αυτά, η επιλογή των κριτηρίων είναι αναμφισβήτητα επηρεασμένη από την προσδοκία εύρεσης μίας λύσης.

Είναι σημαντικό να διαχωρίσουμε τις μεταβλητές που καθορίζουν το πρόβλημα, οι οποίες δίνονται από κάποιο εξωτερικό παράγοντα, από τις μεταβλητές που περιγράφουν την λύση και προκύπτουν από την προγραμματισμένη διαδικασία.

2.5.1 Μεταβλητές που καθορίζουν ένα πρόβλημα προγραμματισμού

Η περιγραφή του προβλήματος ξεκινάει με ένα κατάστημα εργασιών και ένα σετ εργασιών. Οι κάθε μηχανή αναγνωρίζεται από ένα ακέραιο αριθμό από το 1,2,....,m και ανάλογα οι εργασίες από 1,2,.....,n. Τα σχετικά χαρακτηριστικά της εργασίας i που δίνονται ως μέρος της περιγραφής του προβλήματος δηλώνονται από τις εξής παραμέτρους:

r_i ο χρόνος άφιξης. Είναι ο χρόνος κατά τον οποίο η εργασία φτάνει στο κατάστημα από κάποια εξωτερική διαδικασία και εκφράζει τον νωρίτερο χρόνο κατά τον οποίο μπορεί να ξεκινήσει η πρώτη λειτουργία της εργασίας.

d_i η προθεσμία παράδοσης. Είναι η χρονική στιγμή που κάποιος εξωτερικός παράγοντας θα επιθυμούσε να έφευγε η εργασία από το κατάστημα. Είναι ο χρόνος κατά τον οποίο η επεξεργασία της τελευταίας διαδικασίας πρέπει να ολοκληρωθεί.

α_i ισούται με την διαφορά d_i- r_i και εκφράζει τον ολικό χρόνο που επιτρέπεται να βρίσκεται η εργασία στο κατάστημα.

$p_{i,j}$ είναι ο χρόνος επεξεργασίας που απαιτείται από την μηχανή $m_{i,j}$ για να εκτελέσει την κατεργασία της εργασίας i .

2.5.2 Μεταβλητές για την περιγραφή της λύσης ενός προβλήματος προγραμματισμού.

Αρχικά είναι απαραίτητο να προσδιορίσουμε πόσο χρόνο θα περιμένει η εφαρμογή κάθε εργασίας πριν ξεκινήσει η διαδικασία και δηλώνεται ως εξής:

$W_{i,j}$, ο χρόνος αναμονής της εκκίνησης της επεξεργασία της j διαδικασίας της εργασίας i. Ο χρόνος κατά τον οποίο η εργασία θα πρέπει να περιμένει μετά την συμπλήρωση της (j-1) διαδικασίας ώστε να αρχίσει με την j. Ο συνολικός χρόνος αναμονής μιας εργασίας ισούται με το άθροισμα των χρόνων αναμονής των διαδικασιών της εργασίας:

$$W_i = \sum_{j=1}^{g\,i} W_{i,j}$$

Το αποτέλεσμα της διαδικασίας προγραμματισμού για ένα συγκεκριμένο πρόβλημα, το πρόγραμμα, είναι πλήρως ορισμένο αν δοθεί ένα σετ από $W_{i,j}$. Οι επόμενες μεταβλητές που αναφέρονται παρακάτω είναι όλες συναρτήσει των $W_{i,j}$.

C_i ο χρόνος περάτωσης, που είναι ο χρόνος στον οποίο η εργασία i θα ολοκληρώσει την διαδικασία και είναι $C_i = r_i + p_i + W_i$

F_i ο χρόνος ροής, που εκφράζει το χρόνο που η εργασία ij βρίσκεται στο σύστημα και είναι $F_i = p_i - W_i = C_i - r_i$

L_i η βραδύτητα, το ποσό του χρόνου (θετικό ή αρνητικό) κατά το οποίο η πραγματοποίηση της εργασίας ξεπερνά τον καθορισμένο χρόνο και δίνεται $L_i = C_i - d_i = F_i - a_i$

T_i η καθυστέρηση της εργασίας i και είναι $T_i = \max(0, L_i)$

E_i η ενωρίτητα της εργασίας i και είναι $E_i = \max(0, -L_i)$

Από τα παραπάνω μέτρα, το πιο σημαντικό είναι ο χρόνος περάτωσης της εργασίας, μιας και όλα τα υπόλοιπα καθορίζονται από αυτό.

Ο χρόνος ροής μετρά την ανταπόκριση του συστήματος σε πιθανές απαιτήσεις για εξυπηρέτηση. Είναι το ποσό του χρόνου μεταξύ της άφιξης και της αναχώρησης της

εργασίας από το σύστημα. Είναι φανερό πως είναι άρρηκτα συνδεδεμένος με το work-in-process (WIP) κόστος.

Η βραδύτητα μετρά την συμφωνία του προγράμματος με το δοθέν πρόγραμμα από έναν εξωτερικό παράγοντα. Στην ουσία βραβεύει τις εργασίες που τελειώνουν γρήγορα και τιμωρεί αυτές που καθυστερούν.

Η καθυστέρηση αντανακλά το γεγονός ότι, σε πολλές περιπτώσεις, συγκεκριμένα μειονεκτήματα και κόστη θα σχετίζονται με θετική βραδύτητα και κανένα μειονέκτημα ή πλεονέκτημα με θετική.

Η ενωρίτητα εκφράζει την ποινή που δίνεται σε κάποια εργασία όταν τελειώνει νωρίτερα από τον χρόνο που έχει καθοριστεί.

2.5.3 Αντικειμενικές συναρτήσεις

Σε αυτό το σημείο θα αναφέρουμε τις βασικότερες αντικειμενικές συναρτήσεις τις οποίες βελτιστοποιούμε στην επίλυση ενός προβλήματος προγραμματισμού καταστημάτων ροής.

- *Χρόνος περάτωσης (makespan)*. Το πρόγραμμα με τον μικρότερο χρόνο περάτωσης συχνά αντικαθιστά το πρόγραμμα με την μεγαλύτερη χρησιμοποίηση. Η ιδέα είναι ότι τελειώνοντας το σετ των εργασιών νωρίτερα επιτρέπεται σε νέες εργασίες να ξεκινήσουν. Εκφράζεται από την σχέση $C_{\max} = \max_i(C_i)$
- *Μέσος χρόνος ροής με βάρη και Καθυστέρηση με βάρη*. Είναι και οι δύο πολλοί δημοφιλής αντικειμενικές για πολλούς λόγους. Πρώτον, μπορούν να χρησιμοποιηθούν πολύ εύκολα μιας και οι ιδανικές λύσεις τους είναι όμοιες και διαισθητικές. Δεύτερον, είναι ιδιαίτερα δυνατές από την άποψη ότι τα προγράμματα που είναι ιδανικά για αυτές παράγουν προγράμματα για προβλήματα με λίγο διαφορετικές αντικειμενικές. Οι σχέσεις για αυτές είναι $F_{wt} = \sum_i w_i F_i$ και $L_{wt} = \sum_i w_i L_i$.
- *Καθυστέρηση με βάρη/ άλλα μέτρα καθυστέρησης*. Σε πολλές περιπτώσεις η καθυστέρηση με βάρη είναι καλή αντικειμενική αλλά, προβλήματα που την χρησιμοποιούν είναι πολύ δύσκολο να επιλυθούν ακριβώς. Η καθυστέρηση-πρόωρη άφιξη με βάρη είναι επίσης πολύ σημαντική όταν ο πελάτης δεν

θέλει εργασίες με καθυστέρηση, αλλά ούτε τις δέχεται νωρίτερα. Ο αριθμός των καθυστερημένων εργασιών με βάρος αντανακλά την κατάσταση κατά τη οποία οι πελάτες αρνούνται να δεχτούν καθυστερημένες εργασίες και οι παραγγελίες χάνονται. Συγκεκριμένα, η μαθηματική έκφραση κάθε μιας αντικειμενικής είναι:

Καθυστέρηση με βάρη $T_{wt} = \sum_i w_{Ti} T_i$

Καθυστέρηση με βάρη και πρόωρη άφιξη με βάρη $ET_{wt} = \sum_i (w_{Ei} E_i + w_{Ti} T_i)$

Αριθμός των καθυστερημένων εργασιών με βάρος $N_{wt} = \sum_i w_{Ni} \delta(\mathrm{T}_i)$

όπου $\delta(x) = 1$ αν x>0

και $\delta(x) = 0$ αλλιώς

- *Μέγιστος χρόνος ροής , μέγιστη βραδύτητα και μέγιστη καθυστέρηση.* Η ελαχιστοποίηση της μέγιστης καθυστέρησης είναι σημαντική όταν οι πελάτες είναι ανεκτικοί σε μικρές καθυστερήσεις αλλά αναστατώνονται με μεγάλες. Η ελαχιστοποίηση της μέγιστης βραδύτητας είναι σημαντική διότι είναι ένα σχετικά εύκολο πρόβλημα και μπορεί να χρησιμοποιηθεί στην επίλυση άλλων προβλημάτων. Παραδείγματος χάρη, η ελαχιστοποίηση του μέγιστου χρόνου ροής μπορεί να πραγματοποιηθεί θέτοντας τους καθορισμένους χρόνους ίσους με τους χρόνους άφιξης και έτσι να ελαχιστοποιήσουμε την μέγιστη βραδύτητα. Τέλος, υπάρχει μια τεχνική χρήση της ελαχιστοποίησης της μέγιστης βραδύτητας, στην επίλυση προβλημάτων με πολλές πηγές με αντικειμενική συνάρτηση τον χρόνο περάτωσης. Η μαθηματική έκφραση τους είναι

  μέγιστος χρόνος ροής $F_{\max} = \max_i \{F_i\}$

  μέγιστη βραδύτητα $L_{\max} = \max_i \{L_i\}$

  μέγιστη καθυστέρηση $T_{\max} = \max_i \{T_i\}$.

2.6 Η περίπτωση που ελέγξαμε

Κατά την εκπόνηση αυτής της διπλωματικής εργασίας, ασχοληθήκαμε με τον προγραμματισμό καταστημάτων ροής εργασιών κοινής διάταξης (permutation flowshop problem) με αντικειμενική συνάρτηση, την ελαχιστοποίηση του χρόνου περάτωσης των εργασιών (makespan).

Το κριτήριο αυτό, είναι από τα πιο μελετημένα για προβλήματα καταστημάτων ροής εξαιτίας της απλότητας του και της χρησιμότητας του. Επίσης, είναι η μόνη αντικειμενική συνάρτηση που είναι τόσο απλή ώστε να μπορεί να δώσει αναλυτικά αποτελέσματα ακόμη και για προβλήματα πολλών μηχανών και αρκετά απλή ώστε να μπορούμε να χρησιμοποιήσουμε τον αλγόριθμο διακλάδωσης-φράγματος για μέσου μεγέθους προβλήματα.

3. Ανασκόπηση μεθοδολογιών επίλυσης του προβλήματος

Σε αυτό το σημείο θα γίνει μια προσπάθεια να γνωρίσει ο αναγνώστης τις μεθοδολογίες που έχουν εφαρμοστεί για την επίλυση του προβλήματος προγραμματισμού των καταστημάτων ροής, με αντικειμενική συνάρτηση την ελαχιστοποίηση του χρόνου περάτωσης των εργασιών (makespan).

Αρχικά θα αναφερθούμε στον πολύ γνωστό αλγόριθμο του Johnson που εφαρμόζεται στο $N/2/F/c_{max}$ πρόβλημα, όπου έχουμε N εργασίες, 2 μηχανές και το κριτήριο του ελάχιστου χρόνου περάτωσης των εργασιών. Στην συνέχεια παραθέτουμε τους διάφορες ευρετικές μεθόδους όπως τον αλγόριθμο του Palmer, του Gupta, CDS αλγόριθμο και NEH και αμέσως μετά, παρατίθενται κάποιοι πιο μοντέρνοι ευρετικοί αλγόριθμοι που έχουν ονομαστεί μεθευρετικοί και κάποιοι επαναστατικοί αλγόριθμοι, που έχουν εφαρμοστεί για τη λύση του προβλήματος το οποίο συζητάμε σε αυτή εδώ την διπλωματική. Τέλος, γίνεται αναφορά στα μοντέλα του μικτού-ακέραιου γραμμικού προγραμματισμού (MILP) και δίνεται και ένα παράδειγμα για καλύτερη κατανόηση, μιας και στο πρακτικό μέρος αυτής εδώ της εργασίας ασχολούμαστε με την αξιολόγηση της απόδοσης δύο οικογενειών μοντέλων MILP για την επίλυση του προβλήματος ελαχιστοποίησης του χρόνου περάτωσης σε καταστήματα εργασιών ροής.

3.1 Αλγόριθμος Johnson

Η θεωρία προγραμματισμού, στην περιοχή των προβλημάτων καταστημάτων ροής έχει επηρεαστεί από την εργασία του Johnson (1954). Η εργασία του είχε σημαντικά πλεονεκτήματα, τα οποία της έδιναν την δυνατότητα να ασκεί επιρροή σε επόμενες έρευνες. Πρώτον, έδινε έμφαση στις ιδιότητες του προγραμματισμού εργασιών κοινής διάταξης και επικέντρωνε την έρευνα του σε προβλήματα εύχρηστου μεγέθους. Δεύτερον, η ανάλυση για δύο μηχανές φαινόταν να έχει συλλάβει την ουσία των μεγαλύτερων προβλημάτων. Για αυτούς τους λόγους, η εφαρμογή του αλγορίθμου για

την ανακάλυψη ευρετικών τεχνικών για μεγαλύτερου μεγέθους προβλήματα είχε πάντα αξιοσημείωτη επιτυχία.

Ο κανόνας του Johnson λέει ότι η εργασία i προηγείται της εργασίας j σε μια ιδανική ακολουθία εργασιών αν: $\min\{t_{i1}, t_{j2}\} \le \min\{t_{i2}, t_{j1}\}$. Εφαρμόζοντας αυτό τον κανόνα στο πρόβλημα καταστημάτων εργασιών καθορισμένης ροής με δύο μηχανές και κριτήριο απόδοσης την ελαχιστοποίηση του χρόνου περάτωσης ή αλλιώς το $N/2/F/c_{max}$ πρόβλημα, τότε η λύση του μπορεί να προκύψει από την εφαρμογή του εξής αλγορίθμου:

Βήμα 1: Βρες το $\min i\{ti1, ti2\}$

Βήμα 2α: εάν ο ελάχιστος χρόνος περάτωσης απαιτεί την μηχανή 1, τοποθέτησε την εργασία στην πρώτη ελεύθερη θέση στην ακολουθία και πήγαινε στο βήμα 3

Βήμα 2β: εάν ο ελάχιστος χρόνος περάτωσης απαιτεί την μηχανή 2, τοποθέτησε την εργασία στην τελευταία ελεύθερη θέση στην ακολουθία και πήγαινε στο βήμα 3

Βήμα 3: απομάκρυνε την εργασία που έχεις επιλέξει από την λίστα εργασιών και επέστρεψε στο βήμα 1. Συνέχισε μέχρι να συμπληρωθούν όλες θέσεις στην ακολουθία.

Ο παραπάνω αλγόριθμος μπορεί να χρησιμοποιηθεί και στην περίπτωση της επίλυσης του $N/3/F/c_{max}$ προβλήματος όταν η πρώτη ή τρίτη μηχανή επικρατούν της δεύτερης. Δηλαδή θα ισχύει:

1. εάν $\min_k\{t_{k1}\} \ge \max_k\{t_{k2}\}$ τότε η εργασία i προηγείται της εργασία j σε ένα ιδανικό πρόγραμμα όταν $\min\{t_{i1}+t_{i2}, t_{j2}+t_{j3}\} \le \min\{t_{i2}+t_{i3}, t_{j1}+t_{j2}\}$

2.εάν $\min_k\{t_{k3}\} \ge \max_k\{t_{k2}\}$ τότε η εργασία i προηγείται της εργασία j σε ένα ιδανικό πρόγραμμα όταν $\min\{t_{i1}+t_{i2}, t_{j2}+t_{j3}\} \le \min\{t_{i2}+t_{i3}, t_{j1}+t_{j2}\}$

Για την εφαρμογή αυτών των αποτελεσμάτων σε ένα αλγόριθμο, είναι δυνατό να χρησιμοποιήσεις τον αλγόριθμο που περιγράφηκε παραπάνω αλλάζοντας το πρώτο βήμα του. Συγκεκριμένα, θα ξεκινήσουμε ψάχνοντας το ελάχιστο με βάση αυτή την σχέση $\min\{t_{i1}+t_{i2}, t_{i2}+t_{i3}\}$ και όχι ψάχνοντας τον ελάχιστο χρόνο περάτωσης. Επιπρόσθετα, αν δεν υπάρχει κυριαρχία της δεύτερης μηχανής και η εφαρμογή του αλγορίθμου του Johnson φέρει την ίδια ιδανική ακολουθία εργασιών για τα υπό-προβλήματα δύο μηχανών, που εκφράζονται από τα σετ $\{t_{i1}, t_{i2}\}$ και $\{t_{i2}, t_{i3}\}$, τότε η ακολουθία αυτή είναι η ιδανική και για το πρόβλημα των τριών μηχανών.

Σε περιπτώσεις που οι παραπάνω συνθήκες δεν ισχύουν ή αριθμός των μηχανών είναι μεγαλύτερος πρέπει να καταφύγουμε στους ευρετικούς αλγορίθμους, τους οποίους περιγράφουμε στην επόμενη παράγραφο.

3.2 Ευρετικοί αλγόριθμοι

Μετά την παρουσίαση του αλγόριθμου του Johnson θα συνεχίσουμε με τους ευρετικούς αλγορίθμους, οι οποίοι έχουν προταθεί σε πολλές έρευνες. Περιγράφουμε τους πρώτους αλγορίθμους αυτής της κατηγορίας, οι οποίοι χρησιμοποιήθηκαν ως βάση από τους επόμενους ερευνητές για την δημιουργία νέων τεχνικών προγραμματισμού.

3.2.1 Αλγόριθμος του Palmer

Ο Palmer πρότεινε τον αλγόριθμό του χρησιμοποιώντας την ιδέα των «δεικτών κλίσης» για κάθε εργασία, οι οποίοι μετράνε αν η εργασία πραγματοποιείται από ένα μικρότερο σε ένα μεγαλύτερο χρόνο επεξεργασίας στην ακολουθία. Η ακολουθία στην συνέχεια κατασκευάζεται με βάση την φθίνουσα διάταξη των δεικτών κλίσης, με την σκέψη ότι οι εργασίες που έχουν την τάση να κινούνται από τους μικρότερους στους μεγαλύτερους χρόνους επεξεργασίας στην ακολουθία των διαδικασιών, πραγματοποιούνται πιο γρήγορα. Ο δείκτης κλίσης που πρότεινε ο Palmer για την εργασία i υπολογίζεται από την σχέση :

$$SI_i = -\sum_{j=1}^{m} [m-(2j-1)]t_{ij} / 2$$

όπου το t_{ij} είναι ο χρόνος επεξεργασίας της εργασίας i στην μηχανή j και m είναι ο αριθμός των μηχανών. Η διάταξη του προγράμματος βρίσκεται με βάση την εξής σειρά των εργασιών : $SI_{[1]} \geq SI_{[2]} \geq ..... \geq SI_{[n]}$

3.2.2 Αλγόριθμος Gupta

Το 1971 ο Gupta πρότεινε πως για περιπτώσεις περισσότερων των δύο μηχανών ο δείκτης των εργασιών μπορεί να υπολογιστεί από την παρακάτω σχέση:

$$SI_i = e_i / \min_{1 \leq k \leq m-1} \{t_{ik} + t_{i(k+1)}\} \text{ όπου } e_i = \begin{cases} 1 \text{ , αν } t_{i1} < t_{im} \end{cases}$$

$$e_i = \begin{cases} \text{-1, αν } t_{i1} \geq t_{im} \end{cases}$$

τότε η σειρά των εργασιών με βάση την σχέση $SI_{[1]} \geq SI_{[2]} \geq ..... \geq SI_{[n]}$

δίνει ένα καλό πρόγραμμα διάταξης(permutation schedule). Ο Gupta μετά από μια σειρά πειραμάτων, κατέληξε πως ο δικός του αλγόριθμος δίνει καλύτερα αποτελέσματα από αυτόν του Palmer.

3.2.3 CDS αλγόριθμος

Ο Campbell et al.(1970) πρότεινε έναν αλγόριθμο για προβλήματα ελαχιστοποίησης του χρόνου περάτωσης, ο οποίος λέγεται CDS. Χρησιμοποιώντας δύο βασικές αρχές αυτή η διαδικασία δίνει πολύ καλές λύσεις. Οι αρχές που χρησιμοποιεί είναι:

1. ο αλγόριθμος Johnson με έναν ευρετικό τρόπο και
2. σε γενικές γραμμές δημιουργεί διάφορα προγράμματα από τα οποία θα πρέπει να επιλεγεί το καλύτερο.

Ο συγκεκριμένος αλγόριθμος δημιουργεί m-1 τεχνητά προβλήματα δύο μηχανών και στην συνέχεια τα επιλύει εφαρμόζοντας τον αλγόριθμο του Johnson. Η καλύτερη από τις λύσεις που προκύπτουν από τα m-1 προβλήματα είναι και η λύση του προβλήματος m μηχανών. Οι χρόνοι επεξεργασίας των τεχνητών υπό-προβλημάτων για την ioστή εργασία στην j μηχανή στο επίπεδο k υπολογίζονται από την σχέση :

$$t'_{i1} = \sum_{j=1}^{k} t_{ij} \quad \text{και} \quad t'_{i2} = \sum_{j=m-k+1}^{m} t_{ij} \, .$$

Συγκρίνοντας αυτό τον αλγόριθμο με αυτόν του Palmer, ο Campbell κατέληξε πως ο δικός του είναι πιο αποτελεσματικός τόσο για μεγάλα όσο και για μικρά προβλήματα, καθώς επίσης πως δεν υπάρχει διαφορά στον υπολογιστικό χρόνο όταν ο αριθμός των εργασιών είναι μικρότερος του 20.

3.2.4 ΝΕΗ αλγόριθμος

Nawaz et al(1983) πρότειναν τους λεγόμενους νέους ευρετικούς αλγορίθμους, βασιζόμενοι στην υπόθεση ότι μια εργασία που έχει μεγαλύτερο ολικό χρόνο επεξεργασίας σε όλες τις μηχανές, θα πρέπει να έχει μεγαλύτερη προτεραιότητα από μια άλλη με μικρότερο χρόνο. Έτσι πρότειναν την τοποθέτηση των εργασιών με φθίνουσα

σειρά του ολικού χρόνου επεξεργασίας. Στην συνέχεια, ο αλγόριθμος χρησιμοποιεί την ιδέα της μερικής ακολουθίας, η οποία προκύπτει από την σειρά των εργασιών που αναφέρθηκε προηγουμένως. Μια ακολουθία κατασκευάζεται με την εισαγωγή μιας εργασίας κάθε φορά, από την απρογραμμάτιστη ακολουθία στην μερική ακολουθία. Εάν υπήρχαν k εργασίες στην προηγούμενη μερική ακολουθία, με κάθε εισαγωγή εργασίας βρίσκουμε την βέλτιστη μερική ακολουθία ανάμεσα στις k+1 και την επιλέγουμε. Η νέα εργασία μπορεί να τοποθετηθεί σε μία από τις δυνατές k+1 θέσεις στην μερική ακολουθία και αφού διαλέξουμε την καλύτερη θέση, όσο αφορά τον υπολογιζόμενο χρόνο περάτωσης (makespan), η ακολουθία που προκύπτει ετοιμάζεται για την συνέχεια της διαδικασίας, που είναι η εισαγωγή νέας εργασίας. Η εισαγωγή της εργασίας γίνεται με βάση την τοποθέτηση των εργασιών ως προς φθίνουσα σειρά ολικού χρόνου επεξεργασίας. Η ευρετική λύση του προβλήματος αποκτάται μετά από $\frac{n(n-1)}{2}-1$ συνολικές απαριθμήσεις.

3.3 Μεθευρετικοί αλγόριθμοι και εξελικτικές προσεγγίσεις

Μετά την αναφορά στις ευρετικές μεθόδους επίλυσης του προβλήματος που συζητάμε, θα περάσουμε σε μία περιγραφή των σημαντικότερων μεθευρετικών αλγορίθμων που δημιουργήθηκαν για την λύση του ίδιου προβλήματος. Με την χρήση αυτών των αλγορίθμων δίνεται η δυνατότητα στους ερευνητές να ξεπεράσουν τα τοπικά βέλτιστα και να βελτιώσουν τις αρχικές δυνατές λύσεις. Υπάρχουν πολλοί αλγόριθμοι αυτής της κατηγορίας που έχουν χρησιμοποιηθεί σε προβλήματα προγραμματισμού καταστημάτων εργασιών καθορισμένης ροής όπως :
Προσομοιωμένη Ανόπτηση (Simulated Annealing (SA)), Γενετικοί Αλγόριθμοι, Περιορισμένη Αναζήτηση (Tabu search), αλγόριθμοι απληστίας και ο αλγόριθμος των μυρμηγκιών (Ant Colony Algorithm). Τα στοιχεία που χρησιμοποιούν αυτοί οι αλγόριθμοι είναι

- Χρησιμοποιούν έναν αριθμό από επαναληπτικές δοκιμές
- Περιλαμβάνουν έναν ή περισσότερους πράκτορες

- Λειτουργούν βάση ενός μηχανισμού συνεργασίας και ανταγωνισμού
- Περιλαμβάνουν διαδικασίες αυτό-τροποποιήσεων των ευρετικών παραμέτρων ή ακόμα και της αναπαράστασης του προβλήματος.

3.3.1 Προσομοιωμένη Ανόπτηση (Simulated Annealing)

Ο πρώτος που πρότεινε την μέθοδο της Προσομοιωμένης Ανόπτησης (Simulated Annealing) ήταν ο Kirkpatrick, ενώ ο Creny (1985) θεώρησε ότι υπάρχει αναλογία ανάμεσα στην διαδικασία ανόπτησης των υγρών και στην διαδικασία επίλυσης προβλημάτων συνδυαστικής βελτιστοποίησης. Η βασική διαδικασία που ακολουθείται με αυτό τον αλγόριθμο περιγράφεται παρακάτω.

Ξεκινάει από την αρχική λύση του προβλήματος και γεννάει μια νέα δοκιμαστική λύση, από την γειτονιά της τρέχουσας λύσης. Αν η νέα λύση είναι καλύτερη της αρχικής την δεχόμαστε και την κρατάμε ως νέα τρέχουσα λύση. Διαφορετικά, μπορεί είτε να απορριφθεί είτε να γίνει αποδεκτή ανάλογα με την πιθανότητα αποδοχής, η οποία καθορίζεται από την διαφορά που υπάρχει ανάμεσα στις αντικειμενικές συναρτήσεις των δύο λύσεων και από μια παράμετρο ελέγχου που ονομάζεται θερμοκρασία, ακολουθώντας την ορολογία στην θερμοδυναμική. Η διαδικασία ξεκινά από την αρχή για την νέα τρέχουσα λύση. Αρχικά, η θερμοκρασία έχει μια υψηλή τιμή, όπως συμβαίνει στην ανόπτηση, έτσι ώστε όλες οι κινήσεις να γίνουν αποδεκτές. Στην συνέχεια μειώνεται σταδιακά ώστε να μην επιτρέπεται άλλη μετακίνηση. Σε γενικές γραμμές η διαδικασία προσομοιωμένης ανόπτησης μπορεί να περιγραφεί από τα εξής 5 βήματα:

1. Αρχικοποίηση: παράμετροι του προγράμματος ανόπτησης
2. Επιλογή ενός επαναληπτικού μηχανισμού: ένας απλός τρόπος για να γεννάτε μια πτώση από το τρέχον επίπεδο στο επόμενο, με μικρή διαταραχή
3. Αξιολόγηση του νέου επιπέδου, υπολογισμός ΔΕ= (αξία τρέχον επιπέδου-αξία νέου)
4. Εάν το νέο επίπεδο είναι καλύτερο κάνε το τρέχον. Αλλιώς, με βάση πιθανότητες δέξου το ή απόρριψε το
5. Με βάση κανόνα παύσης είτε σταμάτα τις επαναλήψεις είτε συνέχισε από το βήμα 2.

Για την εφαρμογή του αλγορίθμου σε προβλήματα προγραμματισμού καταστημάτων ροής είναι πολύ σημαντικό να καθοριστούν τα εξής :

- Αρχική τιμή της θερμοκρασίας
- Η συνάρτηση που θα καθορίζει πως θα αλλάζει η θερμοκρασία
- Ο αριθμός των εσωτερικών επαναλήψεων
- Το κριτήριο για να σταματήσει ο αλγόριθμος

Επίσης πρέπει να γνωρίζουμε τον τρόπο γέννησης γειτονικών λύσεων, ποια είναι η μείωση της θερμοκρασίας καθώς και την σχέση μεταξύ ποιότητας της λύσης και χρόνου υπολογισμού της.

3.3.2 Γενετικοί αλγόριθμοι

Οι γενετικοί αλγόριθμοι (GA) περιγράφηκαν αρχικά από τον Holland (1975), ο οποίος βασίστηκε στην διαδικασία βιολογικής εξέλιξης. Ο Goldberg τους περιέγραψε ως αλγορίθμους αναζήτησης για βελτιστοποίηση. Κατά την τελευταία δεκαετία έχουν εφαρμοστεί σε πεδία όπως τα προβλήματα συνδυαστικής βελτιστοποίησης. Η εφαρμογή τους σε προβλήματα προγραμματισμού καταστημάτων ροής έχει διατυπωθεί σε πολλές δημοσιεύσεις με σημαντικότερη αυτή του Reeves, ο οποίος σύγκρινε την απόδοση των SA τεχνικών με των GA για προβλήματα καταστημάτων ροής που είχαν εύρος από 20 εργασίες και 5 μηχανές μέχρι 500 εργασίες και 20 μηχανές. Το βασικό συμπέρασμα ήταν πως οι αλγόριθμοι προσομοιωμένης ανόπτησης έδιναν καλύτερα αποτελέσματα στις περιπτώσεις που υπήρχαν μέχρι και 50 εργασίες. Από την άλλη πλευρά οι γενετικοί έδιναν καλύτερες λύσεις σε προβλήματα με μεγάλες τιμές εργασιών από ότι οι προσομοιωμένης ανόπτησης αλγόριθμοι.

Οι γενετικοί αλγόριθμοι, για την επίλυση του προβλήματος προγραμματισμού καταστημάτων εργασιών καθορισμένης ροής, συνήθως χρησιμοποιούν αναπαράσταση με ακεραίους, στην οποία τα χρωμοσώματα είναι οι φορείς των δεικτών των εργασιών και αναπαριστούν την ακολουθία των εργασιών. Αυτό σημαίνει ότι αν μία εργασία είναι στην θέση i του χρωμοσώματος τότε είναι στην ioστή θέση της ακολουθίας.

Ο χώρος της έρευνας καθορίζεται από τον αριθμό των υπαρχόντων χρωμοσωμάτων, που είναι σταθερός και έστω N. Το N αναφέρεται ως το μέγεθος του πληθυσμού. Ο αρχικός πληθυσμός μπορεί να αποτελείται από N τυχαίες ακολουθίες ή

μπορεί να αποκτηθεί από έναν κατασκευαστικό ευρετικό αλγόριθμο. Η συνάρτηση καταλληλότητας ενός χρωμοσώματος, αντικατοπτρίζει την αποτελεσματικότητα του χρόνου περάτωσης που ανταποκρίνεται στην ακολουθία των εργασιών που αφορά. Η καταλληλότητα που αποδίδεται σε ένα αλφαριθμητικό i μπορεί να είναι ανάλογη στην τιμή $f_i = \frac{r_i}{\sum_j r_j}$, όπου r_i είναι ανάλογο του χρόνου περάτωσης του χρωμοσώματος i στον πληθυσμό. Με τυχαία επιλογή χρωμοσωμάτων από τον πληθυσμό, που συνήθως είναι οι γονείς, και αφού τα ζευγαρώσουμε, γεννιέται νέο χρωμόσωμα που κληρονομεί τα χαρακτηριστικά των γονέων του. Το μέγεθος του πληθυσμού παραμένει σταθερό με την απομάκρυνση ενός χρωμοσώματος, αφού κάθε φορά που γεννιέται ένα εισάγεται στον πληθυσμό. Το απομακρυσμένο χρωμόσωμα μπορεί να επιλεχτεί τυχαία ή με βάση την καταλληλότητα που έχει. Αυτό προσομοιώνει ένα πραγματικό σενάριο, όπου χρωμοσώματα που παρουσιάζουν λύσεις με καλό χρόνο περάτωσης έχουν μεγαλύτερη πιθανότητα να παραμείνουν στον πληθυσμό και να επιλεχθούν για περαιτέρω ζευγάρωμα. Το ζευγάρωμα γίνεται με διασταύρωση και τελεστές μετάλλαξης που διαμορφώνουν την επόμενη γενιά. Ο βασικός σκοπός της διασταύρωσης είναι η ανταλλαγή πληροφοριών μεταξύ των τυχαία επιλεγμένων χρωμοσωμάτων των γονέων, με στόχο την παραγωγή καλύτερων απογόνων και την εύρεση καλύτερων γονιδίων. Η δυσκολία που υπάρχει στην εφαρμογή των χειριστών διασταύρωσης σε χρωμοσώματα που είναι μη δυαδικά, όπως συμβαίνει στην ακέραιη αναπαράσταση του προβλήματος καταστημάτων ροής, είναι ότι η αποκωδικοποίηση καταλήγει σε μη υπαρκτά αλφαριθμητικά στα οποία οι εργασίες εμφανίζονται δύο ή και καμία φορά στον απόγονο.

Παρόλα αυτά, πολύ τελεστές διασταύρωσης έχουν προταθεί για χρήση σε προβλήματα καταστημάτων ροής όπως: 2 point crossover (2X), partially mapped crossover (MPX), linear order crossover (LOX), cycle crossover (CX), C1 τελεστής, NABEL τελεστής, multi-parent crossover (MPX) και longest common subsequence crossover (LCSX).

Επιπρόσθετα, εφαρμόζονται και τελεστές μετάλλαξης στους γενετικούς αλγορίθμους. Χρησιμοποιούνται για να διευρύνουν τον χώρο έρευνας και δρουν για να αποφευχθεί η επιλογή και διασταύρωση από μια μόνο συγκεκριμένη περιοχή του χώρου

αυτού. Ακόμη, χρησιμοποιούνται για να αποφευχθεί κόλλημα του αλγορίθμου σε ένα τοπικό βέλτιστο.

Τέλος, παραθέτουμε μια γενική μορφή του γενετικού αλγορίθμου, όπως αναφέρετε στο άρθρο των S.REZA HEJAKI and SAGHAFIAAN, "Flowshop-scheduling with makespan criterion: a review".

Έναρξη:
προσδιορισμός του πληθυσμού, n;
γενιά=0;
γέννησε αρχικό πληθυσμό και ανέθεσε τον σε ένα παλιό πληθυσμό, old_pop;
υπολόγισε την καταλληλότητα κάθε αριθμού του παλιού πληθυσμού΄
επανέλαβε;
επανέλαβε;
στοχαστικά διάλεξε 2 γονείς με πιθανότητα ανάλογη της καταλληλότητας τους;
τυχαία διάλεξε ένα σημείο διασταύρωσης και ξεκίνησε την διασταύρωση των γονέων με πιθανότητα Pc να δημιουργηθούν 2 απόγονοι;
μετάλλαξε κάθε bit κάθε απογόνου με πιθανότητα Pm;
τοποθέτησε τους 2 απόγονους σε ένα νέο πληθυσμό, new_pop;
μέχρι ο νέος πληθυσμός να είναι γεμάτος με n διαφορετικούς απογόνους;
γενιά=γενιά+1;
old pop=new_pop;
υπολόγισε την καταλληλότητα κάθε αριθμού στον παλιό πληθυσμό;
μέχρι κάποια συνθήκη τερματισμού να ισχύσει;

3.3.3 Περιορισμένη αναζήτηση (Tabu Search)

Ο μεθευρετικός αλγόριθμος της περιορισμένης αναζήτησης παρουσιάστηκε αρχικά από τον Glover. Αυτή η μέθοδος χρησιμοποιεί έναν ευρετικό αλγόριθμο για να κινηθεί από την μία λύση στην άλλη. Μια σύντομη περιγραφή του αλγορίθμου αυτού είναι η εξής: Αρχικά έχουμε μια αρχική λύση και εφαρμόζουμε ένα μηχανισμό μετακίνησης για να ερευνήσουμε την γειτονιά της τρέχουσας λύσης και να επιλέξουμε

την πιο κατάλληλη. Για να αποφύγουμε το κόλλημα σε τοπικά ελάχιστα κρατάμε ιστορικές πληροφορίες από τις k τελευταίες επαναλήψεις και το σύνολο των λύσεων που καθορίζονται από αυτές τις πληροφορίες αποτελούν την λίστα περιορισμένης αναζήτησης (tabu list). Η λίστα περιέχει l στοιχείο κάθε φορά και όποτε ένα εισέρχεται, το πιο παλιό διαγράφεται. Αυτή η παράμετρος ονομάζεται μέγεθος λίστας. Για τον καθορισμό της απαιτείται: να ορισθεί μια συνεχής τιμή, να επιλεχτεί τυχαία και να αλλάζει δυναμικά μέσω κατάλληλων προσαρμογών. Παρόλο την απλότητά του, είναι τέχνη ο προσδιορισμός της γειτονιάς, το ψάξιμο ανάμεσα στις γειτονιές, ο καθορισμός της λίστας περιορισμένης αναζήτησης, κτλ. Η κατασκευή αυτών των παραμέτρων επηρεάζει την απόδοση του αλγορίθμου, την ταχύτητα σύγκλισης και τον χρόνο εκτέλεσης.

Έχουν προταθεί διάφορες προσεγγίσεις βασιζόμενες στον αλγόριθμο περιορισμένης αναζήτησης για την επίλυση του προβλήματος καταστημάτων εργασιών καθορισμένης ροής από διάφορους ερευνητές. Αναφέρουμε τους Taillard (1990), Reeves (1993), Mocellin (1995), Nowicki και Smutnicki (1996), Ben-Daya (1998), Mocellin and dos Santos (2000) και γνωστός SPIRIT αλγόριθμος που προτάθηκε από τους Widmer και Hertz (1989).

3.3.4 Αλγόριθμος των Μυρμηγκιών (Ant colony)

Ο αλγόριθμος των μυρμηγκιών προτάθηκε αρχικά από τους Dorigo και Gambardella (1977) και είναι από τους πιο πρόσφατους και ελπιδοφόρους μεθευρετικούς αλγόριθμους για την επίλυση συνδυαστικών προβλημάτων. Ο αλγόριθμος αυτός προσομοιώνει τον τρόπο με τον οποίο τα μυρμήγκια ψάχνουν, συλλέγουν τροφή και επιστρέφουν στην φωλιά τους. Τα μυρμήγκια είναι ικανά να βρίσκουν την φωλιά τους και να επιστρέφουν από την εξόρμηση για φαγητό με την βοήθεια μιας αρωματικής ουσίας, της φερεμόνης. Μεγαλύτερο ποσοστό φερεμόνης σε ένα μονοπάτι, του δίνει μεγαλύτερη πιθανότητα να το ακολουθήσουν τα μυρμήγκια και με αυτό τον τρόπο επιστρέφουν στην φωλιά τους επιλέγοντας τον συντομότερο δρόμο.

Αυτή η συμπεριφορά των μυρμηγκιών μπορεί να χρησιμοποιηθεί για την επίλυση συνδυαστικών προβλημάτων μέσω της προσομοίωσης: η αντικειμενική συνάρτηση θα ανταποκρίνεται στην ποιότητα του φαγητού, τα τεχνητά μυρμήγκια που ψάχνουν το

χώρο της λύσης προσομοιώνουν τα μυρμήγκια στην φύση που ψάχνουν στο περιβάλλον και μία προσαρμοστική μνήμη ανταποκρίνεται στο μονοπάτι φερεμόνης. Ακόμη τα τεχνητά μυρμήγκια είναι εξοπλισμένα με μια τοπική ευρετική συνάρτηση για να καθοδηγούν την έρευνα τους στα σετ των εφικτών λύσεων.

Παρόλο που αυτός ο αλγόριθμος έχει χρησιμοποιηθεί στην επίλυση πολλών προβλημάτων, υπάρχουν λίγες έρευνες που να αφορούν το πρόβλημα καταστημάτων ροής. Οι T'kindt et al (2003) πρότειναν ένα ACO αλγόριθμο για την επίλυση του πολυκριτήριου προβλήματος καταστημάτων ροής με δύο μηχανές, με στόχο την ελαχιστοποίηση του χρόνου περάτωσης. Οι Rajendran και Ziegler (2004) θεώρησαν το πρόβλημα του προγραμματισμού εργασιών με ελεύθερη διάταξη στις μηχανές και με αντικειμενική συνάρτηση την ελαχιστοποίηση του χρόνου περάτωσης, λαμβάνοντας υπόψη την ελαχιστοποίηση του ολικού χρόνου ροής των εργασιών. Οι πρώτες δημοσιεύσεις στις οποίες γινόταν εφαρμογή του αλγορίθμου των μυρμηγκιών στο $N/M/F/c_{max}$ πρόβλημα έγιναν από τους Stützle (1998) και Ying and Liao (2003). Τα τελευταία υπολογιστικά πειράματα έδειξαν ότι αυτός ο μεταευρετικός αλγόριθμος είναι ιδιαίτερα αποτελεσματικός για αυτό το πρόβλημα.

3.4 Μικτός ακέραιος γραμμικός προγραμματισμός

Όπως ήδη έχουμε αναφέρει, σκοπός αυτής της εργασίας είναι να συγκρίνει την απόδοση δύο οικογενειών μοντέλων μικτού ακέραιου γραμμικού προγραμματισμού, για την επίλυση του προβλήματος προγραμματισμού καταστημάτων εργασιών με κοινή διάταξη, με αντικειμενική συνάρτηση την ελαχιστοποίηση του χρόνου περάτωσης των εργασιών. Σε αυτό το σημείο θα παρουσιάσουμε τις βασικές αρχές του μικτού ακέραιου γραμμικού προγραμματισμού.

Ένας μεγάλος αριθμός μοντέλων βελτιστοποίησης έχει συνεχής και ακέραιες μεταβλητές οι οποίες εμφανίζονται γραμμικά στην αντικειμενική συνάρτηση και στους περιορισμούς. Αυτά τα μαθηματικά μοντέλα ονομάζονται MILP (Mixed-Integer Linear Programming) προβλήματα.

Μεγάλο εύρος εφαρμογών μπορούν να μοντελοποιηθούν σαν ένα MILP πρόβλημα και γι'αυτό το λόγο έχουν τραβήξει την προσοχή πολλών ερευνητών στο πεδίο

της Επιχειρησιακής Έρευνας, όπου συμπεριλαμβάνονται προβλήματα χωροταξικής διάταξης, προβλήματα προγραμματισμού και προβλήματα δικτύων σταθερού φορτίου. Επίσης, έχει διαπιστωθεί ότι εφαρμογή αυτών των μοντέλων σε προβλήματα Χημικής Μηχανικής έχει καλά αποτελέσματα. Οι προτεινόμενοι αλγόριθμοι για MILP προβλήματα μπορούν να κατηγοριοποιηθούν ως εξής:

i. Αλγόριθμος διακλάδωσης και φράγματος (Branch and Bound)
ii. Μέθοδος αποσύνθεσης
iii. Μέθοδος λογικής βάσης
iv. Μέθοδος cutting plane

3.4.1 Αλγόριθμος Διακλάδωσης και Φράγματος

Ο αλγόριθμος Διακλάδωσης και Φράγματος είναι μία αποτελεσματική μέθοδος επίλυσης συνδυαστικών προβλημάτων. Όπως μπορεί κανείς να καταλάβει από το όνομα του αλγορίθμου, αποτελείται από δύο ουσιαστικές διαδικασίες, την διαδικασία διακλάδωσης, όπου ένα μεγάλο πρόβλημα χωρίζεται σε 2 ή περισσότερα υποπροβλήματα και την διαδικασία του Φράγματος, όπου υπολογίζεται ένα κάτω όριο για την βέλτιστη λύση ενός υποπροβλήματος.

Με την διαδικασία διακλάδωσης αντικαθίσταται το αρχικό πρόβλημα από μια ομάδα νέων προβλημάτων τα οποία είναι

(α) αμοιβαίως αποκλειόμενα υποπροβλήματα του αρχικού

(β) μερικώς λυμένες εκδοχές του αρχικού προβλήματος

(γ) μικρότερα προβλήματα από το αρχικό

Επίσης τα υποπροβλήματα μπορούν να διαιρεθούν σε μικρότερα, με τον ίδιο τρόπο. Έστω P^o δηλώνει ένα πρόβλημα ακολουθίας μιας μηχανής που περιέχει n εργασίες. Αυτό το πρόβλημα μπορεί να διαιρεθεί σε n υποπροβλήματα $P_1^1, P_2^1, ..., P_n^1$ προσδιορίζοντας την τελευταία θέση στην ακολουθία. Έτσι το P_1^1 είναι το ίδιο πρόβλημα με την εργασία 1 στην τελευταία θέση της ακολουθίας και όμοια και για τα υπόλοιπα. Είναι φανερό πως αυτά τα προβλήματα είναι μικρότερα του αρχικού, αφού μένουν (n-1) θέσεις να καθοριστούν και κάθε ένα πρόβλημα είναι μια μερική λύση του αρχικού. Ακόμη, εάν

λυθεί κάθε υποπρόβλημα τότε η καλύτερη από τις λύσεις θα αντιπροσωπεύει την βέλτιστη λύση του αρχικού.

Κάθε υποπρόβλημα μπορεί να διαιρεθεί στο επόμενο επίπεδο σε υποπροβλήματα και θα ισχύουν, και σε αυτή την περίπτωση, τα ίδια ακριβώς. Στο επίπεδο k κάθε υποπρόβλημα περιέχει k ορισμένες θέσεις στην ακολουθία και μπορεί να διαιρεθεί σε (n-k) υποπροβλήματα. Αν συνεχιστεί αυτή η διαδικασία μέχρι τέλος, θα υπάρχουν n! υποπροβλήματα στο n επίπεδο, με μία εφικτή λύση για το αρχικό πρόβλημα, που σημαίνει ότι θα έπρεπε να εξετάσουμε όλες τις πιθανές ακολουθίες. Για την αποφυγή αυτού ακριβώς, χρησιμοποιείται η διαδικασία του Φράγματος.

Η διαδικασία Φράγματος υπολογίζει ένα κάτω όριο για κάθε λύση των υποπροβλημάτων. Υποθέστε ότι στο ίδιο επίπεδο βρέθηκε μία λύση που έχει μέτρο απόδοσης ίσο με Z. Ακόμη υποθέστε ότι, ένα υποπρόβλημα που δημιουργήθηκε κατά την διαδικασία διακλάδωσης έχει σχετικό κάτω όριο ίσο με b>Z. Σε αυτή την περίπτωση το υποπρόβλημα αυτό παύει να θεωρείται υποψήφιο για την εύρεση της βέλτιστης λύσης γιατί έχει ήδη μεγαλύτερο κόστος από το Z. Τότε το πρόβλημα διαγράφεται και σταματά η διακλάδωση του από κει και πέρα. Με αυτό τον τρόπο μειώνονται οι πιθανές βέλτιστες λύσεις και καταλήγουμε στο βέλτιστο σε λιγότερες επαναλήψεις. Η λύση που χρησιμοποιείται για να συγκρίνουμε τα κλαδιά του δένδρου και να καταλάβουμε ποιος κλάδος πρέπει να διαγραφεί λέγεται δόκιμη λύση (trial solution).

Στην συνέχεια θα χρησιμοποιήσουμε το πρόβλημα της ελαχιστοποίησης της σταθμισμένης καθυστέρησης για να δούμε πως εφαρμόζεται αυτός ο αλγόριθμος. Δηλώνουμε σ μια μερική ακολουθία εργασιών ανάμεσα στις n εργασίες του προβλήματος, iσ η μερική ακολουθία στην οποία η σ ακολουθείται άμεσα από την εργασία i. Μπορούμε να θεωρήσουμε την σ μια ομάδα διατεταγμένων εργασιών έτσι ώστε πριν από αυτή να είναι

σ' = the complement of σ

$$q_\sigma = \sum_{j \in \sigma'} t_j$$

Έστω P_σ^k αναπαριστά ένα υποπρόβλημα στο επίπεδο k του δένδρου διακλάδωσης. Αυτό το πρόβλημα θα είναι το αρχικό, με τις k τελευταίες θέσεις στην ακολουθία καθορισμένες, όπου το σ δηλώνει την μερική καθορισμένη ακολουθία. Σχετική με το P_σ^k

είναι μια αξία, η u_σ, η οποία είναι η συνεισφορά των καθορισμένων εργασιών στην ολική καθυστέρηση και δίνεται από την σχέση $u_\sigma = \sum_{j\in\sigma} w_j T_j$.

Οι χρόνοι Tj στο παραπάνω άθροισμα μπορούν να υπολογιστούν, γιατί ο χρόνος συμπλήρωσης για κάθε j είναι γνωστός παρόλο που δεν έχει καθοριστεί η τελική ακολουθία.

Κανονικά η διακλάδωση θα χωρίσει το P_σ^k πρόβλημα σε (n-k) υποπροβλήματα. Κάθε ένα από αυτά κατασκευάζεται επιλέγοντας μια εργασία $i\in\sigma'$ να είναι η τελευταία στην ακολουθία που μένει. Ο χρόνος συμπλήρωσης της εργασίας i στην μερική ακολουθία iσ είναι q_σ. Για αυτό η αξία που σχετίζεται με το πρόβλημα $P_{i\sigma}^{k+1}$ είναι $u_{i\sigma} = w_i \max\{0, q_\sigma - d_i\} + u_\sigma$.

Στην διαδικασία του φράγματος ψάχνουμε για ένα κάτω όριο το οποίο μπορεί να είναι είτε $b_\sigma = u_\sigma$ είτε $b_\sigma = u_\sigma + \min(w_i \max\{0, q_\sigma - d_i\} + u_\sigma)$. Αφού το υπολογίσουμε μπορούμε να καθορίσουμε πότε η ολοκλήρωση ενός υποπροβλήματος P_σ^k μπορεί να καταλήξει σε μια βέλτιστη λύση και πότε όχι. Αν $b_\sigma < Z$, τότε μπορεί συνεχίζοντας να καταλήξουμε στην βέλτιστη λύση ενώ αν $b_\sigma \geq Z$, τότε αποκλείεται να βρούμε λύση καλύτερη από την Z.

Υπάρχουν 2 μέθοδοι για την επιλογή του προβλήματος P_σ που θα διακλαδιστεί. Η πρώτη ονομάζεται έρευνα βέλτιστου (best first search or jumptracking) και ψάχνει το βέλτιστο b_σ και η δεύτερη ονομάζεται έρευνα εις βάθος (depth first search or backtracking) και ψάχνει το καλύτερο Z στα προβλήματα με μεγαλύτερο βάθος.

3.4.2 Αλγόριθμος Διακλάδωσης και Φράγματος για το πρόβλημα προγραμματισμού εργασιών κοινής διάταξης.

Ο αλγόριθμος διακλάδωσης φράγματος, για το πρόβλημα προγραμματισμού καταστημάτων εργασιών κοινής διάταξης σε M μηχανές, με κριτήριο την ελαχιστοποίηση του χρόνου περάτωσης των εργασιών αναπτύχθηκε από τους Ignall και Schrage (1965) και ανεξάρτητα από τον Lomnicki(1965).

Το δένδρο διακλάδωσης είναι σε αυτή την περίπτωση όμοιο με αυτό που περιγράψαμε παραπάνω με μόνη διαφορά ότι, η διακλάδωση πραγματοποιείται στην πρώτη εργασία που μπαίνει στην ακολουθία, στην συνέχεια στην δεύτερη και συνεχίζει

έτσι ώστε το μερικό πρόγραμμα των εργασιών να αναπαριστά τις k πρώτες εργασίες που καθορίζονται και όχι τις τελευταίες. Για την εύρεση καλού κάτω φράγματος του κριτηρίου, για κάθε κόμβο, αρκεί να πάρουμε το μέγιστο καλό κάτω φράγμα του χρόνου περάτωσης που βρίσκουμε για κάθε μηχανή.

3.4.3 Περίπτωση όπου M=3

Υποθέτουμε ότι σ' είναι η ομάδα των εργασιών που δεν περιλαμβάνονται στην μερική ακολουθία σ. Έστω

$q_1 =$ ο τελευταίος χρόνος ολοκλήρωσης στην μηχανή 1 ανάμεσα στις εργασίες στην σ

$q_2 =$ ο τελευταίος χρόνος ολοκλήρωσης στην μηχανή 2 ανάμεσα στις εργασίες στην σ

$q_3 =$ ο τελευταίος χρόνος ολοκλήρωσης στην μηχανή 3 ανάμεσα στις εργασίες στην σ

το μέγεθος της διαδικασίας που απαιτείται στην μηχανή 1 είναι $\sum_{j \in \sigma'} t_{j1}$

και το κάτω όριο για τον χρόνο περάτωσης είναι $b_1 = q_1 + \sum_{j \in \sigma'} t_{j1} + \min_{j \in \sigma'} \{t_{j2} + t_{j3}\}$.

Ο χρόνος $\{t_{j2} + t_{j3}\}$ είναι ο χρόνος που πρέπει να περάσει, από την στιγμή που η τελευταία εργασία k θα τελειώσει την επεξεργασία της στην μηχανή 1 μέχρι ολόκληρο το πρόγραμμα να ολοκληρωθεί. Με όμοιο τρόπο ισχύει για τις άλλες 2 μηχανές :

$$b_2 = q_2 + \sum_{j \in \sigma'} t_{j2} + \min_{j \in \sigma'} \{t_{j3}\}$$

$$b_3 = q_3 + \sum_{j \in \sigma'} t_{j3} .$$

Εφαρμόζοντας τον αλγόριθμο των Ignall και Schrage το κάτω φράγμα θα είναι

$$B = \max \{b_1, b_2, b_3\}$$

Ο αλγόριθμος αυτός ακολουθεί την μέθοδο «έρευνα του βέλτιστου» σε συνδυασμό με την επόμενη πρόταση, η οποία επιτρέπει σε κάποια κλαδιά να περιοριστούν.

Πρόταση: Υποθέτουμε τις μερικές ακολουθίες $\sigma^{(1)}$ και $\sigma^{(2)}$ που περιέχουν τις ίδιες εργασίες σε διαφορετική σειρά. Αν $q_k^{(1)} \leq q_k^{(2)}$ για κάθε μηχανή k, τότε $\sigma^{(1)}$ κυριαρχεί της $\sigma^{(2)}$ και έτσι η $\sigma^{(2)}$ δεν λαμβάνεται από κει και πέρα υπόψη για την εύρεση βέλτιστης λύσης.

3.4.4. Αριθμητικό παράδειγμα

Για να εφαρμόσουμε τα παραπάνω, θα χρησιμοποιήσουμε το πρόβλημα με 4 εργασίες και 3 μηχανές που περιγράφεται στον παρακάτω πίνακα.

job j	1	2	3	4
t_{j1}	3	11	7	10
t_{j2}	4	1	9	12
t_{j3}	10	5	13	2

Έστω ότι ο πρώτος κόμβος γεννήθηκε με τον αλγόριθμο διακλάδωσης και φράγματος και ανταποκρίνεται στο πρόβλημα P_1^1, για το οποίο η εργασία 1 έχει τοποθετηθεί στην πρώτη θέση της ακολουθίας και $\sigma' = \{2,3,4\}$. Για αυτή την μερική ακολουθία $q_1 = t_{11} = 3$, $q_2 = t_{11} + t_{12} = 3 + 4 = 7$ και $q_3 = t_{11} + t_{12} + t_{13} = 3 + 4 + 10 = 17$.

Ο υπολογισμός του κάτω φράγματος φαίνεται στην συνέχεια:

$b_1 = 3 + 28 + 6 = 37$

$b_2 = 7 + 22 + 2 = 31$

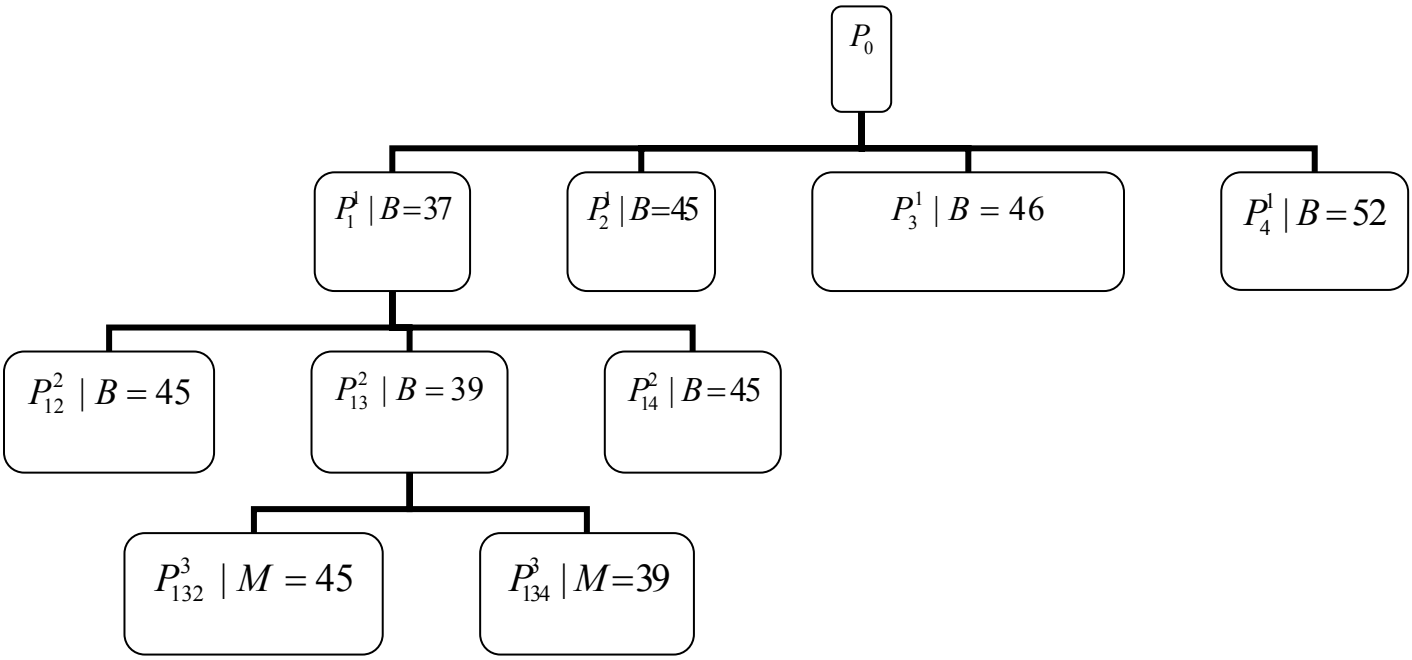
$b_3 = 17 + 20 + 0 = 37$

$B = \max\{37,31,37\} = 37$

Οι υπολογισμοί που απαιτούνται για την απόκτηση της βέλτιστης λύσης φαίνονται στο επόμενο πίνακα:

Partial Sequence	(q_1,q_2,q_3)	(b_1,b_2,b_3)	B
1	(3,7,17)	(37,31,37)	37
2	(11,12,17)	(45,39,42)	45
3	(7,16,29)	(37,35,46)	46
4	(10,22,24)	(37,41,52)	52
12	(14,15,22)	(45,38,37)	45
13	(10,19,32)	(37,34,39)	39
14	(13,25,27)	(37,40,45)	45
132	(21,22,37)	(45,36,39)	45
134	(20,32,34)	(37,38,39)	39

και στην συνέχεια φαίνεται το σχετικό δένδρο διακλάδωσης.



4. Μοντέλα μικτού ακέραιου γραμμικού προγραμματισμού (MILP models)

Σκοπός αυτής της εργασίας είναι η παρουσίαση και η σύγκριση δύο οικογενειών μοντέλων μικτού ακέραιου γραμμικού προγραμματισμού για το απλό πρόβλημα προγραμματισμού καταστημάτων εργασιών κοινής διάταξης. Οι οικογένειες μοντέλων που θα εξετάσουμε έχουν πάρει το όνομα τους από τους ερευνητές που τα έχουν προτείνει και έτσι είναι Wagner και Manne οικογένειες μοντέλων. Ο Wagner χρησιμοποίησε το κλασικό πρόβλημα ανάθεσης για να ελέγξει την ανάθεση των εργασιών στις κατάλληλες θέσεις την διαδικασίας. Σε αντίθεση, ο Manne χρησιμοποιεί ζευγάρια διχοτομημένων περιορισμών και στα 5 μοντέλα της οικογένειας του ή κάποιο αλγεβρικό συνδυασμό αυτών των ζευγαριών ώστε να πραγματοποιήσει την ανάθεση των εργασιών στην ακολουθία της διαδικασίας.

Το μέτρο απόδοσης που έχουμε λάβει υπόψη είναι η ελαχιστοποίηση του χρόνου περάτωσης των εργασιών και εκφράζεται από την σχέση:

$$Makespan = F_{MAX} = \max_i \{F_i\} \qquad (1)$$

όπου Fi είναι ο ολικός χρόνος ροής της εργασίας i στο σύστημα παραγωγής. Έχει θεωρηθεί ότι όλες οι εργασίες είναι διαθέσιμες για κατεργασία την χρονική στιγμή μηδέν.

4.1 Συμβολισμοί που χρησιμοποιούμε στα μοντέλα M ILP

Πριν την περιγραφή των μαθηματικών μοντέλων θα παρουσιάσουμε τα σύμβολα που έχουν χρησιμοποιηθεί και την σημασία του καθενός. Ξεκινώντας, αναφέρουμε την χρήση των δεικτών:

r για τις μηχανές με $1 \leq r \leq M$

i και k για τις εργασίες με $1 \leq i, k \leq N$

j για την θέση στην ακολουθία με $1 \leq j \leq N$.

Ο αριθμός των μηχανών και των εργασιών παριστάνεται από τις μεταβλητές M, N αντίστοιχα. $T = \{T_{ri}\}$ είναι ο πίνακας των χρόνων επεξεργασίας με διαστάσεις M X N και T_{ri} ο χρόνος επεξεργασίας της i εργασίας στην r μηχανή. Το P συμβολίζει μία τιμή αρκετά μεγάλη ώστε να διασφαλίζει ότι μόνο ένα ζευγάρι διχοτομημένων σχέσεων είναι πραγματικό.

Οι μεταβλητές που χρησιμοποιούνται είναι :

B_{rj} χρόνος εκκίνησης της εργασίας στην θέση j στην μηχανή r

C_{ri} χρόνος συμπλήρωσης της εργασίας i στην μηχανή r

D_{ik} 1, αν η εργασία i είναι προγραμματισμένη οποιαδήποτε στιγμή πρίν την εργασία k. Αλλιώς είναι 0. Ισχύει $i < k$

E_{rj} χρόνος ολοκλήρωσης της εργασίας στην θέση j στην μηχανή r

S_{ri} χρόνος εκκίνησης της εργασίας i στην μηχανή r

X_{rj} νεκρός χρόνος στην μηχανή r πριν την εκκίνηση της εργασίας που βρίσκεται στην θέση j της ακολουθίας

Y_{rj} χρόνος αναμονής της εργασίας στην θέση j της ακολουθίας μετά την ολοκλήρωση της επεξεργασίας της στην μηχανή r

Z_{ij} 1, αν η εργασία i είναι τοποθετημένη στην θέση j της ακολουθίας και 0 αλλιώς.

Οι μεταβλητές D_{ik} και Z_{ij} είναι δυαδικοί ακέραιοι και όλες οι άλλες είναι πραγματικές μεταβλητές που παίρνουν ακέραιες τιμές όταν οι χρόνοι επεξεργασίας έχουν ακέραιες τιμές.

4.2 Η οικογένεια μοντέλων MILP Wagner

Η οικογένεια αυτή αποτελείται από τρία μοντέλα, τα WST, Wilson και TS2. Κάθε ένα από αυτά χρησιμοποιεί την εξής αναπαράσταση για το πρόβλημα ανάθεσης των εργασιών στις θέσεις της ακολουθίας.

$$\sum_{j=1}^{N} Z_{ij} = 1 \quad (1 \le i \le N) \qquad (2)$$

$$\sum_{i=1}^{N} Z_{ij} = 1 \quad (1 \le j \le N) \qquad (3)$$

Η σχέση (1) διασφαλίζει ότι κάθε εργασία τοποθετείται σε μία και μόνο μία θέση της ακολουθίας και η σχέση (2) ότι κάθε θέση έχει γεμίσει από μια και μόνο μια εργασία. Στην συνέχεια θα παρουσιάσουμε αναλυτικά κάθε μοντέλο αυτής της οικογένειας.

4.2.1 Το μοντέλο WST

Η αντικειμενική συνάρτηση ελαχιστοποίησης του χρόνου περάτωσης μπορεί να εκφραστεί ως εξής: $F_{max}=C_{max}=C_{MN}=\sum_{i=1}^{N} T_{Mi} + \sum_{p=1}^{N} X_{Mp}$ (O1)

Οι περιορισμοί του μοντέλου αυτού είναι οι (1) , (2) και οι εξής

$$\sum_{i=1}^{N} T_{ri}Z_{i,\,j+1} + X_{r,\,j+1} + Y_{r,\,j+1} = \sum_{i=1}^{N} T_{r+1,\,i}Z_{ij} + X_{r+1,\,j+1} + Y_{rj}$$
$$(1 \le r \le M-1; 1 \le j \le N-1) \qquad (4)$$

$$X_{r+1,1} = X_{r,1} + Y_{r,1} + \sum_{i=1}^{N} T_{ri}Z_{i1} \quad (1 \le r \le M-1) \qquad (5)$$

$$Y_{r1} = 0 \quad (1 \le r \le M-1) \qquad (6)$$

Οι περιορισμοί (3) και (4) διαβεβαιώνουν ότι (α) η εργασία στην θέση j+1 δεν μπορεί να ξεκινήσει την επεξεργασία της στην μηχανή r μέχρι η εργασία που βρίσκεται στην θέση j να έχει συμπληρώσει την επεξεργασία της στην ίδια μηχανή και (β)ότι η εργασία στην θέση j δεν μπορεί να ξεκινήσει την επεξεργασία της στην μηχανή r+1 αν δεν έχει τελειώσει από την μηχανή r. Ο περιορισμός (5) αναγκάζει την εργασία που βρίσκεται στην πρώτη θέση της ακολουθίας να ξεκινήσει αμέσως την επεξεργασία της σε μία από τις μηχανές 2 έως M αφού πρώτα έχει τελειώσει με την προηγούμενη.

4.2.2 Το μοντέλο Wilson

Το μοντέλο αυτό προτάθηκε πρώτη φορά το 1989 από τον Wilson και στο οποίο γίνεται χρήση των χρόνων εκκίνησης των εργασιών στις μηχανές για την δημιουργία των περιορισμών. Η αντικειμενική συνάρτηση και οι περιορισμοί παρατίθενται παρακάτω:

$$F_{max} = B_{MN} + \sum_{i=1}^{N} T_{Mi}Z_{iN} \qquad (O2)$$

$$B_{1j} + \sum_{i=1}^{N} T_{1i} Z_{ij} = B_{1,\,j+1} \quad (1 \le j \le N-1) \qquad (7)$$

$$B_{11} = 0 \qquad (8)$$

$$B_{r1} + \sum_{i=1}^{N} T_{ri} Z_{i1} = B_{r+1,1} \quad (1 \le \mathrm{r} \le \mathrm{M}\text{-}1) \qquad (9)$$

$$B_{rj} + \sum_{i=1}^{N} T_{ri} Z_{ij} \le B_{r+1,\,j} \quad (1 \le \mathrm{r} \le \mathrm{M}\text{-}1; 2 \le \mathrm{j} \le \mathrm{N}) \qquad (10)$$

$$B_{rj} + \sum_{i=1}^{N} T_{ri} Z_{ij} \le B_{r,\,j+1} \quad (2 \le \mathrm{r} \le \mathrm{M}; 1 \le \mathrm{j} \le \mathrm{N}\text{-}1) \qquad (11)$$

Ο περιορισμός (7) αναγκάζει όλες τις εργασίες να ξεκινήσουν την επεξεργασία τους στην μηχανή 1 αμέσως μετά την αναχώρηση της προηγούμενης στη σειρά εργασιάς από αυτή την μηχανή. Ο (8) αναγκάζει την πρώτη εργασία στην ακολουθία, να ξεκινήσει την επεξεργασία στην μηχανή 1 την χρονική στιγμή 0. Ο περιορισμός (9) αναγκάζει την πρώτη στην ακολουθία εργασία να περάσει στην επόμενη μηχανή αμέσως μετά την ολοκλήρωση της επεξεργασίας στην τρέχουσα, δηλαδή να μην υπάρχει καμία καθυστέρηση. Ο (10) διασφαλίζει πως, για όλες τις μηχανές, κάθε εργασία δεν μπορεί να μεταβεί στην επόμενη μηχανή αν δεν έχει ολοκληρώσει την επεξεργασία της στην προηγούμενη. Τέλος, ο ενδέκατος περιορισμός διασφαλίζει πως, για όλες τις μηχανές, μια εργασία δεν μπορεί να ξεκινήσει επεξεργασία σε μία μηχανή αν η προηγούμενη της στην ακολουθία δεν έχει τελειώσει από αυτή την μηχανή.

4.2.3 Μοντέλο TS2

Το μοντέλο TS2 MILP για την επίλυση του προβλήματος προγραμματισμού εργασιών ελεύθερης διάταξης σε κάθε μηχανή αναφέρθηκε πρώτη φορά από τους EF Stafford, FT Tseng και JND Gupta[4] . Σε αυτό το μοντέλο γίνεται χρήση των χρόνων ολοκλήρωση ή τελικούς χρόνους E_{rj} . Οι περιορισμοί καθώς και η αντικειμενική συνάρτηση του μοντέλου είναι:

$$F_{\max} = E_{MN} \qquad (O3)$$

$$E_{rj} + \sum_{i=1}^{N} T_{ri} Z_{i,\,j+1} \le E_{r,\,j+1} \quad (1 \le \mathrm{r} \le \mathrm{M}; 1 \le \mathrm{j} \le \mathrm{N}\text{-}1) \qquad (12)$$

$$E_{rj} + \sum_{i=1}^{N} T_{r+1,i} Z_{ij} \le E_{r+1,j} \quad (1 \le r \le M\text{-}1; 1 \le j \le N) \qquad (13)$$

$$E_{11} \ge \sum_{i=1}^{N} T_{1i} Z_{i1} \qquad (14)$$

Με την χρήση του περιορισμού (12) διασφαλίζεται ότι η εργασία στην θέση j+1 της ακολουθίας δεν μπορεί να τελειώσει σε οποιαδήποτε μηχανή αν η προηγούμενη της στην ακολουθία δεν έχει τελειώσει από αυτή την μηχανή και η εργασία στην θέση j+1 έχει επίσης περάσει από αυτή την μηχανή. Ο επόμενος περιορισμός εξασφαλίζει ότι μια εργασία δεν μπορεί να ολοκληρωθεί στην μηχανή r+1 αν δεν έχει τουλάχιστον τελειώσει από την προηγούμενη και έχει πλήρους περάσει στην r+1. Ο 14ος περιορισμός υποστηρίζει ότι η εργασία που βρίσκεται στην πρώτη θέση της ακολουθίας δεν μπορεί να ολοκληρωθεί στην μηχανή 1 αν δεν έχει περάσει από αυτή. Σε αυτό το μοντέλο έχουμε μείωση κατά M-1 περιορισμούς με την χρήση του περιορισμού (13), ο οποίος δίνει την πληροφορία σχετικά με την εργασία στην πρώτη θέση της ακολουθίας στις μηχανές 2 έως M.

4.3 Οικογένεια μοντέλων MILP Manne

Υπάρχουν δύο υπό-ομάδες σε αυτή την οικογένεια μοντέλων για την επίλυση του προβλήματος που μελετάμε. Μια στην οποία γίνεται χρήση της βασικής ιδέας του Manne για τα ζευγάρια διχοτομημένων περιορισμένων και μια που χρησιμοποιεί την τροποποίηση του Liao-You (LY μοντέλα).

4.3.1 Το μοντέλο SGST(PI)

Σε αυτό το μοντέλο γίνεται χρήση της σταθεράς PI, στην οποία δίνουμε μία μεγάλη τιμή ώστε να διασφαλιστεί ότι μόνο ένα ζευγάρι διχοτομημένης σχέσης υπάρχει για κάθε ζεύγος περιορισμών. Η αντικειμενική συνάρτηση είναι :

$$F_{\max} = C_{\max} \qquad (O4)$$

υπό τους περιορισμούς

$$C_{1i} \ge T_{1i} \ (1 \le i \le N) \qquad (15)$$

$C_{r+1,i} - C_{ri} \geq T_{r+1,i}$ $(1 \leq r \leq M\text{-}1; 1 \leq i \leq N)$ (16)

$C_{ri} - C_{rk} + PD_{ik} \geq T_{ri}$ $(1 \leq r \leq M; 1 \leq i<k \leq N)$ (17)

$C_{rk} - C_{ri} + P(1 - D_{ik}) \geq T_{rk}$ $(1 \leq r \leq M; 1 \leq i<k \leq N)$ (18)

$C_{MAX} \geq C_{Mi}$ $(1 \leq i \leq N)$ (19)

Ο περιορισμός (15) διασφαλίζει ότι μία εργασία δεν μπορεί να ολοκληρώσει την επεξεργασία της στην μηχανή 1 αν δεν έχει ολοκληρωτικά περάσει σε αυτή την μηχανή. Ο 16ος περιορισμός δηλώνει ότι κάθε εργασία δεν μπορεί να ολοκληρωθεί στην r+1 μηχανή αν δεν έχει ολοκληρωθεί στην προηγούμενη και μετά περάσει πλήρως στην τρέχουσα. Αυτό σημαίνει ότι κάθε εργασία επεξεργάζεται κάθε φορά σε μία μόνο μηχανή. Ο 19ος χρησιμοποιείται για να διασφαλίσει ότι ο χρόνος περάτωσης του σετ των εργασιών είναι τουλάχιστον ίσος με τον χρόνο ολοκλήρωσης της τελευταίας εργασίας στην ακολουθία της παραγωγής.

Οι περιορισμοί (17) και (18) διαμορφώνουν τα διχοτομημένα ζευγάρια περιορισμών που σχετίζονται με τα ζευγάρια των εργασιών, i και k, i≠k, έτσι ώστε είτε η εργασία i να προηγείται της k, είτε να την ακολουθεί αλλά όχι και τα δύο.

Στο αρχικό του μοντέλο γραμμικού προγραμματισμού για το πρόβλημα καταστημάτων εργασιών ο Manne χρησιμοποίησε τους αρχικού χρόνους των εργασιών στους περιορισμούς του. Η σχέση των μοντέλων ενωρίτερου αρχικού χρόνου εργασιών με αυτά που παρουσιάζουμε εδώ εκφράζεται από την εξής σχέση:

$C_{ri} = S_{ri} + T_{ri}$ $(1 \leq r \leq M; 1 \leq i \leq N)$ (20)

4.3.2 Το μοντέλο SGST(pi)

Το μοντέλο αυτό είναι το ίδιο με το SGST(PI) με μόνη διαφορά ότι η τιμή της σταθεράς PI είναι μικρότερη και συμβολίζεται ως pi. Ο λόγος της αλλαγής της τιμής της σταθεράς είναι ότι υπάρχουν ενδείξεις ότι μπορεί να απαιτείται λιγότερος υπολογιστικός χρόνος για την επίλυση του προβλήματος.

4.3.3 Το μοντέλο LYeq(PI)

Το όνομα του μοντέλου μπορεί να μεταφραστεί σε Liao-You μοντέλο, με την χρήση ενός περιορισμού ισότητας και έναν περιορισμό μεταβλητών συνόρων που αντικαθιστά κάθε ζευγάρι διχοτομημένων περιορισμών. Τα ζεύγη των διχοτομημένων

περιορισμών είναι συνδυασμένα σε μία απλή σχέση, η οποία στην συνέχεια τίθεται ίση με μία πλεονάζουσα μεταβλητή Q_{rik}. Η συνδυασμένη εξίσωση είναι :

$$PD_{ik} + C_{ri} - C_{rk} - T_{ri} = Q_{rik} \quad (1 \le r \le M ; 1 \le i < k \le N) \qquad (21)$$

Η εξίσωση που ορίζει την μεταβλητή Q_{rik} είναι

$$Q_{ik} \le P - T_{ri} - T_{rk} \quad (1 \le r \le M; 1 \le i < k \le N) \qquad (22)$$

Η αντικειμενική συνάρτηση είναι όμοια με αυτή του SGST(PI) μοντέλου και οι περιορισμοί του (15),(16),(19) μαζί με τις 2 παραπάνω σχέσεις. Έτσι είναι

$$F_{\max} = C_{\max} \qquad (O4)$$

$$C_{1i} \ge T_{1i} \ (1 \le \text{iN}) \qquad (15)$$

$$C_{r+1,i} - C_{ri} \ge T_{r+1,i} \quad (1 \le \text{r} \le \text{M-1}; 1 \le \text{i} \le \text{N}) \qquad (16)$$

$$C_{MAX} \ge C_{Mi} \quad (1 \le \text{i} \le \text{N}) \qquad (19)$$

4.3.4 Το μοντέλο LYsub(PI)

Έχει διατυπωθεί από χειριστές λογισμικών πακέτων μαθηματικού προγραμματισμού ότι η χρήση ενσωματωμένων οριακών μεταβλητών έχει ως αποτέλεσμα μικρότερους υπολογιστικούς χρόνους από όταν γίνεται χρήση ανισοτήτων για να οριοθετηθούν αυτές οι μεταβλητές. Έτσι, αυτό το μοντέλο δημιουργήθηκε με σκοπό να εξετάσει ακριβώς αυτή την ιδιότητα και για αυτό παραμένει ίδιο με το LYeq(PI), με μόνη διαφορά την αντικατάσταση της εξίσωσης (22) από ενσωματωμένα , στο λογισμικό, άνω όρια για την μεταβλητή Q_{rik}. Επειδή το σχόλιο στην LINDO για την χρήση άνω ορίου είναι SUB το μοντέλο αυτό αναφέρεται ως LYsub μοντέλο. Η τιμή του PI παραμένει ίδια και εδώ. Το μοντέλο μπορεί να περιγραφεί ως εξής:

$$F_{\max} = C_{\max} \qquad (O4)$$

$$C_{1i} \ge T_{1i} \ (1 \le \text{iN}) \qquad (15)$$

$$C_{r+1,i} - C_{ri} \ge T_{r+1,i} \quad (1 \le \text{r} \le \text{M-1}; 1 \le \text{i} \le \text{N}) \qquad (16)$$

$$C_{MAX} \ge C_{Mi} \quad (1 \le \text{i} \le \text{N}) \qquad (19)$$

$$PD_{ik} + C_{ri} - C_{rk} - T_{ri} = Q_{rik} \quad (1 \le r \le M ; 1 \le i < k \le N) \qquad (21)$$

$$SUB(Q_{rik}) \qquad (22)$$

4.3.5 Το μοντέλο LYsub(pi)

Όπως είναι εύκολο κανείς να καταλάβει από το όνομα αυτού του μοντέλου, είναι ίδίο με το LYsub(PI) με μόνη διαφορά στην τιμή της σταθεράς P. Εδώ η τιμή του P είναι ίση με 100N. Αυτό το μοντέλο, όπως και το SGST(pi) παρουσιάστηκαν για πρώτη φορά στο άρθρο των EF Stafford, FT Tseng και JND Gupta[4] στο οποίο βασίστηκε το πειραματικό μέρος αυτής εδώ της εργασίας.

Σε αυτό το κεφάλαιο παρουσιάστηκαν 8 μοντέλα μικτού ακέραιου γραμμικού προγραμματισμού. Στην συνέχεια θα περάσουμε στην αξιολόγηση της απόδοσης τους με βάση τον υπολογιστικό χρόνο που απαιτεί το καθένα από αυτά για την επίλυση του προβλήματος προγραμματισμού καταστημάτων ροής εργασιών ελεύθερης διάταξης με κριτή

5. Πειραματικά Αποτελέσματα

Μετά την παρουσίαση των μοντέλων MILP, θα περάσουμε στην σύγκριση τους, ως προς τον υπολογιστικό χρόνο που απαιτούν για την επίλυση του προβλήματος προγραμματισμού καταστημάτων εργασιών κοινής διάταξης στις μηχανές, με αντικειμενική συνάρτηση την ελαχιστοποίηση του χρόνου περάτωσης των εργασιών. Το πειραματικό μέρος αυτής της εργασίας πραγματοποιήθηκε στο Ερευνητικό Ινστιτούτο «Institute of Industrial Engineering and Control» του Πολυτεχνείου της Καταλονίας (Technical University of Catalonia (UPC)) στην Βαρκελώνη, υπό την επίβλεψη του καθηγητή Albert Subia Coromina και της καθηγήτριας Amaia Lusa Garcia, ως μέρος ενός project του ινστιτούτου με τίτλο «Solving the P-flowshop problem with heuristics and exact algorithms».

5.1 Σύγκριση των μοντέλων MILP

Χρησιμοποιήθηκαν 9 ζεύγη (M,N), με τιμές για τις μηχανές M(5,10,20) και για τις εργασίες N(20,50,100) και 10 διαφορετικές περιπτώσεις δεδομένων για κάθε ζεύγος (M,N). Οι χρόνοι επεξεργασίας των εργασιών σε κάθε μηχανή ήταν τυχαίοι ακέραιοι που προέκυψαν από ομοιόμορφη κατανομή στο (1,100). Είναι γνωστό από παλαιότερες έρευνες ότι αυτή η κατανομή των χρόνων επεξεργασίας είναι η πιο δύσκολη για επίλυση των προβλημάτων προγραμματισμού καταστημάτων ροής, με οποιαδήποτε μέθοδο. Τα δεδομένα για την επίλυση του προβλήματος (M, N, T_{ri}, LB και UB) τα πήραμε από την ιστοσελίδα του Eric Taillard, μια βάση δεδομένων που χρησιμοποιούν οι ερευνητές για πειράματα δοκιμασίας επιδόσεων ενός λογισμικού, σε αυτή την περίπτωση του υπολογιστικού χρόνου. Η επίλυση των προβλημάτων έγινε με την χρήση του λογισμικού ILOG CPLEX 8.1 χρησιμοποιώντας το OPL studio σε ένα Dell Pentium IV 1,8 GHz προσωπικό υπολογιστή με 512MB μνήμη RDRAM.

Αρχικά, μοντελοποιήσαμε τα μαθηματικά μοντέλα που παρουσιάσαμε στο 4^{ο} Κεφάλαιο με την χρήση της γλώσσας OPL ILOG στην CPLEX έτσι ώστε να μπορέσουμε

να συνεχίσουμε στην εκτέλεση τους. Η μορφή των μοντέλων με την χρήση αυτού του λογισμικού φαίνεται στο Παράστημα Α.

Στην συνέχεια, κάναμε κάποια αρχικά δοκιμαστικά πειράματα, διαλέγοντας να εκτελέσουμε τα μοντέλα μόνο για την περίπτωση (M,N)=(5,20), εξετάζοντας 3 διαφορετικά αρχεία δεδομένων για κάθε μοντέλο, δηλαδή 24 μοντέλα. Αυτό έγινε για να πιστοποιήσουμε πως με την χρήση του λογισμικού που επιλέξαμε, καταλήγουμε στα ίδια συμπεράσματα με αυτά των EF Stafford, FT Tseng και JND Gupta[4], ότι δηλαδή τα μοντέλα της οικογένειας Wagner υπερέχουν από αυτά της οικογένειας Manne, κάτι που μπορεί κανείς να διαπιστώσει κοιτώντας τον Πίνακα 1, παρακάτω.

MODEL	M	N	INSTANCE	TIME	OBJ
wst	5	20	1	12.703	1279
wst	5	20	2	3599.916	1359
wst	5	20	3	3599.916	1081
Average				**2404.178333**	**1239.66667**
wilson	5	20	1	0.485	1164
wilson	5	20	2	1.328	1349
wilson	5	20	3	1.781	1073
Average				**1.198**	**1195.33333**
TS2	5	20	1	3599.964	1282
TS2	5	20	2	7.563	1359
TS2	5	20	3	3599.9	1083
Average				**2402.475667**	**1241.33333**
SGSTPI	5	20	1	3600.024	1297
SGSTPI	5	20	2	3600.024	1366
SGSTPI	5	20	3	3600.04	1127
Average				**3600.029333**	**1263.33333**
SGST_pi	5	20	1	3600.024	1297
SGST_pi	5	20	2	3600.087	1366
SGST_pi	5	20	3	3600.097	1130
Average				**3600.069333**	**1264.33333**
LYsubPI	5	20	1	3600.009	1297
LYsubPI	5	20	2	3600.023	1366
LYsubPI	5	20	3	3600.024	1129
Average				**3600.018667**	**1264**
LYsub_pi	5	20	1	3600.004	1304
LYsub_pi	5	20	2	3600.023	1367
LYsub_pi	5	20	3	3600.008	1118
Average				**3600.011667**	**1263**
LYeqPI	5	20	1	3599.924	1297
LYeqPI	5	20	2	3600.023	1366
LYeqPI	5	20	3	3600.024	1129
Average				**3599.990333**	**1264**

Πίνακας 1

Είναι εμφανές ότι τα μοντέλα WST, Wilson, TS2υπερέχουν των υπολοίπων και για αυτό συνεχίσαμε την έρευνα μας μόνο με αυτά.

Στα αρχεία που δημιουργήσαμε για να εκτελέσουμε τα προγράμματα, έχουμε βάλει την εντολή `wst_20_5_01.setFloatParameter("epGap", 0.001);`
με την οποία θέτουμε ανοχή στην διαδικασία επίλυσης ίση με 0.001. Αυτό σημαίνει ότι αν δεν σταματήσουμε την διαδικασία για κάποιο λόγο πριν βρεθεί η βέλτιστη λύση, τότε η διαφορά ανάμεσα στην λύση που παίρνουμε και στην βέλτιστη είναι το πολύ 0.1%. Επίσης προσθέσαμε την εντολή

```
wst_20_5_01.setFloatParameter("epAGap", 0.999);
```

με την οποία θέτουμε μια απόλυτη ανοχή για την τιμή της αντικειμενικής συνάρτησης ίση με 0.999. Απόλυτη ανοχή είναι η μέγιστη διαφορά μεταξύ της επιθυμητής λύσης της αντικειμενικής συνάρτησης και αυτής που τελικά παίρνουμε. Η αντικειμενική συνάρτηση στα μοντέλα που ελέγχουμε παίρνει ακέραιες τιμές, οπότε η ελάχιστη διαφορά που μπορεί να υπάρχει είναι ίση με 1 και γι'αυτό τον λόγο η πρόσθεση αυτής της ανοχής δεν επιφέρει αλλαγή στα αποτελέσματα αλλά την δοκιμάζουμε μήπως βοηθήσει στην επίλυση.

Αφού προσθέσαμε την απόλυτη ανοχή περνάμε στην εκτέλεση των 3 μοντέλων της οικογένειας Wagner . Λαμβάνοντας υπόψη τα αρχικά αποτελέσματα, επειδή το TS2 μοντέλο απαιτεί πολύ μεγάλο υπολογιστικό χρόνο, ακόμη και για μικρό αριθμό μηχανών και εργασιών, αποφασίσαμε να συνεχίσουμε τα πειράματα χρησιμοποιώντας 10 περιπτώσεις δεδομένων μόνο για M=5 και N=20 ή 50 ή 100. Για M=10 και N=(20,50,100) χρησιμοποιήσαμε μόνο 5 από τα 10 αρχεία δεδομένων και για M=20 και N=(20,50,100) μόνο τα 3 από τα 10 αρχεία. Τα ολοκληρωτικά αποτελέσματα για κάθε μοντέλο χωριστά φαίνονται στο Παράρτημα Β.

Εδώ δίνουμε την μέση, την ελάχιστη και την μέγιστη τιμή του υπολογιστικού χρόνου που απαιτείται για την επίλυση του προβλήματος καθώς και της τιμής της αντικειμενικής συνάρτησης.

	M=5 | N=20		M=10 | N=20		M=20 | N=20	
MODEL	**TIME**	**OBJ**	**TIME**	**OBJ**	**TIME**	**OBJ**
wilson						
Average	4.1218	1192.5	3600.013	1500.4	3600.06	2260.9
min	0.25	1073	3599.92	1353	3599.997	2159
max	20.515	1349	3600.118	1648	3600.129	2341
wst						
Average	1896.356	1223.9	3393.817	1542.8	3600.142	2296.1
min	16.687	1081	1532.938	1415	3600.072	2190
max	3599.963	1359	3601.89	1696	3600.329	2413
TS2						
Average	2108.184	1222.8	3600.089	1547.4	3600.136	2310
min	2.75	1083	3600.085	1406	3600.111	2184
max	3601.256	1359	3600.101	1702	3600.148	2392

	M=5 | N=50		M=10 | N=50		M=20 | N=50	
MODEL	**TIME**	**OBJ**	**TIME**	**OBJ**	**TIME**	**OBJ**
wilson						
Average	389.5251	2726.8	3600.225	3060.6	3600.611	4047.75
min	1.047	2548	3600.121	2916	3600.571	3862
max	3600.035	2837	3600.282	3188	3600.705	4404
wst						
Average	1267.456	2737.9	3600.31	3155.5	3600.75	4070.2
min	13.578	2552	3600.227	3026	3600.587	3927
max	3600.462	2864	3600.442	3246	3601.335	4181
TS2						
Average	554.3467	2737.6	3600.344	3115.8	3600.784	4127.333
min	3.75	2554	3600.332	3026	3600.773	4057
max	3600.156	2863	3600.352	3199	3600.789	4223

	M=5 | N=100		M=10 | N=100		M=20 | N=100	
MODEL	**TIME**	**OBJ**	**TIME**	**OBJ**	**TIME**	**OBJ**
wilson						
Average	57.4977	5209.4	3601.087	5797.3	no solution	no solution
min	10.969	4960	3601.055	5515		
max	133.092	5435	3601.134	6104		
wst						
Average	1892.317	5249.1	3601.246	5898.6	no solution	no solution
min	150.356	5018	3601.212	5518		
max	3600.697	5493	3601.275	6295		
TS2						
Average	2458.309	5252.1	3601.351	5853.2	no solution	no solution
min	186.672	5018	3601.32	5624		
max	3600.704	5493	3601.383	6057		

Παρατηρώντας τα αποτελέσματα, βλέπουμε πως και τα τρία μοντέλα απαιτούν περισσότερο από μία ώρα για να δώσουν την βέλτιστη λύση όταν αυξάνεται ο αριθμός των μηχανών πάνω από 5. Σε αυτές τις περιπτώσεις εξετάζουμε την τιμή της αντικειμενικής συνάρτησης και διαπιστώνουμε ότι το μοντέλο Wilson δίνει την καλύτερη λύση κάθε φορά. Επίσης, κανένα από τα τρία μοντέλα δεν δίνει μια εφικτή λύση για M=20 και N=100.

Όταν έχουμε ένα σύστημα παραγωγής με 5 μηχανές παρατηρούμε ότι για N=20 υπερτερεί το μοντέλο Wilson και όταν N=50 το μοντέλο TS2 εμφανίζει σημαντική βελτίωση στον υπολογιστικό χρόνο που απαιτεί σε σχέση με τις περιπτώσεις N=20 και N=100. Για N=100 η κατάσταση δεν αλλάζει και το Wilson δίνει τα καλύτερα αποτελέσματα.

Λαμβάνοντας υπόψη τα παραπάνω αποτελέσματα, περάσαμε στην εφαρμογή ενός πάνω και ενός κάτω ορίου στην αντικειμενική συνάρτηση, με βάση τις σημειώσεις του καθηγητή Pastor Moreno Rafael του Πολυτεχνείου της Καταλονίας (UPC). Για την εφαρμογή αυτών των ορίων προσθέσαμε στα πρότυπα μαθηματικά μοντέλα έναν επιπλέον περιορισμό, έναν για την εφαρμογή του πάνω ορίου και έναν για την εφαρμογή του κάτω ορίου και τα γράψαμε στο ILOG OPL. Η μορφή που πήρε κάθε ένα από αυτά φαίνεται στο παράρτημα Α.

Εφαρμόσαμε τα δύο αυτά όρια (Upper Bound, Lower Bound) στα ζεύγη (M,N)=(5,20), (M,N)=(5,50), (M,N)=(5,100) που έδωσαν ικανοποιητικά αποτελέσματα, για να εξετάσουμε το ενδεχόμενο βελτιστοποίησης του υπολογιστικού χρόνου. Σε κάθε ζευγάρι χρησιμοποιήσαμε και τα 10 διαφορετικά αρχεία δεδομένων. Τα αποτελέσματα για όλες τις περιπτώσεις φαίνονται στο Παράρτημα Β. Εδώ δίνουμε μόνο τα συγκεντρωτικά αποτελέσματα της μέσης τιμής, της μέγιστης και της ελάχιστης τιμής του υπολογιστικού χρόνου και της αντικειμενικής συνάρτησης.

Upper Bound						
	M=5 | N=20		M=5 | N=50		M=5 | N=100	
MODEL	**TIME**	**OBJ**	**TIME**	**OBJ**	**TIME**	**OBJ**
wilson_UB						
Average	**361.0675**	**1192.5**	**20.0675**	**2726.5**	**90.3053**	**5208.8**
min	0.312	1073	1.14	2548	12.359	4956
max	3599.94	1349	79.83	2837	435.951	5435
wst_UB						
Average	**1385.813**	**1222.9**	**1322.247**	**2732**	**1455.453**	**5249**
min	25.361	1081	11.657	2552	141.351	5018
max	3600.325	1359	3600.481	2863	3600.919	5493
TS2_UB						
Average	**2153.63**	**1222**	**no solution**		**no solution**	
min	2.344	1081				
max	3600.688	1359				

Lower Bound						
	M=5 | N=20		M=5 | N=50		M=5 | N=100	
MODEL	**TIME**	**OBJ**	**TIME**	**OBJ**	**TIME**	**OBJ**
wilson_LB						
Average	**17.4224**	**1204.3**	**113.3535**	**2727.2**	**56.2255**	**5218.6**
min	0.109	1073	0.859	2548	12.015	4963
max	62.231	1349	925.101	2838	142.684	5437
wst_LB						
Average	**1156.809**	**1223.6**	**1362.048**	**2737.5**	**1956.338**	**5248.7**
min	10.922	1091	14.625	2552	168.673	5018
max	3600.604	1359	3600.195	2863	3600.887	5493
TS2_LB						
Average	**1953.177**	**1223.3**	**610.2058**	**2465.272**	**2604.752**	**5250.9**
min	9.672	1082	2.312	2.724	323.627	5021
max	3600.116	1359	3600.168	2863	3600.685	5495

Με μια πρώτη ματιά στα παραπάνω αποτελέσματα μπορεί κανείς να αντιληφθεί ότι, η εφαρμογή των ορίων στην αντικειμενική συνάρτηση δεν επιφέρει καλύτερα αποτελέσματα εκτός από μεμονωμένες περιπτώσεις. Στην περίπτωση του Wilson μοντέλου με την εφαρμογή του πάνω ορίου για M=5 και N=50 υπάρχει μεγάλη μείωση του υπολογιστικού χρόνου αλλά δεν είναι αρκετό για να καταλήξουμε σε ένα σίγουρο συμπέρασμα. Για το μοντέλο TS2 η εφαρμογή των ορίων είχε αντίθετα αποτελέσματα από αυτά που επιζητούσαμε. Στην περίπτωση του άνω ορίου δεν είχαμε λύση που να ανταποκρίνεται στα κριτήρια μας μετά από μια ώρα υπολογιστικού χρόνου. Όσο αφορά

το μοντέλο WST δεν μπορούμε να θεωρήσουμε ότι υπάρχει κάποια επίδραση από την εφαρμογή του άνω και κάτω ορίου στην αντικειμενική συνάρτηση.

Στην συνέχεια, αποφασίσαμε να πειραματιστούμε χωρίς την απόλυτη ανοχή στην αντικειμενική συνάρτηση και να κρατήσουμε μόνο το πάνω και κάτω όριο. Τα αποτελέσματα ολόκληρης της διαδικασίας φαίνονται στο Παράρτημα Β.

Upper Bound						
	M=5 | N=20		**M=5 | N=50**		**M=5 | N=100**	
MODEL	**TIME**	**OBJ**	**TIME**	**OBJ**	**TIME**	**OBJ**
wilson_UB						
Average	**3600.089**	**1571.333**	**19.8737**	**2726.5**	**90.1413**	**5208.8**
min	3600.073	1473	1.157	2548	12.358	4956
max	3600.114	1650	77.876	2837	435.564	5435
wst_UB						
Average	**1388.239**	**1222.9**	**1322.247**	**2732**	**1455.452**	**5249**
min	25.419	1081	11.657	2552	141.351	5018
max	3600.07	1359	3600.481	2863	3600.919	5493
TS2_UB						
Average	**2153.335**	**1222**	**855.8453**	**2737.6**	**2457.416**	**5251.4**
min	2.265	1081	4.421	2552	141.932	5022
max	3600.053	1359	3600.018	2863	3601.048	5493

Lower Bound						
	M=5 | N=20		**M=5 | N=50**		**M=5 | N=100**	
MODEL	**TIME**	**OBJ**	**TIME**	**OBJ**	**TIME**	**OBJ**
wilson_LB						
Average	**3600.093**	**1567.667**	**112.3211**	**2727.2**	**55.4519**	**5218.6**
min	3600.086	1462	0.86	2548	11.922	4963
max	3600.101	1661	915.646	2838	141.595	5437
wst_LB						
Average	**1157.462**	**1223.6**	**1362.048**	**2737.5**	**1956.338**	**5248.7**
min	10.906	1091	14.625	2552	168.673	5018
max	3600.345	1359	3600.195	2863	3600.887	5493
TS2_LB						
Average	**1954.477**	**1223.3**	**609.143**	**2737.4**	**2491.299**	**5242**
min	9.672	1082	3.2656	2552	322.831	5021
max	3600.086	1359	3600.195	2863	3600.727	5495

Σε γενικές γραμμές, η αφαίρεση της απόλυτης ανοχής δεν επέφερε κάποια βελτίωση στα αποτελέσματα, με εξαίρεση το μοντέλο TS2, για το οποίο πήραμε λύσεις για N=50 και N=100 όταν εφαρμόζουμε πάνω όριο στην αντικειμενική συνάρτηση σε αντίθεση με τα προηγούμενα πειράματα.

5.2 Σχολιασμός Αποτελεσμάτων

Από την σύγκριση των 8 μοντέλων μικτού ακέραιου γραμμικού προγραμματισμού (MILP), που παρουσιάσαμε στο 4^{ο} Κεφάλαιο, ως προς τον υπολογιστικό χρόνο που απαιτούν για την επίλυση του προβλήματος προγραμματισμού εργασιών κοινής διάταξης, με αντικειμενική συνάρτηση την ελαχιστοποίηση του χρόνου περάτωσης των εργασιών, τα βασικά συμπεράσματα που καταλήγουμε είναι:

- ✓ Τα 3 μοντέλα της οικογένειας Wagner υπερτερούν των 5 της οικογένειας Manne, κάτι που το αναμέναμε με βάση προηγούμενες έρευνες.
- ✓ Η εφαρμογή της απόλυτης ανοχής στην τιμή της αντικειμενικής συνάρτησης δεν επέφερε κάποια διαφοροποίηση στα αποτελέσματα, το Wilson μοντέλο είναι καλύτερο από τα άλλα δύο, κάτι που αναμέναμε.
- ✓ Για σταθερό αριθμό εργασιών και αύξηση των μηχανών του συστήματος τα αποτελέσματα είναι απογοητευτικά αφού και για τα τρία μοντέλα, ο χρόνος που απαιτείται για επίλυση ξεπερνά την 1 ώρα και στην περίπτωση M=20 και N=100 δεν υπάρχει λύση πάνω στην μια ώρα.
- ✓ Η εφαρμογή του άνω και κάτω ορίου που πρότεινε ο καθηγητής Pastor Moreno Rafael δεν βοηθά δραστικά στην μείωση του υπολογιστικού χρόνου που απαιτείται για την επίλυση των προβλημάτων.
- ✓ Η εφαρμογή του άνω και κάτω ορίου επιφέρει μεμονωμένα καλά αποτελέσματα και δεν μπορούμε να καταλήξουμε σε ασφαλές συμπέρασμα για το αν κάποιο από τα 2 όρια που εφαρμόσαμε στην αντικειμενική συνάρτηση υπερτερεί.
- ✓ Η χρήση πάνω ορίου στο TS2 μοντέλο επιφέρει μη επιθυμητά αποτελέσματα αφού δεν υπάρχει εφικτή λύση σε μία ώρα επίλυσης.
- ✓ Η εφαρμογή των άνω και κάτω ορίων χωρίς την απόλυτη ανοχή 0.999 δεν επιφέρει καλύτερα αποτελέσματα .

Με βάση την έρευνα και τα αποτελέσματα αυτής εργασίας, υπάρχουν αρκετά σημεία που θα μπορούσαν να αποτελέσουν αρχή για περαιτέρω έρευνα. Πρώτον, θα ήταν καλό να ελεγχθεί η συμπεριφορά των μοντέλων SGST(PI) και LYsub(PI), της οικογένειας Manne, σε περίπτωση που μειωθεί η τιμή της σταθεράς PI. Δεύτερον, να ελέγξουμε την συμπεριφορά των μοντέλων σε περίπτωση που θεωρηθούν ως ακέραιες,

οι πραγματικές μεταβλητές οι οποίες παίρνουν ακέραια τιμή όταν ο χρόνος εκτέλεσης των εργασιών p_{ij} είναι ακέραιος.

Να ελεγχθεί το ενδεχόμενο πιθανής βελτίωσης που θα μπορούσε να επιφέρει η αλλαγή των παραμέτρων που χρησιμοποιήθηκαν από την CPLEX. Τέλος, ίσως να βοηθούσε η ανάθεση προτεραιότητας στις μεταβλητές κατά την διαδικασία διακλάδωσης.

6. Επίλογος

Σε αυτή την διπλωματική εργασία, μελετήσαμε μεθόδους προγραμματισμού εκτέλεσης εργασιών σε μια γραμμή παραγωγής (flowshop, κατάστημα ροής) με σκοπό την ελαχιστοποίηση του συνολικού χρόνου περάτωσης. Οι μέθοδοι αυτές διακρίνονται σε ευρετικές μεθόδους (GA, SA, Tabu Search, Ant Colony) και σε ακριβείς. Μετά την περιγραφή των μεθόδων δώσαμε έμφαση στις ακριβείς μεθόδους, οι οποίες μοντελοποιούν το πρόβλημα ως πρόβλημα μικτού ακέραιου γραμμικού προγραμματισμού. Στην κατηγορία αυτή ανήκουν 8 παραλλαγές: WST, Wilson, TS2, LYsub(PI), LYsub(pi), LYeq(PI), SGST(PI) και SGST(pi).

Η συνεισφορά της εργασίας είναι η συγκριτική αξιολόγηση των μοντέλων. Από αριθμητικά πειράματα με προβλήματα 5-10-20 μηχανών και 20-50-100 εργασιών προκύπτει ότι καλύτερη φαίνεται να είναι η οικογένεια μοντέλων Wagner, η οποία προτείνεται για πρακτικές εφαρμογές ως η πλέον αποδοτική από απόψεως χρόνων εκτέλεσης.

ΠΑΡΑΡΤΗΜΑ Α

1. WST MODEL

```
//DATA DECLARATION
int M=...;
int N=...;
int t[1..M,1..N]=...;
int cotainf=...;
int cotasup=...;

//VARIABLES
var int z[1..N,1..N] in 0..1; //zij: 1 if job i is assigned to seq
position j. O otherwise
var float+ x[1..M,1..N]; //Xrj: idle time on machine r before the start
of job in seq position j
var float+ y[1..M,1..N]; //Yrj: waiting time of job in seq position j
after it finishes processing on machine r

//CONSTRAINTS
constraint n2[1..N];
constraint n3[1..N];
constraint n4[1..M-1,1..N-1];
constraint n5[1..M-1];
constraint n6[1..M-1];

//MODEL
minimize
   sum(i in 1..N)t[M,i]+sum(p in 1..N)x[M,p]

subject to {
   forall (i in 1..N)
      n2[i]:sum(j in 1..N)z[i,j]=1;

   forall (j in 1..N)
      n3[j]:sum(i in 1..N)z[i,j]=1;

   forall(r in [1..M-1])
      forall(j in [1..N-1])
         n4[r,j]:sum(i in
1..N)(t[r,i]*z[i,j+1])+x[r,j+1]+y[r,j+1]=sum(i in
1..N)(t[r+1,i]*z[i,j])+x[r+1,j+1]+y[r,j];

   forall(r in [1..M-1])
      n5[r]:x[r+1,1]=x[r,1]+y[r,1]+sum(i in 1..N)(t[r,i]*z[i,1]);

   forall(r in [1..M-1])
      n6[r]:y[r,1]=0;

};
```

2. WST with Lower Bound

```
//DATA DECLARATION
int M=...;
int N=...;
int t[1..M,1..N]=...;
int cotainf=...;
int cotasup=...;

//VARIABLES
var int z[1..N,1..N] in 0..1; //zij: 1 if job i is assigned to seq
position j. O otherwise
var float+ x[1..M,1..N]; //Xrj: idle time on machine r before the start
of job in seq position j
var float+ y[1..M,1..N]; //Yrj: waiting time of job in seq position j
after it finishes processing on machine r
var float+ obj; //objective function

//CONSTRAINTS
constraint n2[1..N];
constraint n3[1..N];
constraint n4[1..M-1,1..N-1];
constraint n5[1..M-1];
constraint n6[1..M-1];

//MODEL
minimize
   obj

subject to {

   obj=sum(i in 1..N)t[M,i]+sum(p in 1..N)x[M,p];
   obj>=cotainf;

   forall (i in 1..N)
      n2[i]:sum(j in 1..N)z[i,j]=1;

   forall (j in 1..N)
      n3[j]:sum(i in 1..N)z[i,j]=1;

   forall(r in [1..M-1])
      forall(j in [1..N-1])
         n4[r,j]:sum(i in
1..N)(t[r,i]*z[i,j+1])+x[r,j+1]+y[r,j+1]=sum(i in
1..N)(t[r+1,i]*z[i,j])+x[r+1,j+1]+y[r,j];

   forall(r in [1..M-1])
      n5[r]:x[r+1,1]=x[r,1]+y[r,1]+sum(i in 1..N)(t[r,i]*z[i,1]);

   forall(r in [1..M-1])
      n6[r]:y[r,1]=0;

};
```

3. WST with Upper Bound

```
//DATA DECLARATION
int M=...;
int N=...;
int t[1..M,1..N]=...;
int cotainf=...;
int cotasup=...;

//VARIABLES
var int z[1..N,1..N] in 0..1; //zij: 1 if job i is assigned to seq
position j. O otherwise
var float+ x[1..M,1..N]; //Xrj: idle time on machine r before the start
of job in seq position j
var float+ y[1..M,1..N]; //Yrj: waiting time of job in seq position j
after it finishes processing on machine r
var float+ obj; //objective function

//CONSTRAINTS
constraint n2[1..N];
constraint n3[1..N];
constraint n4[1..M-1,1..N-1];
constraint n5[1..M-1];
constraint n6[1..M-1];

//MODEL
minimize
   obj

subject to {

   obj=sum(i in 1..N)t[M,i]+sum(p in 1..N)x[M,p];
   obj<=cotasup;

   forall (i in 1..N)
      n2[i]:sum(j in 1..N)z[i,j]=1;

   forall (j in 1..N)
      n3[j]:sum(i in 1..N)z[i,j]=1;

   forall(r in [1..M-1])
      forall(j in [1..N-1])
         n4[r,j]:sum(i in
1..N)(t[r,i]*z[i,j+1])+x[r,j+1]+y[r,j+1]=sum(i in
1..N)(t[r+1,i]*z[i,j])+x[r+1,j+1]+y[r,j];

   forall(r in [1..M-1])
      n5[r]:x[r+1,1]=x[r,1]+y[r,1]+sum(i in 1..N)(t[r,i]*z[i,1]);

   forall(r in [1..M-1])
      n6[r]:y[r,1]=0;

};
```

4. WILSON MODEL

```
//DATA DECLARATION
int M=...;
int N=...;
int t[1..M,1..N]=...;
int cotainf=...;
int cotasup=...;

//VARIABLES
var int z[1..N,1..N] in 0..1; //zij: 1 if job i is assigned to seq
position j. O otherwise
var float+ b[1..M,1..N]; //brj: start time of job in seq position j on
machine r

//CONSTRAINTS
constraint n2[1..N];
constraint n3[1..N];
constraint n7[1..N-1];
constraint n8;
constraint n9[1..M-1];
constraint n10[1..M-1,2..N];
constraint n11[2..M,1..N-1];

//MODEL
minimize
   b[M,N] + sum(i in 1..N)t[M,i]*z[i,N]

subject to {

   forall (i in 1..N)
      n2[i]:sum(j in 1..N)z[i,j]=1;

   forall (j in 1..N)
      n3[j]:sum(i in 1..N)z[i,j]=1;

   n8: b[1,1]=0;

   forall(r in [1..M-1])
         n9[r]:b[r,1]+sum(i in 1..N)(t[r,i]*z[i,1])=b[r+1,1];

   forall(r in [1..M-1])
      forall(j in 2..N)
         n10[r,j]:b[r,j]+sum(i in 1..N)(t[r,i]*z[i,j])<=b[r+1,j];

   forall(r in 2..M)
      forall(j in [1..N-1])
         n11[r,j]:b[r,j]+sum(i in 1..N)(t[r,i]*z[i,j])<=b[r,j+1];

};
```

5. WILSON with Lower Bound

```
//DATA DECLARATION
int M=...;
int N=...;
int t[1..M,1..N]=...;
int cotainf=...;
int cotasup=...;

//VARIABLES
var int z[1..N,1..N] in 0..1; //zij: 1 if job i is assigned to seq
position j. O otherwise
var float+ b[1..M,1..N]; //brj: start time of job in seq position j on
machine r
var float+ obj; //objective function

//CONSTRAINTS
constraint n2[1..N];
constraint n3[1..N];
constraint n7[1..N-1];
constraint n8;
constraint n9[1..M-1];
constraint n10[1..M-1,2..N];
constraint n11[2..M,1..N-1];

//MODEL
minimize
   obj

subject to {

   obj=b[M,N] + sum(i in 1..N)t[M,i]*z[i,N];

   obj>=cotainf;

   forall (i in 1..N)
      n2[i]:sum(j in 1..N)z[i,j]=1;

   forall (j in 1..N)
      n3[j]:sum(i in 1..N)z[i,j]=1;

   n8: b[1,1]=0;

   forall(r in [1..M-1])
         n9[r]:b[r,1]+sum(i in 1..N)(t[r,i]*z[i,1])=b[r+1,1];

   forall(r in [1..M-1])
      forall(j in 2..N)
         n10[r,j]:b[r,j]+sum(i in 1..N)(t[r,i]*z[i,j])<=b[r+1,j];

   forall(r in 2..M)
      forall(j in [1..N-1])
         n11[r,j]:b[r,j]+sum(i in 1..N)(t[r,i]*z[i,j])<=b[r,j+1];

};
```

6. WILSON with Upper Bound

```
//DATA DECLARATION
int M=...;
int N=...;
int t[1..M,1..N]=...;
int cotainf=...;
int cotasup=...;

//VARIABLES
var int z[1..N,1..N] in 0..1; //zij: 1 if job i is assigned to seq
position j. O otherwise
var float+ b[1..M,1..N]; //brj: start time of job in seq position j on
machine r
var float+ obj; //objective function

//CONSTRAINTS
constraint n2[1..N];
constraint n3[1..N];
constraint n7[1..N-1];
constraint n8;
constraint n9[1..M-1];
constraint n10[1..M-1,2..N];
constraint n11[2..M,1..N-1];

//MODEL
minimize
   obj

subject to {

   obj=b[M,N] + sum(i in 1..N)t[M,i]*z[i,N];

   obj<=cotasup;

   forall (i in 1..N)
      n2[i]:sum(j in 1..N)z[i,j]=1;

   forall (j in 1..N)
      n3[j]:sum(i in 1..N)z[i,j]=1;

   n8: b[1,1]=0;

   forall(r in [1..M-1])
         n9[r]:b[r,1]+sum(i in 1..N)(t[r,i]*z[i,1])=b[r+1,1];

   forall(r in [1..M-1])
      forall(j in 2..N)
         n10[r,j]:b[r,j]+sum(i in 1..N)(t[r,i]*z[i,j])<=b[r+1,j];

   forall(r in 2..M)
      forall(j in [1..N-1])
         n11[r,j]:b[r,j]+sum(i in 1..N)(t[r,i]*z[i,j])<=b[r,j+1];

};
```

7. TS2 MODEL

```
//DATA DECLARATION
int M=...;
int N=...;
int t[1..M,1..N]=...;
int cotainf=...;
int cotasup=...;

//VARIABLES
var int z[1..N,1..N] in 0..1; //zij: 1 if job i is assigned to seq
position j. O otherwise
var float+ e[1..M,1..N];

//CONSTRAINTS
constraint n2[1..N];
constraint n3[1..N];
constraint n12[1..M,1..N-1];
constraint n13[1..M-1,1..N];
constraint n14;

//MODEL
minimize
   e[M,N]

subject to {

   forall (i in 1..N)
      n2[i]:sum(j in 1..N)z[i,j]=1;

   forall (j in 1..N)
      n3[j]:sum(i in 1..N)z[i,j]=1;

   forall(r in [1..M])
      forall(j in [1..N-1])
         n12[r,j]:e[r,j]+sum(i in 1..N)(t[r,i]*z[i,j+1])<=e[r,j+1];

   forall(r in [1..M-1])
      forall(j in [1..N])
         n13[r,j]:e[r,j]+sum(i in 1..N)(t[r+1,i]*z[i,j])<=e[r+1,j];

   n14:e[1,1]>=sum(i in 1..N)(t[1,i]*z[i,1]);

};
```

8. TS2 with Lower Bound

```
//DATA DECLARATION
int M=...;
int N=...;
int t[1..M,1..N]=...;
int cotainf=...;
int cotasup=...;

//VARIABLES
var int z[1..N,1..N] in 0..1; //zij: 1 if job i is assigned to seq
position j. O otherwise
var float+ e[1..M,1..N];
var float+ obj;

//CONSTRAINTS
constraint n2[1..N];
constraint n3[1..N];
constraint n12[1..M,1..N-1];
constraint n13[1..M-1,1..N];
constraint n14;

//MODEL
minimize
   obj

subject to {

   obj=e[M,N];
   obj>=cotainf;

   forall (i in 1..N)
      n2[i]:sum(j in 1..N)z[i,j]=1;

   forall (j in 1..N)
      n3[j]:sum(i in 1..N)z[i,j]=1;

   forall(r in [1..M])
      forall(j in [1..N-1])
         n12[r,j]:e[r,j]+sum(i in 1..N)(t[r,i]*z[i,j+1])<=e[r,j+1];

   forall(r in [1..M-1])
      forall(j in [1..N])
         n13[r,j]:e[r,j]+sum(i in 1..N)(t[r+1,i]*z[i,j])<=e[r+1,j];

   n14:e[1,1]>=sum(i in 1..N)(t[1,i]*z[i,1]);

};
```

9. TS2 with Upper Bound

```
//DATA DECLARATION
int M=...;
int N=...;
int t[1..M,1..N]=...;
int cotainf=...;
int cotasup=...;

//VARIABLES
var int z[1..N,1..N] in 0..1; //zij: 1 if job i is assigned to seq
position j. O otherwise
var float+ e[1..M,1..N];
var float+ obj;

//CONSTRAINTS
constraint n2[1..N];
constraint n3[1..N];
constraint n12[1..M,1..N-1];
constraint n13[1..M-1,1..N];
constraint n14;

//MODEL
minimize
   obj

subject to {

   obj=e[M,N];
   obj<=cotasup;

   forall (i in 1..N)
      n2[i]:sum(j in 1..N)z[i,j]=1;

   forall (j in 1..N)
      n3[j]:sum(i in 1..N)z[i,j]=1;

   forall(r in [1..M])
      forall(j in [1..N-1])
         n12[r,j]:e[r,j]+sum(i in 1..N)(t[r,i]*z[i,j+1])<=e[r,j+1];

   forall(r in [1..M-1])
      forall(j in [1..N])
         n13[r,j]:e[r,j]+sum(i in 1..N)(t[r+1,i]*z[i,j])<=e[r+1,j];

   n14:e[1,1]>=sum(i in 1..N)(t[1,i]*z[i,1]);

};
```

10. SGST(PI) MODEL

```
//DATA DECLARATION
int M=...;
int N=...;
int t[1..M,1..N]=...;
int cotainf=...;
int cotasup=...;
int p=30000;
struct couple{
   int a;
   int b;
};
{couple} ct178={<a,b>|a in [1..N-1] & b in [2..N]: b>a}; //to define
range of constraints 17 and 18

//VARIABLES
var int d[1..N,1..N] in 0..1;
var float+ c[1..M,1..N];
var float+ cmax;

//CONSTRAINTS
constraint n15[1..N];
constraint n16[1..M-1,1..N];
constraint n17[1..M,ct178];
constraint n18[1..M,ct178];
constraint n19[1..N];

//MODEL
minimize
  cmax

subject to {
   forall (i in [1..N])
      n15[i]:c[1,i]>=t[1,i];

   forall (r in [1..M-1])
      forall(i in [1..N])
      n16[r,i]:c[r+1,i]-c[r,i]>=t[r+1,i];

   forall(r in [1..M])
      forall(i in [1..N-1])
         forall(k in [i+1..N])
   //      forall(ordered i,k in [1..N])
            n17[r,<i,k>]:c[r,i]-c[r,k]+(p*d[i,k])>=t[r,i];


   forall(r in [1..M])
      forall(i in [1..N-1])
         forall(k in [i+1..N])
   //      forall(ordered i,k in [1..N])
            n18[r,<i,k>]:c[r,k]-c[r,i]+p*(1-d[i,k])>=t[r,k];

   forall(i in 1..N)
      n19[i]:cmax>=c[M,i];
};
```

11. SGST_pi MODEL

```
//DATA DECLARATION
int M=...;
int N=...;
int t[1..M,1..N]=...;
int cotainf=...;
int cotasup=...;
int p=(max(i in 1..M, j in 1..N)t[i,j])*N;
struct couple{
   int a;
   int b;
};

{couple} ct178={<a,b>|a in [1..N-1] & b in [2..N]: b>a}; //to define
range of constraints 17 and 18

//VARIABLES
var int d[1..N,1..N] in 0..1;
var float+ c[1..M,1..N];
var float+ cmax;

//CONSTRAINTS
constraint n15[1..N];
constraint n16[1..M-1,1..N];
constraint n17[1..M,ct178];
constraint n18[1..M,ct178];
constraint n19[1..N];

//MODEL
minimize
  cmax

subject to {
   forall (i in [1..N])
      n15[i]:c[1,i]>=t[1,i];

   forall (r in [1..M-1])
      forall(i in [1..N])
      n16[r,i]:c[r+1,i]-c[r,i]>=t[r+1,i];

   forall(r in [1..M])
      forall(i in [1..N-1])
         forall(k in [i+1..N])
   //      forall(ordered i,k in [1..N])
            n17[r,<i,k>]:c[r,i]-c[r,k]+(p*d[i,k])>=t[r,i];

   forall(r in [1..M])
      forall(i in [1..N-1])
         forall(k in [i+1..N])
   //      forall(ordered i,k in [1..N])
            n18[r,<i,k>]:c[r,k]-c[r,i]+p*(1-d[i,k])>=t[r,k];

   forall(i in 1..N)
      n19[i]:cmax>=c[M,i];
};
```

12. LYsub(PI) MODEL

```
//DATA DECLARATION
int M=...;
int N=...;
int t[1..M, 1..N]=...;
int cotainf=...;
int cotasup=...;
int p=30000;
struct couple{
   int a;
   int b;
};
{couple} ct212={<a,b>|a in [1..N-1] & b in [2..N]: b>a}; //to define
range of variable q and constraints 21 and 22

int qbound[r in 1..M, ik in ct212]=0;
initialize
forall(r in 1..M)
   forall(i in [1..N-1])
      forall(k in [i+1..N])
         qbound[r,<i,k>]=p-t[r,i]-t[r,k];

//VARIABLES
var int d[1..N,1..N] in 0..1;
var float+ c[1..M,1..N];
var float+ q[r in 1..M,ik in ct212] in 0..qbound[r,ik];
var float+ cmax;

//CONSTRAINTS
constraint n15[1..N];
constraint n16[1..M-1,1..N];
constraint n19[1..N];
constraint n21[1..M,ct212];

//MODEL
minimize
  cmax
subject to {

   forall (i in [1..N])
      n15[i]:c[1,i]>=t[1,i];

   forall (r in [1..M-1])
      forall(i in [1..N])
      n16[r,i]:c[r+1,i]-c[r,i]>=t[r+1,i];

   forall(i in 1..N)
      n19[i]:cmax>=c[M,i];

   forall(r in [1..M])
      forall(i in [1..N-1])
         forall(k in [i+1..N])
            n21[r,<i,k>]:p*d[i,k]+c[r,i]-c[r,k]-t[r,i]=q[r,<i,k>];

};
```

13. LYsub_pi MODEL

```
//DATA DECLARATION
int M=...;
int N=...;
int t[1..M,1..N]=...;
int cotainf=...;
int cotasup=...;
int p=(max(i in 1..M, j in 1..N)t[i,j])*N;
struct couple{
   int a;
   int b;
};
{couple} ct212={<a,b>|a in [1..N-1] & b in [2..N]: b>a}; //to define
range of variable q and constraints 21 and 22

int qbound[r in 1..M, ik in ct212]=0;
initialize
forall(r in 1..M)
   forall(i in [1..N-1])
      forall(k in [i+1..N])
         qbound[r,<i,k>]=p-t[r,i]-t[r,k];

//VARIABLES
var int d[1..N,1..N] in 0..1;
var float+ c[1..M,1..N];
var float+ q[r in 1..M,ik in ct212] in 0..qbound[r,ik];
var float+ cmax;

//CONSTRAINTS
constraint n15[1..N];
constraint n16[1..M-1,1..N];
constraint n19[1..N];
constraint n21[1..M,ct212];

//MODEL
minimize
  cmax

subject to {
   forall (i in [1..N])
      n15[i]:c[1,i]>=t[1,i];

   forall (r in [1..M-1])
      forall(i in [1..N])
      n16[r,i]:c[r+1,i]-c[r,i]>=t[r+1,i];

   forall(i in 1..N)
      n19[i]:cmax>=c[M,i];

   forall(r in [1..M])
      forall(i in [1..N-1])
         forall(k in [i+1..N])
            n21[r,<i,k>]:p*d[i,k]+c[r,i]-c[r,k]-t[r,i]=q[r,<i,k>];

};
```

14. LYeq(PI) MODEL

```
//DATA DECLARATION
int M=...;
int N=...;
int t[1..M,1..N]=...;
//int P=...;
int cotainf=...;
int cotasup=...;
int p=30000;
struct couple{
   int a;
   int b;
};
{couple} ct212={<a,b>|a in [1..N-1] & b in [2..N]: b>a}; //to define
range of variable q and constraints 21 and 22

//VARIABLES
var int d[1..N,1..N] in 0..1;
var float+ c[1..M,1..N];
var float+ q[1..M,ct212];
var float+ cmax;

//CONSTRAINTS
constraint n15[1..N];
constraint n16[1..M-1,1..N];
constraint n19[1..N];
constraint n21[1..M,ct212];
constraint n22[1..M,ct212];

//MODEL
minimize
  cmax

subject to {
   forall (i in [1..N])
      n15[i]:c[1,i]>=t[1,i];

   forall (r in [1..M-1])
      forall(i in [1..N])
      n16[r,i]:c[r+1,i]-c[r,i]>=t[r+1,i];

   forall(i in 1..N)
      n19[i]:cmax>=c[M,i];

   forall(r in [1..M])
      forall(i in [1..N-1])
         forall(k in [i+1..N])
            n21[r,<i,k>]:p*d[i,k]+c[r,i]-c[r,k]-t[r,i]=q[r,<i,k>];

   forall(r in [1..M])
      forall(i in [1..N-1])
         forall(k in [i+1..N])
            n22[r,<i,k>]:q[r,<i,k>]<=p-t[r,i]-t[r,k];

};
```

Παράδειγμα script αρχείου για την εκτέλεση των πειραμάτων

```
//DECLARATION OF MODELS
Model wst_20_5_01("Z:\MILP\Models\wst.mod","Z:\MILP\Data\20_5_01.dat") editMode;
Model wst_20_5_02("Z:\MILP\Models\wst.mod","Z:\MILP\Data\20_5_02.dat") editMode;
Model wst_20_5_03("Z:\MILP\Models\wst.mod","Z:\MILP\Data\20_5_03.dat") editMode;
Model wst_20_5_04("Z:\MILP\Models\wst.mod","Z:\MILP\Data\20_5_04.dat") editMode;
Model wst_20_5_05("Z:\MILP\Models\wst.mod","Z:\MILP\Data\20_5_05.dat") editMode;
Model wst_20_5_06("Z:\MILP\Models\wst.mod","Z:\MILP\Data\20_5_06.dat") editMode;
Model wst_20_5_07("Z:\MILP\Models\wst.mod","Z:\MILP\Data\20_5_07.dat") editMode;
Model wst_20_5_08("Z:\MILP\Models\wst.mod","Z:\MILP\Data\20_5_08.dat") editMode;
Model wst_20_5_09("Z:\MILP\Models\wst.mod","Z:\MILP\Data\20_5_09.dat") editMode;
Model wst_20_5_10("Z:\MILP\Models\wst.mod","Z:\MILP\Data\20_5_10.dat") editMode;

//FILE
ofile RESULTS("Z:\MILP\RESULTS\Results_wst.txt");
RESULTS <<"Model / M / N / Instance / Time / Obj"<<endl;

//SOLVING MODELS - wst without bounds, 10 instances
wst_20_5_01.setFloatParameter("tiLim", 3600);
wst_20_5_01.setFloatParameter("epGap", 0.001);
//wst_20_5_01.setFloatParameter("epAGap", 0.999);
if wst_20_5_01.nextSolution() then
{
   cout <<"wst / 5 / 20 / 1 /"<<wst_20_5_01.getTime()<<" / "<<wst_20_5_01.objectiveValue()<<endl;
   RESULTS <<"wst / 5 / 20 / 1 /"<<wst_20_5_01.getTime()<<" / "<<wst_20_5_01.objectiveValue()<<endl;
}

wst_20_5_02.setFloatParameter("tiLim", 3600);
wst_20_5_02.setFloatParameter("epGap", 0.001);
//wst_20_5_02.setFloatParameter("epAGap", 0.999);
if wst_20_5_02.nextSolution() then
{
   cout <<"wst / 5 / 20 / 2 /"<<wst_20_5_02.getTime()<<" / "<<wst_20_5_02.objectiveValue()<<endl;
   RESULTS <<"wst / 5 / 20 / 2 /"<<wst_20_5_02.getTime()<<" / "<<wst_20_5_02.objectiveValue()<<endl;
}
```

```
wst_20_5_03.setFloatParameter("tiLim", 3600);
wst_20_5_03.setFloatParameter("epGap", 0.001);
//wst_20_5_03.setFloatParameter("epAGap", 0.999);
if wst_20_5_03.nextSolution() then
{
   cout <<"wst / 5 / 20 / 3 /"<<wst_20_5_03.getTime()<<" / "<<wst_20_5_03.objectiveValue()<<endl;
   RESULTS <<"wst / 5 / 20 / 3 /"<<wst_20_5_03.getTime()<<" / "<<wst_20_5_03.objectiveValue()<<endl;
}

wst_20_5_04.setFloatParameter("tiLim", 3600);
wst_20_5_04.setFloatParameter("epGap", 0.001);
//wst_20_5_04.setFloatParameter("epAGap", 0.999);
if wst_20_5_04.nextSolution() then
{
   cout <<"wst / 5 / 20 / 4 /"<<wst_20_5_04.getTime()<<" / "<<wst_20_5_04.objectiveValue()<<endl;
   RESULTS <<"wst / 5 / 20 / 4 /"<<wst_20_5_04.getTime()<<" / "<<wst_20_5_04.objectiveValue()<<endl;
}

wst_20_5_05.setFloatParameter("tiLim", 3600);
wst_20_5_05.setFloatParameter("epGap", 0.001);
//wst_20_5_05.setFloatParameter("epAGap", 0.999);
if wst_20_5_05.nextSolution() then
{
   cout <<"wst / 5 / 20 / 5 /"<<wst_20_5_05.getTime()<<" / "<<wst_20_5_05.objectiveValue()<<endl;
   RESULTS <<"wst / 5 / 20 / 5 /"<<wst_20_5_05.getTime()<<" / "<<wst_20_5_05.objectiveValue()<<endl;
}

wst_20_5_06.setFloatParameter("tiLim", 3600);
wst_20_5_06.setFloatParameter("epGap", 0.001);
//wst_20_5_06.setFloatParameter("epAGap", 0.999);

if wst_20_5_06.nextSolution() then
{
   cout <<"wst / 5 / 20 / 6 /"<<wst_20_5_06.getTime()<<" / "<<wst_20_5_06.objectiveValue()<<endl;
   RESULTS <<"wst / 5 / 20 / 6 /"<<wst_20_5_06.getTime()<<" / "<<wst_20_5_06.objectiveValue()<<endl;
}
```

```
wst_20_5_07.setFloatParameter("tiLim", 3600);
wst_20_5_07.setFloatParameter("epGap", 0.001);
//wst_20_5_07.setFloatParameter("epAGap", 0.999);
if wst_20_5_07.nextSolution() then
{
   cout <<"wst / 5 / 20 / 7 /"<<wst_20_5_07.getTime()<<" / "<<wst_20_5_07.objectiveValue()<<endl;
   RESULTS <<"wst / 5 / 20 / 7 /"<<wst_20_5_07.getTime()<<" / "<<wst_20_5_07.objectiveValue()<<endl;
}

wst_20_5_08.setFloatParameter("tiLim", 3600);
wst_20_5_08.setFloatParameter("epGap", 0.001);
//wst_20_5_08.setFloatParameter("epAGap", 0.999);
if wst_20_5_08.nextSolution() then
{
   cout <<"wst / 5 / 20 / 8 /"<<wst_20_5_08.getTime()<<" / "<<wst_20_5_08.objectiveValue()<<endl;
   RESULTS <<"wst / 5 / 20 / 8 /"<<wst_20_5_08.getTime()<<" / "<<wst_20_5_08.objectiveValue()<<endl;
}

wst_20_5_09.setFloatParameter("tiLim", 3600);
wst_20_5_09.setFloatParameter("epGap", 0.001);
//wst_20_5_09.setFloatParameter("epAGap", 0.999);
if wst_20_5_09.nextSolution() then
{
   cout <<"wst / 5 / 20 / 9 /"<<wst_20_5_09.getTime()<<" / "<<wst_20_5_09.objectiveValue()<<endl;
   RESULTS <<"wst / 5 / 20 / 9 /"<<wst_20_5_09.getTime()<<" / "<<wst_20_5_09.objectiveValue()<<endl;
}

wst_20_5_10.setFloatParameter("tiLim", 3600);
wst_20_5_10.setFloatParameter("epGap", 0.001);
//wst_20_5_10.setFloatParameter("epAGap", 0.999);
if wst_20_5_10.nextSolution() then
{
   cout <<"wst / 5 / 20 / 10 /"<<wst_20_5_10.getTime()<<" / "<<wst_20_5_10.objectiveValue()<<endl;
   RESULTS <<"wst / 5 / 20 / 10 /"<<wst_20_5_10.getTime()<<" / "<<wst_20_5_10.objectiveValue()<<endl;
}

//CLOSING FILE
RESULTS.close();
```

ΠΑΡΑΡΤΗΜΑ Β

1. Αποτελέσματα μοντέλου Wilson για M=5 και N=20

				with 0.999						without 0.999			
				normal		with LB		with UB		with LB		with UB	
MODEL	M	N	instance	time	obj	time	obj	time	obj	time	obj	time	obj
wilson	5	20	1	0.484	1164	0.109	1232	1.641	1165	0.093	1232	1.625	1165
	5	20	2	0.875	1349	0.625	1349	0.829	1349	0.516	1349	0.828	1349
	5	20	3	0.453	1073	4.078	1073	2.703	1073	3.437	1073	2.719	1073
	5	20	4	0.672	1269	15.749	1268	1.547	1268	11.625	1268	1.547	1268
	5	20	5	0.094	1149	15.858	1198	0.625	1149	0.094	1198	0.625	1149
	5	20	6	0.25	1193	16.39	1194	0.407	1193	0.531	1194	0.39	1193
	5	20	7	15.203	1234	16.827	1234	0.312	1234	0.442	1234	0.313	1234
	5	20	8	1.734	1197	20.968	1198	2.203	1197	4.11	1198	2.187	1197
	5	20	9	0.938	1208	21.389	1208	0.468	1208	0.422	1208	0.469	1208
	5	20	10	20.515	1089	62.231	1089	3599.94	1089	40.501	1089	3600.07	1089
Average				**4.1218**	***1192.5***	**17.4224**	***1204.3***	**361.068**	***1192.5***	**6.1771**	***1204.3***	**361.077**	***1192.5***
min				**0.25**	***1073***	**0.109**	***1073***	**0.312**	***1073***	**0.093**	***1073***	**0.313**	***1073***
max				**20.515**	***1349***	**62.231**	***1349***	**3599.94**	***1349***	**40.501**	***1349***	**3600.07**	***1349***

2. Αποτελέσματα μοντέλου Wilson για M=5 και N=50

				with 0.999						without 0.999			
				normal		with LB		with UB		with LB		with UB	
MODEL	M	N	instance	time	obj	time	obj	time	obj	time	obj	time	obj
wilson	5	50	1	1.047	2712	0.859	2712	1.781	2712	0.86	2712	1.797	2712
	5	50	2	54.984	2832	41.984	2832	20.969	2833	41.907	2832	20.953	2833
	5	50	3	3600.035	2622	70.342	2620	79.83	2620	69.969	2620	77.876	2620
	5	50	4	164.811	2744	72.139	2744	65.673	2744	71.813	2744	65.469	2744
	5	50	5	1.75	2837	1.016	2838	1.14	2837	1.031	2838	1.157	2837
	5	50	6	30.343	2831	6.688	2829	8.625	2829	6.641	2829	8.657	2829
	5	50	7	4.593	2682	925.101	2689	4.297	2682	915.646	2689	4.328	2682
	5	50	8	2.735	2678	2.906	2678	1.36	2678	2.891	2678	1.391	2678
	5	50	9	29.719	2548	11.234	2548	15.718	2548	11.172	2548	15.812	2548
	5	50	10	5.234	2782	1.266	2782	1.282	2782	1.281	2782	1.297	2782
Average				**389.5251**	**2726.8**	**113.3535**	**2727.2**	**20.0675**	**2726.5**	**112.321**	***2727.2***	**19.8737**	***2726.5***
min				**1.047**	**2548**	**0.859**	**2548**	**1.14**	**2548**	**0.86**	***2548***	**1.157**	***2548***
max				**3600.035**	**2837**	**925.101**	**2838**	**79.83**	**2837**	**915.646**	***2838***	**77.876**	***2837***

3. Αποτελέσματα μοντέλου Wilson για M=5 και N=100

				with 0.999						without 0.999			
				normal		with LB		with UB		with LB		with UB	
MODEL	M	N	instance	time	obj	time	obj	time	obj	time	obj	time	obj
wilson	5	100	1	66.624	5349	12.015	5437	93.173	5347	11.922	5437	93.36	5347
	5	100	2	44.796	5241	97.988	5243	435.951	5240	96.345	5243	435.564	5240
	5	100	3	87.452	5173	50.686	5173	48.904	5172	49.344	5173	48.904	5172
	5	100	4	10.969	4960	19.952	4963	12.359	4956	19.531	4963	12.358	4956
	5	100	5	133.092	5244	112.403	5244	52.748	5245	111.157	5244	52.951	5245
	5	100	6	24.156	5121	31.499	5121	37.78	5121	31.234	5121	37.905	5121
	5	100	7	22.781	5221	14.64	5221	17.046	5224	14.578	5221	17.093	5224
	5	100	8	112.264	5076	142.684	5074	121.154	5074	141.595	5074	121.044	5074
	5	100	9	52.718	5435	29.327	5435	30.438	5435	29.094	5435	29.859	5435
	5	100	10	20.125	5274	51.061	5275	53.5	5274	49.719	5275	52.375	5274
Average				**57.4977**	***5209.4***	**56.2255**	***5218.6***	**90.3053**	***5208.8***	**55.4519**	***5218.6***	**90.1413**	***5208.8***
min				**10.969**	***4960***	**12.015**	***4963***	**12.359**	***4956***	**11.922**	***4963***	**12.358**	***4956***
max				**133.092**	***5435***	**142.684**	***5437***	**435.951**	***5435***	**141.595**	***5437***	**435.564**	***5435***

4. Αποτελέσματα WST μοντέλου για M=5 και N=20

				with 0.999						without 0.999			
				normal		with LB		with UB		with LB		with UB	
MODEL	M	N	instance	time	obj	time	obj	time	obj	time	obj	time	obj
wst	5	20	1	16.687	1279	10.922	1278	159.57	1279	10.906	1278	159.486	1279
	5	20	2	3599.96	1359	82.141	1359	2329.39	1359	81.984	1359	2350.39	1359
	5	20	3	3599.96	1081	3600.6	1091	329.626	1081	3599.936	1091	332.455	1081
	5	20	4	3599.96	1297	413.424	1294	3600.33	1297	418.382	1294	3600.07	1297
	5	20	5	419.726	1235	177.423	1235	3600.29	1235	178.794	1235	3600.05	1235
	5	20	6	19.999	1195	3600.06	1195	53.206	1195	3600.345	1195	53.391	1195
	5	20	7	3599.96	1247	3600.08	1239	3600.26	1239	3600.254	1239	3600.07	1239
	5	20	8	937.201	1207	46.469	1206	28.267	1206	46.894	1206	28.235	1206
	5	20	9	328.978	1230	25.156	1230	131.835	1230	25.251	1230	132.823	1230
	5	20	10	2841.12	1109	11.813	1109	25.361	1108	11.876	1109	25.419	1108
Average				**1896.36**	**1223.9**	**1156.81**	**1223.6**	**1385.81**	**1222.9**	**1157.462**	**1223.6**	**1388.24**	**1222.9**
min				**16.687**	**1081**	**10.922**	**1091**	**25.361**	**1081**	**10.906**	**1091**	**25.419**	**1081**
max				**3599.96**	**1359**	**3600.6**	**1359**	**3600.33**	**1359**	**3600.345**	**1359**	**3600.07**	**1359**

5. Αποτελέσματα WST μοντέλου για M=5 και N=50

				with 0.999						without 0.999			
				normal		with LB		with UB		with LB		with UB	
MODEL	M	N	instance	time	obj	time	obj	time	obj	time	obj	time	obj
wst	5	50	1	113.092	2725	19.109	2725	22.736	2725	19.109	2725	22.736	2725
	5	50	2	3600.13	2838	1910.14	2834	276.327	2834	1910.138	2834	276.327	2834
	5	50	3	183.387	2622	259.739	2622	93.176	2621	259.739	2622	93.176	2621
	5	50	4	3600.15	2755	3600.18	2751	3600.43	2756	3600.181	2751	3600.43	2756
	5	50	5	13.578	2864	134.938	2863	3600.48	2863	134.938	2863	3600.48	2863
	5	50	6	26.109	2830	14.625	2829	11.657	2829	14.625	2829	11.657	2829
	5	50	7	770.329	2725	3600.2	2730	172.572	2725	3600.195	2730	172.572	2725
	5	50	8	731.205	2685	365.8	2685	524.464	2683	365.8	2685	524.464	2683
	5	50	9	3600.46	2552	3600.18	2552	3598.38	2552	3600.18	2552	3598.38	2552
	5	50	10	36.109	2783	115.579	2784	no solution	no solution	115.579	2784	no solution	no solution
Average				**1267.46**	**2737.9**	**1362.05**	**2737.5**	**1322.25**	**2732**	**1362.048**	**2737.5**	**1322.25**	**2732**
min				**13.578**	**2552**	**14.625**	**2552**	**11.657**	**2552**	**14.625**	**2552**	**11.657**	**2552**
max				**3600.46**	**2864**	**3600.2**	**2863**	**3600.48**	**2863**	**3600.195**	**2863**	**3600.48**	**2863**

6. Αποτελέσματα WST μοντέλου για M=5 και N=100

				with 0.999						without 0.999			
				normal		with LB		with UB		with LB		with UB	
MODEL	M	N	instance	time	obj	time	obj	time	obj	time	obj	time	obj
wst	5	100	1	150.356	5493	168.673	5493	150.789	5493	168.673	5493	150.789	5493
	5	100	2	3600.67	5268	3600.59	5278	3600.89	5274	3600.585	5278	3600.89	5274
	5	100	3	1337.62	5179	1066.37	5180	1262.27	5178	1066.374	5180	1262.27	5178
	5	100	4	2662.7	5018	2985.21	5018	684.222	5018	2985.206	5018	684.222	5018
	5	100	5	964.825	5253	323.11	5250	141.351	5250	323.11	5250	141.351	5250
	5	100	6	207.746	5139	301.874	5137	272.233	5139	301.874	5137	272.23	5139
	5	100	7	1237.74	5251	315.016	5250	417.287	5251	315.016	5250	417.287	5251
	5	100	8	3600.7	5099	3600.78	5096	3600.92	5104	3600.778	5096	3600.92	5104
	5	100	9	1560.14	5451	3600.89	5458	823.947	5448	3600.887	5458	823.947	5448
	5	100	10	3600.68	5340	3600.87	5327	3600.62	5335	3600.872	5327	3600.62	5335
Average				**1892.32**	**5249.1**	**1956.34**	**5248.7**	**1455.45**	**5249**	**1956.338**	**5248.7**	**1455.45**	**5249**
min				**150.356**	**5018**	**168.673**	**5018**	**141.351**	**5018**	**168.673**	**5018**	**141.351**	**5018**
max				**3600.7**	**5493**	**3600.89**	**5493**	**3600.92**	**5493**	**3600.887**	**5493**	**3600.92**	**5493**

7. Αποτελέσματα TS2 μοντέλου για M=5 και N=20

				with 0.999						without 0.999			
				normal		with LB		with UB		with LB		with UB	
MODEL	**M**	**N**	**instance**	**time**	**obj**	**time**	**obj**	**time**	**obj**	**time**	**obj**	**time**	**obj**
TS2	5	20	1	3601.256	1282	66.391	1278	75.547	1278	73.335	1278	75.53	1278
	5	20	2	7.594	1359	698.895	1359	3600.073	1359	699.833	1359	3600.045	1359
	5	20	3	3599.796	1083	725.489	1082	3600.051	1081	730.66	1082	3600.053	1081
	5	20	4	3599.877	1295	3600.02	1299	3600.688	1301	3600.086	1299	3600.011	1301
	5	20	5	3600.047	1235	3600.116	1235	no solution	no solution	3600.055	1235	no solution	no solution
	5	20	6	2.75	1195	31.094	1196	2.344	1195	30.954	1196	2.265	1195
	5	20	7	195.938	1234	3600.027	1239	3600.063	1239	3600.054	1239	3600.045	1239
	5	20	8	2862.219	1206	3600.025	1206	14.188	1207	3600.07	1206	14.25	1207
	5	20	9	3600.032	1230	3600.045	1230	3600.109	1230	3600.055	1230	3599.942	1230
	5	20	10	12.328	1109	9.672	1109	1289.61	1108	9.672	1109	1287.872	1108
Average				**2108.184**	***1222.8***	**1953.177**	***1223.3***	**2153.63**	***1222***	**1954.4774**	***1223.3***	**2153.335**	***1222***
min				**2.75**	***1083***	**9.672**	***1082***	**2.344**	***1081***	**9.672**	***1082***	**2.265**	***1081***
max				**3601.256**	***1359***	**3600.116**	***1359***	**3600.688**	***1359***	**3600.086**	***1359***	**3600.053**	***1359***

8. Αποτελέσματα TS2 μοντέλου για M=5 και N=50

				with 0.999						without 0.999			
				normal		with LB		with UB		with LB		with UB	
MODEL	M	N	instance	time	obj	time	obj	time	obj	time	obj	time	obj
TS2	5	50	1	3.75	2724	2.312	2.724	5	2724	3.2656	2724	4.89	2724
	5	50	2	3600.156	2839	3600.168	2838	no solution	no solution	3600.195	2838	3600.018	2842
	5	50	3	104.187	2623	100.657	2622	no solution	no solution	100.438	2622	154.87	2621
	5	50	4	414.828	2751	616.629	2753	no solution	no solution	618.457	2753	152.994	2753
	5	50	5	343.672	2863	368.893	2863	no solution	no solution	364.596	2863	228.601	2863
	5	50	6	14.593	2831	162.595	2831	no solution	no solution	161.361	2831	49.061	2829
	5	50	7	474.703	2725	86.125	2726	no solution	no solution	86.032	2726	690.254	2726
	5	50	8	80.672	2684	14.062	2683	no solution	no solution	14.062	2683	73.326	2684
	5	50	9	481.203	2554	1143.039	2552	no solution	no solution	1135.476	2552	3600.018	2552
	5	50	10	25.703	2782	7.578	2782	no solution	no solution	7.547	2782	4.421	2782
Average				**554.3467**	***2737.6***	**610.2058**	***2465.3***			**609.14296**	***2737.4***	**855.8453**	***2737.6***
min				**3.75**	***2554***	**2.312**	***2.724***			**3.2656**	***2552***	**4.421**	***2552***
max				**3600.156**	***2863***	**3600.168**	***2863***			**3600.195**	***2863***	**3600.018**	***2863***

9. Αποτελέσματα TS2 μοντέλου για M=5 και N=100

				with 0.999						without 0.999			
				normal		with LB		with UB		with LB		with UB	
MODEL	M	N	instance	time	obj	time	obj	time	obj	time	obj	time	obj
TS2	5	100	1	186.672	5493	323.627	5495	no solution	no solution	322.831	5495	141.932	5493
	5	100	2	3600.657	5296	3600.679	5276	no solution	no solution	3600.71	5276	3601.048	5283
	5	100	3	3600.657	5183	3600.66	5186	no solution	no solution	3600.71	5186	3539.017	5178
	5	100	4	3600.671	5018	3600.648	5021	no solution	no solution	3600.727	5021	3600.702	5022
	5	100	5	1624.969	5252	736.724	5250	no solution	no solution	734.552	5250	1384.205	5255
	5	100	6	709.477	5136	2724.607	5140	no solution	no solution	2705.345	5140	825.682	5135
	5	100	7	457.92	5246	658.578	5249	no solution	no solution	655.394	5249	679.634	5251
	5	100	8	3600.704	5094	3600.641	5101	no solution	no solution	3600.695	5101	3600.651	5100
	5	100	9	3600.679	5468	3600.685	5460	no solution	no solution	3600.726	5460	3600.632	5465
	5	100	10	3600.679	5335	3600.672	5331	no solution	no solution	no	no	3600.656	5332
Average				**2458.309**	***5252.1***	**2604.752**	***5250.9***			**2491.29889**	***5242***	**2457.416**	***5251.4***
min				**186.672**	***5018***	**323.627**	***5021***			**322.831**	***5021***	**141.932**	***5022***
max				**3600.704**	***5493***	**3600.685**	***5495***			**3600.727**	***5495***	**3601.048**	***5493***

10. Αποτελέσματα Wilson μοντέλου για M=10 και N= 20 ή 50 ή 100

with 0.999													
MODEL	**M**	**N**	**instance**	**time**	**obj**	**M**	**N**	**time**	**obj**	**M**	**N**	**time**	**obj**
wilson	**10**	**20**	1	3600.118	1588	**10**	**50**	3600.282	3090	**10**	**100**	3601.055	5901
	10	20	2	3600.062	1648	10	50	3600.121	2926	10	100	3601.072	5515
	10	20	3	3599.92	1459	10	50	3600.224	2916	10	100	3601.102	5784
	10	20	4	3600	1,365	10	50	3600.212	3188	10	100	3601.103	6104
	10	20	5	3600.017	1466	10	50	3600.228	3074	10	100	3601.087	5736
	10	20	6	3600.016	1353	10	50	3600.228	3067	10	100	3601.134	5526
	10	20	7	3600.011	1416	10	50	3600.272	3136	10	100	3601.071	5767
	10	20	8	3599.973	1532	10	50	3600.228	3092	10	100	3601.119	5782
	10	20	9	3600.017	1593	10	50	3600.212	2992	10	100	3601.055	5979
	10	20	10	3599.999	1584	10	50	3600.243	3125	10	100	3601.072	5879
Average				**3600.013**	**1500.4**			**3600.225**	**3060.6**			**3601.087**	**5797.3**
min				**3599.92**	**1353**			**3600.121**	**2916**			**3601.055**	**5515**
max				**3600.118**	**1648**			**3600.282**	**3188**			**3601.134**	**6104**

11. Αποτελέσματα Wilson μοντέλου για M=20 και N= 20 ή 50 ή 100

with 0.999													
MODEL	**M**	**N**	**instance**	**time**	**obj**	**M**	**N**	**time**	**obj**	**M**	**N**	**time**	**obj**
wilson	**20**	**20**	1	3600.063	2229	**20**	**50**	no exist	no exist	**20**	**100**	no solution	no solution
	20	20	2	3599.997	2159	20	50	3600.628	4404	20	100	no solution	no solution
	20	20	3	3600.056	2341	20	50	no exist	no exist	20	100	no solution	no solution
	20	20	4	3600.04	2265	20	50	3600.571	4122	20	100	no solution	no solution
	20	20	5	3600.041	2159	20	50	3600.571	4000	20	100	no solution	no solution
	20	20	6	3600.056	2309	20	50	3600.705	3862	20	100	no solution	no solution
	20	20	7	3600.129	2315	20	50	3600.619	4049	20	100	no solution	no solution
	20	20	8	3600.079	2263	20	50	3600.587	4032	20	100	no solution	no solution
	20	20	9	3600.063	2311	20	50	3600.618	3927	20	100	no solution	no solution
	20	20	10	3600.079	2258	20	50	3600.587	3986	20	100	3602.79	7534
Average				**3600.06**	**2260.9**			**3600.611**	**4047.75**			**no solution**	**no solution**
min				**3599.997**	**2159**			**3600.571**	**3862**				
max				**3600.129**	**2341**			**3600.705**	**4404**				

12. Αποτελέσματα WST μοντέλου για M=10 και N= 20 ή 50 ή 100

with 0.999													
MODEL	**M**	**N**	**instance**	**time**	**obj**	**M**	**N**	**time**	**obj**	**M**	**N**	**time**	**obj**
wst	10	20	1	3600.223	1609	10	50	3600.29	3160	10	100	3601.259	5927
	10	20	2	3600.046	1696	10	50	3600.274	3110	10	100	3601.212	5522
	10	20	3	3600.517	1525	10	50	3600.227	3026	10	100	3601.259	5890
	10	20	4	3600.641	1415	10	50	3600.337	3222	10	100	3601.227	6161
	10	20	5	3600.234	1447	10	50	3600.442	3146	10	100	3601.259	5679
	10	20	6	3600.266	1440	10	50	3600.289	3172	10	100	3601.244	5518
	10	20	7	3600.703	1491	10	50	3600.368	3244	10	100	3601.259	6004
	10	20	8	3600.714	1577	10	50	3600.291	3162	10	100	3601.227	6041
	10	20	9	1532.938	1594	10	50	3600.274	3067	10	100	3601.243	6295
	10	20	10	3601.89	1634	10	50	3600.306	3246	10	100	3601.275	5949
Average				**3393.817**	**1542.8**			**3600.31**	**3155.5**			**3601.246**	**5898.6**
min				**1532.938**	**1415**			**3600.227**	**3026**			**3601.212**	**5518**
max				**3601.89**	**1696**			**3600.442**	**3246**			**3601.275**	**6295**

13. Αποτελέσματα WST μοντέλου για M=20 και N= 20 ή 50 ή 100

with 0.999												
MODEL	**M**	**N**	**instance**	**time**	**obj**	**M**	**N**	**time**	**obj**	**M**	**N**	**time**	**obj**
wst	20	20	1	3600.329	2364	20	50	3600.665	4181	20	100	no solution	no solution
	20	20	2	3600.118	2190	20	50	3600.587	4017	20	100	no solution	no solution
	20	20	3	3600.181	2413	20	50	3600.649	4081	20	100	3603.696	7566
	20	20	4	3600.087	2302	20	50	3601.335	4041	20	100	no solution	no solution
	20	20	5	3600.087	2190	20	50	3600.67	3927	20	100	no solution	no solution
	20	20	6	3600.087	2308	20	50	3600.65	4080	20	100	no solution	no solution
	20	20	7	3600.103	2342	20	50	3600.666	4062	20	100	no solution	no solution
	20	20	8	3600.072	2248	20	50	3600.712	4071	20	100	no solution	no solution
	20	20	9	3600.087	2354	20	50	3600.806	4083	20	100	no solution	no solution
	20	20	10	3600.273	2250	20	50	3600.756	4159	20	100	no solution	no solution
Average				**3600.142**	**2296.1**			**3600.75**	**4070.2**				
min				**3600.072**	**2190**			**3600.587**	**3927**				
max				**3600.329**	**2413**			**3601.335**	**4181**				

14. Αποτελέσματα TS2 μοντέλου για M=10 ή 20 και N= 20 ή 50 ή 100

with 0.999													
MODEL	**M**	**N**	**instance**	**time**	**obj**	**M**	**N**	**time**	**obj**	**M**	**N**	**time**	**obj**
TS2	10	20	1	3600.086	1607	10	50	3600.339	3156	10	100	3601.351	6018
	10	20	2	3600.085	1702	10	50	3600.346	3058	10	100	3601.351	5624
	10	20	3	3600.101	1543	10	50	3600.351	3026	10	100	3601.32	5807
	10	20	4	3600.086	1406	10	50	3600.352	3199	10	100	3601.383	6057
	10	20	5	3600.086	1479	10	50	3600.332	3140	10	100	3601.351	5760
Average				**3600.089**	**1547.4**			**3600.344**	**3115.8**			**3601.351**	**5853.2**
min				**3600.085**	**1406**			**3600.332**	**3026**			**3601.32**	**5624**
max				**3600.101**	**1702**			**3600.352**	**3199**			**3601.383**	**6057**

with 0.999													
MODEL	**M**	**N**	**instance**	**time**	**obj**	**M**	**N**	**time**	**obj**	**M**	**N**	**time**	**obj**
TS2	20	20	1	3600.111	2354	20	50	3600.773	4223	20	100	no solution	no solution
	20	20	2	3600.148	2184	20	50	3600.789	4057	20	100	no solution	no solution
	20	20	3	3600.148	2392	20	50	3600.789	4102	20	100	3603.445	8026
Average				**3600.136**	**2310**			**3600.784**	**4127.333**				
min				**3600.111**	**2184**			**3600.773**	**4057**				
max				**3600.148**	**2392**			**3600.789**	**4223**				

ΒΙΒΛΙΟΓΡΑΦΙΑ

1. EF STAFFORD JR, FT TSENG and JND GUPTA, "Comparative evaluation of MILP flowshop models", *Journal of the Operational Research Society* **56**, 88-101 ,2005
2. S.REZA HEJAKI and SAGHAFIAAN, "Flowshop-scheduling with makespan criterion: a review", *International Journal of Production Research*, **43**, 14, 2005
3. KENNETH R.BAKER, *"Introduction to Sequencing and Scheduling"*, John Willey & Sons, New York, 1974
4. THIMAS E.MORTON & DAVID W.PENTICO, *"Heuristic Scheduling Systems"*, John Willey & Sons, New York, 1993
5. RICHARD W.CONWAY, WILLIAM L.MAXWELL, LOUIS W.MILLER, *"Theory of Scheduling"*, Dover Publications, New York, 1967
6. CHRISTODOULOS A. FLOUDAS, *"Nonlinear and Mixed-Integer Optimization"*, Oxford University Press, New York, 1995
7. ΒΑΣΙΛΗΣ Σ. ΚΟΥΙΚΟΓΛΟΥ, *«Προγραμματισμός Παραγωγής»*, Σημειώσεις Μεταπτυχιακού Μαθήματος, Πολυτεχνείο Κρήτης, Χανιά, 2002
8. ΑΘΑΝΑΣΙΟΣ ΜΥΓΔΑΛΑΣ, ΙΩΑΝΝΗΣ ΜΑΡΙΝΑΚΗΣ, *«Συνδυαστική Βελτιστοποίηση»*, σημειώσεις μαθήματος, Πολυτεχνείο Κρήτης, Χανιά, 2002
9.ΓΙΑΝΝΗΣ Α. ΦΙΛΗΣ, *«Συστήματα Παραγωγής (Αποθέματα, Πρόβλεψη, Προγραμματισμός»*, Πολυτεχνείο Κρήτης, Χανιά, 2001
10. ILOG S.A (2001) *ILOG OPL 3.5 User's Manual*,
11. ILOG S. A (2002), *ILOG CPLEX 8.0 User's Manual*