ΕΘΝΙΚΟ ΚΑΙ ΚΑΠΟΔΙΣΤΡΙΑΚΟ ΠΑΝΕΠΙΣΤΗΜΙΟ ΑΘΗΝΩΝ

ΣΧΟΛΗ ΘΕΤΙΚΩΝ ΕΠΙΣΤΗΜΩΝ
ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΤΗΛΕΠΙΚΟΙΝΩΝΙΩΝ

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

ΧΡΟΝΟΔΡΟΜΟΛΟΓΗΣΗ ΙΠΤΑΜΕΝΟΥ ΠΡΟΣΩΠΙΚΟΥ ΑΕΡΟΠΟΡΙΚΩΝ ΕΤΑΙΡΙΩΝ

Ιωάννης Γ. Κοψιδάς

Επιβλέπων καθηγητής: Παναγιώτης Σταματόπουλος, Επίκουρος καθηγητής

ΑΘΗΝΑ

ΟΚΤΩΒΡΙΟΣ 2015

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

Χρονοδρομολόγηση Ιπτάμενου Προσωπικού Αεροπορικών Εταιριών

Όνομα Π. Επώνυμο
Α.Μ.: 1116200800042

ΕΠΙΒΛΕΠΩΝ: **Παναγιώτης Σταματόπουλος,** Επίκουρος καθηγητής

Περίληψη

Σκοπός της παρούσας πτυχιακής εργασίας είναι η δημιουργία αλγορίθμου που εξετάζει και υλοποιεί την ανάθεση πληρωμάτων (crew assignment) - συνδυασμών πτήσεων (pairings) σε πιλότους - με περιορισμούς που ενεργοποιούμε εμείς σε κάθε εκτέλεση και τον έλεγχο της εγκυρότητας ενός σεναρίου αναθέσεων συνδυασμών πτήσεων σε πιλότους. Το πρόβλημα επιλύεται με την γλώσσα προγραμματισμού C++ και ενεργοποιεί τους περιορισμούς μέσω ορισμάτων από την γραμμή εντολών για να είναι εύχρηστο και φιλικό προς τον χρήστη. Τα αποτελέσματα δείχνουν ότι ο αλγόριθμος είναι σε θέση να αναθέσει αποτελεσματικά τους συνδυασμούς πτήσεων και να ελέγξει την εγκυρότητα των υπαρχουσών αναθέσεων.

Αφού υλοποιηθεί η μέθοδος γίνονται προσομοιώσεις για να αξιολογηθεί τελικά η ικανότητα της μεθόδου να αναθέτει και να ελέγχει τα ταιριάσματα πτήσεων ανάλογα με τους περιορισμούς που ενεργοποιούμε, την δίκαιη ανάθεση στους πιλότους, αλλά και την ταχύτητα με την οποία το κάνει αναθέσεις. Τέλος παρουσιάζονται τα αποτελέσματα και βγαίνει ένα τελικό συμπέρασμα.

Το πρόγραμμα είναι φιλικό στον χρήστη και δίνει την δυνατότητα να ενεργοποιεί όσους περιορισμούς χρειάζεται πολύ εύκολα με την χρήση ορισμάτων από την γραμμή εντολών. Σημαντικό στοιχείο είναι η υλοποίηση του αλγορίθμου με τρόπο που διευκολύνει την αλλαγή των υπαρχόντων περιορισμών και την υλοποίηση επιπλέον περιορισμών ώστε να επεκταθεί χωρίς πρόβλημα η προσομοίωση με το σκεπτικό ότι το περιβάλλον της κάθε αεροπορικής εταιρείας μπορεί να αλλάζει με δυναμικό και απρόβλεπτο τρόπο.

Λέξεις Κλειδιά: ανάθεση πληρωμάτων, έλεγχος εγκυρότητας, προγραμματισμός με περιορισμούς, C++

ΠΕΡΙΕΧΟΜΕΝΑ

1. Εισαγωγή

1.1 ΓΕΝΙΚΑ

Η αποτελεσματική λειτουργία μιας αεροπορικής εταιρίας, απαιτεί τον κατάλληλο προγραμματισμό και τις απαραίτητες μεθοδολογίες, που χρησιμεύουν σε κάθε μία από τις ιδιαίτερα πολύπλοκες και πολυδιάστατες δραστηριότητες μιας τέτοιας επιχείρησης. Η χρήση της επιχειρησιακής έρευνας, και των μοντέλων βελτιστοποίησης ειδικότερα, έχει γίνει απαραίτητη στο χώρο των εναέριων μεταφορών, λόγω της πολυπλοκότητας διαχείρισης συστημάτων τέτοιας μεγάλης κλίμακας. Η επιχειρησιακή έρευνα έχει δημιουργήσει ένα σοβαρό αντίκτυπο στον τρόπο με τον οποίο οι σημερινές αεροπορικές εταιρίες διοικούνται και ανταγωνίζονται η μια την άλλη. Οδηγούμενες από την επιτακτική ανάγκη των διοικήσεων για κατάκτηση ανταγωνιστικού μεριδίου αγοράς, οι αεροπορικές εταιρίες στρέφονται σε ανεπτυγμένες τεχνικές βελτιστοποίησης, ώστε να δημιουργήσουν τα κατάλληλα συστήματα υποστήριξης αποφάσεων, για τον καλύτερο έλεγχο και λειτουργία της “αεροπορικής” επιχείρησης και των διαδικασιών της. Πριν από εβδομήντα χρόνια, η αεροπορική βιομηχανία δημιουργήθηκε ως μέσο για την γρηγορότερη παράδοση ταχυδρομικού υλικού. Από τότε η βιομηχανία αυτή έχει εξελιχθεί σε μια πολύ μεγάλη και περίπλοκη επιχείρηση, για τη μεταφορά επιβατών και εμπορευμάτων. Το 1998, οι κορυφαίες 100 αερογραμμές μετέφεραν συνολικά παγκοσμίως, σχεδόν 1,3 δισεκατομμύρια επιβάτες, με ένα μέγεθος στόλου περίπου 9.800 αεροπλάνων, ενώ 1.5 εκατομμύριο άτομα απασχολούνταν επαγγελματικά σε αυτές και συμπλήρωναν ένα εισόδημα που έφτανε τα 287 δισεκατομμύρια δολάρια με τα λειτουργικά κέρδη να πλησιάζουν τα 18 δισεκατομμύρια δολάρια. Μαζί με το αυξανόμενο μέγεθος επήλθε και μια αυξανόμενη λειτουργική πολυπλοκότητα, που επιδεινώνεται από τον ανταγωνισμό ανάμεσα στους επιχειρηματικούς κύκλους.

1.1.1. Το προϊόν και το περιβάλλον λειτουργίας των αεροπορικών εταιριών

Το αντικείμενο μιας αεροπορικής εταιρίας είναι ξεχωριστό. Το προϊόν που προσφέρεται από τις αεροπορικές εταιρίες αντιπροσωπεύεται από πτήσεις που μεταφέρουν επιβάτες ή εμπορεύματα από διάφορες τοποθεσίες σε διαφορετικούς προορισμούς. Η εμπορευσιμότητα και επιτυχία συνάμα αυτού του προϊόντος εξαρτάται σε μεγάλο βαθμό από τους εξής παράγοντες:

· Ακρίβεια

· Επικαιρότητα

· Λειτουργικότητα

· Ποιότητα

· Κόστος εξυπηρέτησης

· Εύκολη πρόσβαση στο προϊόν

Τα παραπάνω κριτήρια γίνονται αντιληπτά από τον πελάτη επιβάτη ως, ευέλικτα ωράρια πτήσεων, ακριβής αναχώρηση πτήσεων δίχως καθυστερήσεις, ικανοποιητική εξυπηρέτηση εν ώρα πτήσης, κατάλληλη μεταχείριση των αποσκευών και προσιτός-εύκολος τρόπος έκδοσης εισιτηρίων. Για να ανταπεξέλθουν επιτυχώς στις παραπάνω απαιτήσεις και για να προσφέρουν ένα υψηλής ποιότητας αλλά και χαμηλού κόστους

προϊόν, οι αεροπορικές εταιρίες επενδύουν τεράστια υλικά αλλά και ψυχικά / πνευματικά αποθέματα στην προσπάθεια να εξελίξουν οικονομικά αποδοτικές μεθόδους αντιμετώπισης των κυριότερων προβλημάτων τους, όπως είναι κατηγοριοποίηση των τιμών των εισιτηρίων, ο προγραμματισμός των πτήσεων, σχεδιασμός δικτύου πτήσεων, ακολουθία των πτήσεων, ταίριασμα πληρωμάτων, ανάθεση πυλών αναχώρησης και άφιξης ανά αεροδρόμιο, πρόγραμμα συντήρησης αεροσκαφών, προγράμματα εκπαίδευσης πληρωμάτων καθώς και διαδικασίες διαχείρισης των αποσκευών.

Η πολυπλοκότητα που διέπει τα παραπάνω προβλήματα προέρχεται από το ίδιο το περιβάλλον στο οποίο λειτουργεί μια αεροπορική, πιο αναλυτικά:

· Το τεράστιο επιχειρηματικό μέγεθος όσο αφορά, το γεωγραφικό εύρος στο οποίο εκτείνεται η επιχείρηση, τα αποθέματα που υποχρεωτικά πρέπει να υπάρχουν στην διάθεση της επιχείρησης (μηχανήματα, προσωπικό κτλ.), το μέγεθος και η ποικιλομορφία της αγοράς στην οποία απευθύνεται η επιχείρηση. Το εξαιρετικά δυναμικό περιβάλλον μέσα στο οποίο κινείται μια τέτοιου είδους επιχείρηση το οποίο εναλλάσσεται διαρκώς. Δεν είναι λίγες οι φορές που η εταιρία καλείται να αλλάξει τα πρωταρχικά της πλάνα, είτε λόγω καιρού, είτε λόγω μηχανικών προβλημάτων των αεροσκαφών, είτε λόγω ασθένειας του πληρώματος, είτε ακόμα και από τις διακυμάνσεις για παράδειγμα στην τιμή του πετρελαίου, ή μιας απρόσμενης τρομοκρατικής ενέργειας που μπορεί να επηρεάσει το μέγεθος της αγοράς (παράδειγμα η 11 η Σεπτεμβρίου).

· Οι νομικοί και άλλοι περιορισμοί που διέπουν τις εναέριες μεταφορές και μπορεί να σχετίζονται με την συντήρηση-κατάσταση των αεροσκαφών ή την νομιμότητα όσο αφορά την εκπαίδευση του πληρώματος και την ετοιμότητα του, καθώς και τα μέτρα ασφαλείας που πρέπει να λαμβάνονται.

·Τα στενά οικονομικά όρια που τίθενται στον προγραμματισμό και σχεδιασμό της εταιρίας, ώστε να εκμεταλλευτούν όλοι οι δυνατοί πόροι στη μέγιστη αποδοτικότητα τους, δεν αφήνουν περιθώρια ελιγμών ή λάθους στα συστήματα οργάνωσης και διοίκησης που αφορούν την επιχείρηση.

1.1.2 Η επιχειρησιακή έρευνα στον χώρο τον αεροπορικών εταιριών

Για να διατηρήσουν μια σταθερή αποδοτικότητα, οι αεροπορικές εταιρίες, πρέπει να αντιλαμβάνονται άμεσα τις παρούσες συνθήκες της αγοράς προκειμένου να ισορροπήσουν την προσφορά με τη ζήτηση. Λαμβάνοντας λοιπόν υπόψη την γεωγραφική διασπορά της επιχείρησής τους, των υψηλών επενδυτικών δαπανών (τόσο σε χρήμα όσο και σε εργασία) και των αναμενόμενων εμποδίων που απορρέουν από την διεθνή αεροπλοΐα, δεν είναι καθόλου παράξενο, το ότι η αεροπορική βιομηχανία, έχει οδηγήσει άλλες επιστήμες, όπως η επιχειρησιακή έρευνα, στην εφαρμογή ειδικών τεχνικών βελτιστοποίησης για να αντιμετωπίσει τα επιχειρησιακά ζητήματα. Από τεχνολογικής άποψης, η ταχύτητα των επεξεργαστών Η/Υ είναι πλέον ικανοποιητική για την επίλυση πολύπλοκων και μεγάλης κλίμακας προβλημάτων σε πραγματικό χρόνο, η χωρητικότητα των αποθηκευτικών συσκευών των Η/Υ είναι αρκετά μεγάλη για να φιλοξενήσει επιχειρησιακά δεδομένα, το διαδίκτυο είναι αρκετά αξιόπιστο για να μεταφέρει με διάφορους τρόπους ταχύτατα τις απαραίτητες πληροφορίες και δεδομένα στους άμεσα ενδιαφερόμενους. Παράλληλα το κόστος όλου αυτού του απαραίτητου τεχνολογικού εξοπλισμού, είναι αρκετά χαμηλό, που να εξασφαλίζει την άμεση απόσβεση του. Από μεθοδολογικής άποψης, τις τελευταίες δύο δεκαετίες έχει υπάρξει μια θεαματική ανάπτυξη όσο αφορά τους αλγόριθμους βελτιστοποίησης και τις τεχνικές επίλυσης προβλημάτων. Πλέον τα μοντέλα βελτιστοποίησης έχουν έρθει πιο κοντά στην

πραγματικότητα και παρόλα αυτά είναι επαρκώς επιλύσιμα. Ο χρόνος επίλυσης προβλημάτων μεγάλης κλίμακας και πολυπλοκότητας έχει γίνει κατά πολύ μικρότερος, με αποτέλεσμα τα προβλήματα αυτά να μπορούν να επιλυθούν και από μικρότερα σε ισχύ συστήματα και να προσφέρουν στους χρήστες τους άμεσα και έγκυρα αποτελέσματα.

Έτσι λοιπόν γίνεται αντιληπτό ότι, οι μέθοδοι της επιχειρησιακής έρευνας που περιλαμβάνουν μεθόδους βελτιστοποίησης, τεχνικές επίλυσης και συστήματα υποστήριξης αποφάσεων έχουν γίνει σημαντικό εργαλείο των αεροπορικών στην προσπάθεια τους να διατηρήσουν αλλά και να επεκτείνουν την θέση τους στην άκρως ανταγωνιστική αγορά των μεταφορών.

Μερικές από τις δραστηριότητες-λειτουργίες των αεροπορικών, για τις οποίες τα τελευταία χρόνια έχουν εξελιχθεί μέθοδοι επίλυσης και βελτιστοποίησης από τον χώρο της επιχειρησιακής έρευνας είναι οι παρακάτω:

· Σχεδιασμός δικτύου και κατασκευή προγράμματος πτήσεων

· Ανάθεση αεροσκαφών σε πτήσεις

· Ακολουθίες πτήσεων που ανατίθενται σε αεροσκάφη (δρομολόγηση αεροσκαφών)

· Βελτιστοποίηση ταιριάσματος πληρωμάτων

· Ανάθεση πληρωμάτων σε πτήσεις

· Διαχείριση εισροών

· Διαχείριση μη προγραμματισμένων ενεργειών

· Διαχείριση εναέριας κυκλοφορίας

Στο επόμενο κεφάλαιο ακολουθεί μια ανάλυση των παραπάνω προβλημάτων, καθώς και μια συνοπτική αναφορά των μεθόδων που έχουν χρησιμοποιηθεί από τον κλάδο της επιχειρησιακής έρευνας για την επίλυση τους.

1.1.3. Ανάλυση των προβλημάτων των αεροπορικών εταιριών και μέθοδοι επίλυσης τους

α) Σχεδιασμός δικτύου και κατασκευή προγράμματος πτήσεων.

Η κατασκευή του προγράμματος πτήσεων ανά προορισμό καθώς και η κατασκευή του δικτύου που θα περιλαμβάνει το σύνολο των προορισμών που εξυπηρετεί μια αεροπορική εταιρία, είναι πρωταρχικής σημασίας για την σωστή και κερδοφόρα λειτουργία της. Το πρόγραμμα των πτήσεων συγκεκριμένα προσδιορίζει ένα σύνολο από ανταποκρινόμενους σταθμούς αναχώρησης και άφιξης τους οποίους εξυπηρετεί η κάθε αεροπορική εταιρία καθώς και τις ακριβείς ώρες αναχώρησης και άφιξης για κάθε συνδυασμό αυτών. Η απόφαση της αεροπορικής για το ποιους σταθμούς θα εξυπηρετεί, εξαρτάται σε μεγάλο βαθμό από τις απαιτήσεις της εκάστοτε αγοράς, τη διαθεσιμότητα και τα χαρακτηριστικά του στόλου της, το διαθέσιμο ανθρώπινο δυναμικό αλλά και από την συμπεριφορά του ανταγωνισμού. Πριν όμως από την δημιουργία συγκεκριμένου προγράμματος πτήσεων, η αεροπορική εταιρία καλείται να καθορίσει το δίκτυο της, το δίκτυο δηλαδή των επίγειων σταθμών ανάμεσα στους οποίους θα κινείται. Οι αλλαγές στις απαιτήσεις της αγοράς καθώς και οι αλλαγές στα νομικά πλαίσια που διέπουν την αεροπλοΐα και έχουν επιφέρει την απελευθέρωση στην επιλογή προορισμών για κάθε αεροπορική, έχουν αλλάξει τον τρόπο σχεδιασμού των δικτύων πτήσεων, από το πρότυπο «από σημείο σε σημείο» που επικρατούσε παλαιότερα και περιλάμβανε ουσιαστικά ζεύγη ανταποκρινόμενων σταθμών, στο πρότυπο ενός κεντρικού σταθμού και διαφόρων περιφερειακών σταθμών που βρίσκονται σε μια ακτίνα

γύρω από τον κεντρικό (πρότυπο Hub and Spoke), ανάλογα με την εμβέλεια των δυνατοτήτων της αεροπορικής εταιρίας.

Διάφορες έρευνες έχουν γίνει για την βελτιστοποίηση όχι μόνο του δικτύου πτήσεων αλλά και διαφόρων υπό-προβλημάτων που σχετίζονται με αυτό, όπως η εύρεση κατάλληλης τοποθεσίας για τον κεντρικό σταθμό και η συχνότητα των πτήσεων που καλύπτουν το δίκτυο. Για την επιλογή του κεντρικού σταθμού έχουν χρησιμοποιηθεί μέθοδοι διακλάδωσης και φραγμών, καθώς και μαθηματικού προγραμματισμού. Για τον προσδιορισμό βέλτιστης συχνότητας πτήσεων έχουν προταθεί μέθοδοι όπως η θεωρία ακαθόριστων συνόλων.

β) Ανάθεση αεροσκαφών σε πτήσεις (Fleet Assignment).

Εφόσον μια αεροπορική εταιρία έχει καθορίσει το δίκτυο και το πρόγραμμα των πτήσεων της, καλείται πλέον να αναθέσει τα αεροσκάφη του στόλου της, στις διάφορες πτήσεις που συνθέτουν το δίκτυο της. Συνήθως οι αεροπορικές εταιρίες έχουν στην κατοχή τους διαφορετικού τύπου αεροσκάφη, με διαφορετικά χαρακτηριστικά και κόστη, όπως χωρητικότητα όσο αφορά τις διαθέσιμες θέσεις, κατανάλωση καυσίμων, και επιτρεπόμενο φορτωτικό βάρος. Επιδιώκεται όπως είναι αναμενόμενο η βέλτιστη ανάθεση των αεροσκαφών, λαμβάνοντας αυστηρά υπόψη τόσο τους παραπάνω φυσικούς περιορισμούς αλλά και την ποικιλομορφία της αγοράς και τις λοιπές επιχειρηματικές επιδιώξεις.

Ο προγραμματισμός ανάθεσης των αεροσκαφών γίνεται στην πραγματικότητα μήνες πριν την πρακτική εφαρμογή του, καθώς δίδεται έτσι το περιθώριο αναθεώρησης και βελτίωσης του αποτελέσματος κάτω από τις πραγματικές υπάρχουσες συνθήκες. Σημαντική παρατήρηση αποτελεί ότι κατά το σχεδιασμό ανάθεσης πτήσεων σε αεροσκάφη, γίνεται η υπόθεση ότι το πρόγραμμα των πτήσεων παραμένει σταθερό και επαναλαμβάνεται περιοδικά (ανά μέρα ή εβδομάδα συνήθως), παρόλο που κάτι τέτοιο μπορεί να μην συμβαίνει στην πραγματικότητα (π.χ. απρογραμμάτιστες πτήσεις charter).

Η πλειονότητα των μεθόδων που έχουν χρησιμοποιηθεί για την βελτιστοποίηση και επίλυση του προβλήματος ανάθεσης αεροσκαφών σε πτήσεις, αναπαριστούν το πρόγραμμα των πτήσεων ως χωροχρονικό δίκτυο ροής, με τόξα ανάμεσα στους διάφορους σταθμούς και μια λίστα πιθανών πτήσεων (τόξων) σε κάθε σταθμό. Οι διάφορες αναχωρήσεις και αφίξεις σε κάθε σταθμό αναπαριστώνται ως κόμβοι. Ταυτόχρονα υπάρχουν και τόξα που αναπαριστούν την επίγεια παραμονή του αεροσκάφους μέχρι την ώρα της επόμενης αναχώρησης του. Κάθε ένας συνδυασμός κόμβων και τόξων θα πρέπει να ανατίθεται σε ένα και μόνο αεροσκάφος, ώστε να ικανοποιείται ο λογικός περιορισμός που απαιτεί κάθε πτήση να εκτελείται μόνο μια φορά και από ένα αεροσκάφος. Άλλοι περιορισμοί που λαμβάνονται συνήθως υπόψη, είναι ο αριθμός των διαθέσιμων αεροσκαφών αλλά και των χαρακτηριστικών τους (π.χ. πτητική εμβέλεια, χωρητικότητα). Το μοντέλο αυτό καταλήγει να επιλύεται συνήθως με μεθόδους μικτού, ακέραιου προγραμματισμού.

Ένας άλλος τρόπος μοντελοποίησης του προβλήματος, πολύ πιο αναλυτικός είναι με την δημιουργία ενός δικτύου συνδέσεων, όπου οι κόμβοι είναι οι πτήσεις και τα τόξα η σύνδεση μεταξύ των πτήσεων. Μιας και οι πιθανοί τρόποι σύνδεσης των πτήσεων μεταξύ τους είναι πολλοί, συχνά οδηγούμαστε σε μεγαλύτερης κλίμακας δίκτυα απ' ότι στην περίπτωση του χωροχρονικού δικτύου. Από την άλλη μεριά όμως το δεύτερο μοντέλο είναι πολύ πιο ακριβές και κοντά στο πραγματικό δίκτυο πτήσεων.

γ) Δρομολόγηση αεροσκαφών (Aircraft Routing).

Το πρόβλημα δρομολόγησης αεροσκαφών, είναι ουσιαστικά το πρόβλημα δημιουργίας ενός δρομολογίου για κάθε αεροσκάφος του στόλου της αεροπορικής εταιρίας, δεδομένου του συνόλου των πτήσεων που πρέπει να πραγματοποιήσει η εταιρία, έχοντας ως στόχο την ελαχιστοποίηση του κόστους. Ένας δεύτερος τρόπος

μοντελοποίησης του προβλήματος είναι έμμεσος και προσπαθεί να μεγιστοποιήσει έμμεσες αξίες όπως την ικανοποίηση της επιθυμίας των πελατών για περισσότερες πτήσεις χωρίς ενδιάμεσους σταθμούς, μεταξύ διαφόρων πόλεων.

Τα δημιουργούμενα δρομολόγια θα πρέπει βέβαια να υπακούν σε κάποιους περιορισμούς. Δυο τυπικοί περιορισμοί στην κατασκευή δρομολογίων είναι ο περιορισμός συντήρησης των αεροσκαφών και ο περιορισμός εξισορρόπησης της διανομής των πτήσεων στα διάφορα αεροσκάφη. Ο περιορισμός συντήρησης, αφορά φυσικά τους τεχνικούς ελέγχους και επισκευές που θα πρέπει να γίνονται ανά τακτά χρονικά διαστήματα, στα αεροσκάφη. Ο χρόνος που θα πετάει ένα αεροσκάφος συνεχόμενα χωρίς συντήρηση είναι περιορισμένος και πρέπει να υπακούει στους κανόνες ασφαλείας. Υπάρχουν πολλών ειδών συντηρήσεις που γίνονται σε αεροσκάφη, υπάρχουν έλεγχοι που κρατάνε λίγες ώρες όπως επίσης και επισκευές που μπορεί να διαρκέσουν και αρκετές μέρες, συνήθως όμως τέτοιου είδους μακροχρόνιες επισκευές δεν λαμβάνονται υπόψη κατά την μοντελοποίηση και επίλυση του προβλήματος, αφού είναι πιο σπάνιες.

Η σωστή διανομή των πτήσεων, εξασφαλίζει ότι όλα τα αεροσκάφη θα χρησιμοποιηθούν εξίσου κατά την λειτουργία του προγράμματος της αεροπορικής εταιρίας, ώστε να μην επιφορτίζεται μόνο μέρος του στόλου, πράγμα που οδηγεί σε αυξανόμενα κόστη συντήρησης καθώς και αυξανόμενα διαφυγόντα κέρδη εκμετάλλευσης του τεχνικού εξοπλισμού της εταιρίας.

δ) Βελτιστοποίηση ταιριάσματος πληρωμάτων (crew pairing).

Το πρόβλημα του ταιριάσματος των πληρωμάτων, έγκειται ουσιαστικά στο κατάλληλο ταίριασμα κάθε πληρώματος σε μια ακολουθία πτήσεων, που αποτελεί τμήμα του συνολικού προγράμματος πτήσεων. Επόμενο είναι λοιπόν το παρόν πρόβλημα να έπεται της λύσης του σχεδιασμού δικτύου και προγράμματος πτήσεων, αλλά και της ανάθεσης αεροσκαφών σε πτήσεις.

Η ακολουθία των πτήσεων που ανατίθεται σε κάθε πλήρωμα έχει ως αρχή και τέλος την σταθερή επίγεια βάση του πληρώματος και ο αριθμός των πτήσεων που την αποτελούν διέπεται από κάποιους περιορισμούς, τόσο νομικούς ή συμβατικούς, που έχουν να κάνουν κυρίως με τις επιτρεπόμενες ώρες συνεχούς πτήσης, αλλά και φυσικούς που έχουν να κάνουν με τον αριθμό των διαθέσιμων πληρωμάτων σε κάθε βάση. Αυτός ο αριθμός των διαθέσιμων πληρωμάτων σε κάθε βάση μεταφράζεται κατά την μοντελοποίηση σε ελάχιστο και μέγιστο αριθμό διαθέσιμων ωρών πτήσης από κάθε βάση.

Το ιδανικό θα ήταν βέβαια, κάθε πλήρωμα να ακολουθεί την πορεία ενός αεροσκάφους, έτσι ώστε σε περίπτωση καθυστερήσεων να μην υπάρχουν προβλήματα κατά την ανταλλαγή των πληρωμάτων, κάτι τέτοιο όμως είναι ιδιαίτερα σπάνιο και συμβαίνει μόνο σε μικρής εμβέλειας και δυνατότητας αεροπορικές εταιρίες. Παρόλα αυτά το ζητούμενο κατά την επίλυση του ταιριάσματος πληρωμάτων με πτήσεις, είναι ο περιορισμός στο ελάχιστο του αριθμού αλλαγών πληρωμάτων, λαμβάνοντας βέβαια υπόψη τους όποιους νομικούς και φυσικούς περιορισμούς.

Οι μέθοδοι που έχουν χρησιμοποιηθεί για την επίλυση και βελτιστοποίηση της λύσης του προβλήματος βασίζονται συνήθως σε ένα καθορισμένο πρόβλημα διαχωρισμού, κατά το οποίο οι πτήσεις που θα πρέπει να καλυφθούν εμφανίζονται σε γραμμές, ενώ σε στήλες εμφανίζονται όλα τα πιθανά ταιριάσματα που μπορούν να καλύψουν την συγκεκριμένη πτήση ικανοποιώντας τους περιορισμούς. Από το σύνολο αυτό των πιθανών ταιριασμάτων και έχοντας μια πρώτη λύση χρησιμοποιούνται μεθευρετικοί αλγόριθμοι ή μέθοδοι δημιουργίας στηλών (column generation) για την βελτίωση της υπάρχουσας λύσης. Επειδή σε μεσαίου και μεγάλου μεγέθους αεροπορικές εταιρίες, ο αριθμός των πιθανών ταιριασμάτων είναι εξαιρετικά μεγάλος, πολλές φορές λαμβάνεται υπόψη κατά την επεξεργασία τους, μόνο ένα υποσύνολο αυτών. Η επιλογή τόσο του

υποσυνόλου των πιθανών λύσεων αλλά και της πρώτης λύσης παίζει σημαντικό ρόλο στην ποιότητα της τελικής λύσης που προκύπτει από τον ευρετικό αλγόριθμο.

Ενθαρρυντικά είναι πάντως τα αποτελέσματα τέτοιου είδους εφαρμογών στο συγκεκριμένο πρόβλημα των αεροπορικών, για παράδειγμα το 1989 μεγάλη αμερικανική αεροπορική εταιρία δήλωσε πως μόνο από την εφαρμογή της παραπάνω μεθόδου στο πρόβλημα ταιριάσματος των πληρωμάτων της σε πτήσεις, είχε αύξηση των κερδών της κατά 18 εκατ. δολάρια το χρόνο.

ε) Ανάθεση πληρωμάτων σε πτήσεις (crew assignment).

Λόγω της πολυπλοκότητας προγραμματισμού των πληρωμάτων των αεροπορικών εταιριών έκτος από το ταίριασμα πληρωμάτων που αποτελεί την πρώτη φάση αυτής της διαδικασίας, γίνεται και η φάση ανάθεσης ονομαστικά πλέον, πληρωμάτων σε δρομολόγια, με τους όποιους ιδιαίτερους περιορισμούς προκύπτουν από την συγκεκριμένη αυτή αντιστοίχιση εργαζομένων στα πληρώματα που προέκυψαν από την επίλυση της πρώτης φάσης. Αρχικά γίνεται λοιπόν το ανώνυμο ταίριασμα πληρωμάτων, (ως βάρδιες για παράδειγμα) σε μια ακολουθία πτήσεων με τον τρόπο που περιγράφηκε παραπάνω, έτσι ώστε να καλύπτονται οι ανάγκες σε πλήρωμα, σε κάθε πτήση. Στη συνέχεια αυτά τα ταιριάσματα-ζεύγη πληρωμάτων με πτήσεις, σε συνδυασμό με άλλες δραστηριότητες όπως τα επίγεια καθήκοντα, τα καθήκοντα εφεδρείας σε περίπτωση ανάγκης, καθώς και οι εκτός εργασίας φραγμοί όπως άδειες ή οποιαδήποτε άλλα προσωπικά προβλήματα που περιορίζουν την δυνατότητα του εργαζομένου να παρευρεθεί στην δουλειά του συγκεκριμένες ώρες, τοποθετούνται διαδοχικά σε κάποιες λίστες και ανατίθενται σε μεμονωμένα μέλη του πληρώματος (ονομαστικά πλέον). Η ανάθεση αυτή γίνεται συνήθως για κάθε μήνα ξεχωριστά και κάθε λίστα περιέχει συνήθως κάποια ιστορικά στοιχεία (π.χ. ημερομηνίες αδειών) που βοηθούν στο προγραμματισμό επόμενων μηνών.

Και σε αυτό το πρόβλημα, σύνθετοι κανόνες και οι κανονισμοί που προέρχονται από τη νομοθεσία και τις συμβατικές συμφωνίες πρέπει να καλυφτούν από τις λύσεις στην προσπάθεια βελτίωσης κάποιας αντικειμενικής συνάρτησης που αφορά τη λειτουργία και το κέρδος φυσικά της εταιρίας.

Για την επίλυση του προβλήματος ανάθεσης πληρωμάτων, έχει προτιμηθεί αυτή να γίνεται ξεχωριστά για κάθε ειδικότητα, για παράδειγμα συνήθως οι πιλότοι έχουν συγκεκριμένη εκπαίδευση σε ένα ειδικό τύπο αεροσκάφους και έτσι θα μπορούν να πετάνε μόνο με αυτόν τον τύπο, σε αντίθεση με τους αεροσυνοδούς που συνήθως έχουν την δυνατότητα να εργαστούν σε διαφορετικού τύπου αεροσκάφη. Έτσι σε τελική ανάλυση και χρησιμοποιώντας συνήθως ακέραιο προγραμματισμό, μεγιστοποιείται η συμβολή κάθε πιθανού μέλους πληρώματος στο πρόγραμμα πτήσεων, μεγιστοποιώντας έτσι το κέρδος που μπορεί να προέλθει από αυτό.

Μια πολύ συγκεκριμένη υποπερίπτωση του προβλήματος ανάθεσης πληρωμάτων σε πτήσεις (crew assignment), λοιπόν, θα επιλύσουμε, με την χρήση ενός εκτελέσιμου προγράμματος γραμμένου στην αντικειμενοστραφής γλώσσα C++, στην παρούσα διπλωματική εργασία.

στ) Διαχείριση εισροών (Revenue Management).

Μια αεροπορική, προσφέρει συνήθως τα εισιτήρια της, που χαρακτηρίζονται από την προέλευση και τον προορισμό της πτήσης, σε διάφορες κατηγορίες τιμής. Αυτές οι κατηγορίες τιμής περιλαμβάνουν όχι μόνο τις συνήθεις κατηγορίες, διακεκριμένη (business) και οικονομική (economy), που εγκαθίστανται και σε χωριστά μέρη του αεροσκάφους, αλλά περιλαμβάνουν και κατηγορίες τιμής, για τις οποίες η διαφορά στη τιμή δικαιολογείται από τους διαφορετικούς παράγοντες όπως για π.χ. την επιλογή που προσφέρουν στον επιβάτη για ακύρωση του εισιτηρίου σε περίπτωση που αυτός δεν θέλει να το χρησιμοποιήσει, ή κάποιες ρυθμίσεις διανυκτέρευσης σε κάποιο ενδιάμεσο σταθμό που μπορεί να επιθυμεί ο πελάτης. Επομένως οι θέσεις σε μια πτήση είναι

προϊόντα που μπορούν να προσφερθούν σε διαφορετικούς τύπους πελατών για διαφορετικές τιμές. Δεδομένου ότι τα εισιτήρια για μια πτήση πρέπει να πωληθούν προτού να απογειωθεί το αεροπλάνο, το προϊόν είναι φθαρτό και η διαχείριση εισροών πρέπει να εφαρμοστεί έγκαιρα ώστε να εξασφαλιστεί κέρδος.

Σημαντικό εργαλείο της διαχείρισης εισοδήματος μιας αεροπορικής, είναι η διαχείριση του προβλήματος ελέγχου διαθέσιμων θέσεων (seat inventory control problem). Αυτό το πρόβλημα αφορά την διανομή του πεπερασμένου αριθμού καθισμάτων στη ζήτηση, που εμφανίζεται κατά τη χρονική διάρκεια πριν την αναχώρηση της πτήσης. Ο στόχος είναι να βρεθεί ο σωστός συνδυασμός τύπου επιβατών στις πτήσεις έτσι ώστε τα εισοδήματα να μεγιστοποιούνται. Η βέλτιστη κατανομή του αριθμού καθισμάτων στις διάφορες κατηγορίες πρέπει έπειτα να μεταφραστεί σε μια πολιτική ελέγχου κρατήσεων, η οποία θα αποφασίζει εάν πρέπει να γίνει αποδεκτό ένα αίτημα κράτησης όταν αυτό φθάνει στον έλεγχο. Δεν είναι περίεργο, σε ένα ορισμένο χρονικό σημείο, να είναι πιο κερδοφόρα η απόρριψη ενός αιτήματος κράτησης, προκειμένου να υπάρχει η δυνατότητα να γίνει δεκτό ένα αίτημα κράτησης ενός άλλου επιβάτη σε ένα μετέπειτα χρονικό σημείο, που να αρμόζει περισσότερο στην πολιτική διανομής θέσεων της εταιρίας.

Άλλα σημαντικά θέματα που έχουν μεγάλη σημασία στη διαχείριση εισροών είναι η πρόβλεψη ζήτησης, η τιμολόγηση των εισιτηρίων κάθε κατηγόριας και η δυνατότητα κρατήσεων θέσεων μεγαλύτερων σε αριθμό από τις διαθέσιμες θέσεις του αεροσκάφους (overbooking). Η πρόβλεψη της ζήτησης είναι κρίσιμο εργαλείο στη διαχείριση εισροών των αεροπορικών, μιας και οι πολιτικές ελέγχου κράτησης χρησιμοποιούν τις προβλέψεις ζήτησης για να καθορίσουν την βέλτιστη στρατηγική ελέγχου κράτησης. Εάν μια αεροπορική εταιρία λάβει υπόψη της χαμηλής ποιότητας εκτίμηση της ζήτησης, θα οδηγηθεί σε μια στρατηγική ελέγχου κράτησης με ελάχιστη απόδοση. Οι αεροπορικές εταιρίες έρχονται συχνά αντιμέτωπες με προβλήματα όπως, ακυρώσεις εισιτηρίων και μη εμφάνιση επιβατών στο αεροδρόμιο (no-show), για οποιουσδήποτε λόγους. Επομένως, προκειμένου να αποτραπεί η απογείωση ενός αεροσκάφους με μεγάλο αριθμό κενών καθισμάτων, οι αεροπορικές εταιρίες τείνουν να δέχονται περισσότερες κρατήσεις από τον διαθέσιμο αριθμό θέσεων σε μια πτήση. Τέλος η τιμολόγηση είναι προφανώς πολύ σημαντική διαδικασία με άμεσο αντίκτυπο στα εισοδήματα μιας αεροπορικής εταιρίας. Στην πραγματικότητα, η τιμολόγηση των διαφόρων κατηγοριών είναι η αφετηρία του προβλήματος διαχείρισης εισροών.

Όσο αφορά το πρόβλημα της μη εμφάνισης επιβατών στο αεροδρόμιο, και των κρατήσεων παραπάνω θέσεων, υποθέτοντας ότι οι αφίξεις των επιβατών είναι κατανομής Poisson, σε ένα συγκεκριμένο χρονικό διάστημα και χρησιμοποιώντας ένα δισδιάστατο στοχαστικό μοντέλο δυναμικού προγραμματισμού, μεγάλη αεροπορική εταιρία της Ευρώπης έχει αποδείξει πως μπορεί να έχει αύξηση κερδών έως και 2 εκατ. δολάρια το χρόνο.

ζ) Διαχείριση μη προγραμματισμένων ενεργειών.

Όπως έχει αναφερθεί και παραπάνω, οι αεροπορικές εταιρίες, σχεδιάζουν και εκδίδουν τελικά προς το κοινό το πρόγραμμα των πτήσεων τους. Πολλές φορές όμως, προβλήματα έλλειψης δυναμικού, όπως για παράδειγμα κάποια τεχνική βλάβη σε κάποιο από τα αεροσκάφη του στόλου της εταιρίας ή ακόμα και μια ραγδαία επιδείνωση του καιρού, αποτρέπουν τις αεροπορικές εταιρίες από την πραγματοποίηση του προγράμματος πτήσεων τους, πράγμα που μπορεί να επιφέρει μεγάλο οικονομικό κόστος στην αεροπορική τόσο άμεσα, όσο και έμμεσα με την απώλεια δυσαρεστημένων πελατών.

Όταν καταστάσεις σαν και αυτές εμφανίζονται στο προσκήνιο, απαιτείται άμεση ανταπόκριση, ώστε με τις κατάλληλες ενέργειες η αεροπορική να επανέλθει στο κανονικό της πρόγραμμα, με την δυνατόν χαμηλότερη απώλεια σε κέρδος και σε

ικανοποίηση των πελατών τους. Έτσι λοιπόν γίνεται αντιληπτό πως η διαχείριση μη προγραμματισμένων ενεργειών, έχει να κάνει, κατά την διαδικασία επίλυσης της, με τους εξής τομείς μιας αεροπορικής: δρομολόγηση αεροσκαφών, προγραμματισμός πληρώματος και αποκατάσταση των πελατών.

Η δρομολόγηση αεροσκαφών και ο προγραμματισμός πληρωμάτων γίνεται με τις διαδικασίες που αναφέρθηκαν και παραπάνω, επαναληπτικά έως ότου βρεθεί λύση που να ικανοποιεί τους νέους περιορισμούς έλλειψης δυναμικού που προέκυψαν. Κατά την διαδικασία επίλυσης της δρομολόγησης των αεροσκαφών, απαιτείται να βρεθούν οι κατάλληλες ακολουθίες πτήσεων που θα ανατεθούν σε όλα τα διαθέσιμα αεροσκάφη, ώστε να εκτελεσθεί σύντομα το κανονικό πρόγραμμα πτήσεων της αεροπορικής. Πάνω σε αυτό το πρόβλημα πολλοί μελετητές έχουν προτείνει διάφορες μεθόδους, όπως την εφαρμογή μιας μεθόδου διακλάδωσης και φραγμών για την ελαχιστοποίηση της καθυστέρησης των πελατών, καθώς και την ελαχιστοποίηση των ακυρώσεων πτήσεων. Σήμερα οι αεροπορικές εταιρίες, ως επί των πλείστων, χρησιμοποιούν ευρετικούς αλγόριθμους που προσαρμόζονται στο αρχικό πρόγραμμα και παράγουν δυνατούς συνδυασμούς πτήσεων (νέα προγράμματα δηλαδή) που να εξυπηρετούν την αεροπορική εταιρία, ή έναν συνδυασμό απλών μεθόδων με την βοήθεια της επαγγελματικής τους εμπειρίας για να επιλύσουν το προκύπτον πρόβλημα.

η) Διαχείριση εναέριας κυκλοφορίας.

Το παρόν πρόβλημα αφορά τον χειρισμό της εναέριας κυκλοφορίας, παρακολουθώντας και κατευθύνοντας την πορεία κάθε αεροσκάφους από την απογείωση έως και την προσγείωση του. Η διαχείριση της εναέριας κυκλοφορίας είναι φυσικά αρμοδιότητα της εκάστοτε εθνικής υπηρεσίας που ελέγχει τον εθνικό εναέριο χώρο, αλλά οι ενέργειες αυτής της υπηρεσίας και η εύρυθμη λειτουργία της, επηρεάζει σε μεγάλο βαθμό την λειτουργία και προγραμματισμό των αεροπορικών εταιριών. Ο προγραμματισμός της εναέριας κυκλοφορίας συνήθως σχεδιάζεται υποθέτοντας μη προβληματικές συνθήκες δηλαδή καλοκαιρία και μικρή συμφόρηση στους διάφορους επίγειους σταθμούς (αεροδρόμια). Συχνά όμως προκύπτουν προβλήματα όπως επιδείνωση του καιρού ή κάποια τεχνική βλάβη σε επίγειο σταθμό. Σε περιπτώσεις σαν και αυτές υπάρχει ανάγκη αλλαγής του αρχικού πλάνου κατά τέτοιο τρόπο ώστε να δημιουργηθεί όσο το δυνατόν μικρότερη συμφόρηση, τόσο στον αέρα όσο και στα διάφορα αεροδρόμια.

Παραδοσιακοί τρόποι αντιμετώπισης τέτοιων προβλημάτων από το προσωπικό ελέγχου εναέριας κυκλοφορίας, είναι η καθυστέρηση απογείωσης νέων αεροσκαφών, ώστε να αποτραπεί η δημιουργία μεγαλύτερης συμφόρησης, ανά-δρομολόγηση των αεροσκαφών, εναέριες καθυστερήσεις ώστε να παραταθεί ο χρόνος μέχρι την προσγείωση στο αεροδρόμιο και αποστασιοποίηση των αεροσκαφών που βρίσκονται εν πτήση, ώστε να μην υπάρξει συμφόρηση σε συγκεκριμένο τμήμα του εναέριου χώρου.

Για να αντεπεξέλθουν σε τέτοιου είδους προβλήματα διαχείρισης της εναέριας κυκλοφορίας, έχουν επιστρατευτεί μέθοδοι επιχειρησιακής έρευνας που αφορούν την πρόβλεψη δυνατοτήτων κάθε επίγειου σταθμού σε προβληματικές καταστάσεις, η την μοντελοποίηση του εναέριου δικτύου, ως ένα μεγάλης κλίμακας μη γραμμικό δίκτυο, η ακόμα και την χρήση ευρετικών αλγορίθμων που αναπαράγουν λύσεις με βάση το αρχικό πρόγραμμα που έχει σχεδιαστεί. Τέλος έχουν χρησιμοποιηθεί επίσης μοντέλα γραμμικού, ακέραιου, μεικτού ακέραιου προγραμματισμού για την επίλυση του προβλήματος.

θ) Μερικά ακόμη προβλήματα.

Μερικά ακόμα προβλήματα που έχουν να κάνουν με την λειτουργία των αεροπορικών εταιριών και έχουν βρει λύση μέσω της επιστήμης της επιχειρησιακής έρευνας είναι, η ανάθεση πυλών αεροδρομίων σε αφιχθέντες πτήσεις, διαχείριση καυσίμων και διαχείριση ανθρωπίνου δυναμικού.

Η βέλτιστη ανάθεση αφιχθέντων πτήσεων σε αεροδρόμια είναι ιδιαίτερα σημαντική, μιας και αφορά την σωστή διαχείριση τεραστίων ποσοτήτων αποσκευών και αριθμού επιβατών. Κατά την επίλυση του προβλήματος πρέπει να ληφθούν υπόψη το μέγεθος του αεροσκάφους που καταφθάνει, της διαδρομής που θα πρέπει να διανύσουν οι επιβάτες ως την παραλαβή των αποσκευών ή ως την πύλη αναχώρησης της επόμενης τους πτήσης, τις ανάγκες ανεφοδιασμού και προετοιμασίας του αεροσκάφους για την επόμενη του πτήση, την πορεία και διαχείριση των αποσκευών.

Λόγω της φύσης του προβλήματος, της αβεβαιότητας ακριβούς ώρας άφιξης των πτήσεων, την απαίτηση επίλυσης σε πραγματικό χρόνο του προβλήματος, των πολλαπλών αντικειμενικών στόχων προς βελτιστοποίηση, πολλές αεροπορικές εταιρίες χρησιμοποιούν έμπειρα συστήματα για την επίλυση του προβλήματος.

Το κόστος των καυσίμων είναι το μεγαλύτερο κόστος που επιβαρύνει μια αεροπορική εταιρία. Ουσιαστική μείωση του κόστους μπορεί να προέλθει από την αποθήκευση μεγάλων ποσοτήτων καυσίμων ή από την μεταφορά τους από σταθμό σε σταθμό, στην περίπτωση όπου μεταξύ των δυο περιοχών που αυτοί υφίστανται υπάρχουν μεγάλες διακυμάνσεις στην τιμή των καυσίμων. Οι μέθοδοι που έχουν χρησιμοποιηθεί για την ελαχιστοποίηση του κόστους των καυσίμων περιέχουν κυρίως γραμμικό προγραμματισμό και οι συνήθεις μεταβλητές που καθορίζουν το πρόβλημα, είναι οι τιμές των καυσίμων, οι δυνατότητες αποθήκευσης καυσίμων κάθε επίγειου σταθμού και οι δυνατότητες του προμηθευτή όσο αφορά τις ποσότητες με τις οποίες αυτός μπορεί να προμηθεύσει την αεροπορική.

Η διαχείριση ανθρωπίνου δυναμικού περιλαμβάνει την πρόσληψη, εκπαίδευση, και στελέχωση του προσωπικού της εταιρίας. Το προσωπικό των αεροπορικών εταιριών αφορά δυο μεγάλες κατηγορίες, το εναέριο (πιλότοι, αεροσυνοδοί) και το επίγειο (τεχνικούς ,μηχανικούς, συνοδούς εδάφους, υπαλλήλων γραφείου κ.λ.π.). Όπως είναι λογικό η δραστηριότητα και εργασία του προσωπικού στο σύνολο του εξαρτάται σε πολύ μεγάλο βαθμό από το προγραμματισμό όλων των παραπάνω προβλημάτων που αναφέρθηκαν, για παράδειγμα το προσωπικό του αεροδρομίου θα δραστηριοποιηθεί γύρω από την ώρα που μια πτήση ετοιμάζεται να προσγειωθεί ή να απογειωθεί από το αεροδρόμιο. Οι τεχνικές της επιχειρησιακής έρευνας καλούνται αξιοποιήσουν στο μέγιστο δυνατό βαθμό το σύνολο του ανθρώπινου δυναμικού, λαμβάνοντας υπόψη κάποιους περιορισμούς όπως μέγιστο επιτρεπτό ωράριο εργασίας, ημερομηνίες αδειών των εργαζομένων, αλλά και το πρόγραμμα πτήσεων που έχει σχεδιαστεί, ώστε να επέλθει το μέγιστο κέρδος.

1.2 ΤΑΙΡΙΑΣΜΑ (CREW PAIRING) ΚΑΙ ΑΝΑΘΕΣΗ ΠΛΗΡΩΜΑΤΩΝ (CREW ASSIGNMENT)

Υπάρχει μία μεγάλη ομάδα προβλημάτων που υπόκεινται στο ευρύ πεδίο του σχεδιασμού, του προγραμματισμού και της ανάθεση πόρων τα οποία είναι δύσκολα να μοντελοποιηθούν και ακόμα πιο δύσκολα να επιλυθούν. Η λύση σε τέτοια προβλήματα αποτελείται από μια κατάλληλη ανάθεση τιμών στις μεταβλητές που μοντελοποιούν τον τομέα του προβλήματος με τέτοιο τρόπο που να υπάρχει συμμόρφωση στους περιορισμούς. Τέτοια προβλήματα συχνά αναφέρονται ως συνδυαστικά προβλήματα εύρεσης, με την φιλοσοφία ότι πρέπει να ψάξουμε έναν εφικτό συνδυασμό τιμών για τις εξαρτημένες μεταβλητές.

Σε ένα συνδυαστικό πρόβλημα εύρεσης, κάποιος μπορεί να ψάχνει για μία, πολλές ή και όλες τις εφικτές λύσεις. Η εύρεση μίας ή μερικών λύσεων μπορεί να είναι αρκετά εύκολη ή πολύ δύσκολη διαδικασία και εξαρτάται από την πυκνότητα του δειγματικού

χώρου. Από την άλλη, η εύρεση πιθανής λύσης μπορεί να μην είναι εφικτή, ή ακόμα και ανεπιθύμητη σε περίπτωση που το πλήθος των εφικτών λύσεων είναι υπερβολικά μεγάλο. Όμως αυτό που συνήθως είναι απαιτούμενο είναι η εύρεση της βέλτιστης λύσης σύμφωνα με μια δεδομένη αντικειμενική συνάρτηση. Έτσι μιλάμε για προβλήματα βελτιστοποίησης, τα οποία είναι είδος προβλήματος που η επιχειρησιακή έρευνα ασχολείται εδώ και πολλά χρόνια.

Τα συνδυαστικά προβλήματα εύρεσης ελκύουν, επίσης, την προσοχή των ερευνητών της τεχνητής νοημοσύνης, που έχουν αναπτύξει πληθώρα μεθόδων και ευρετικών αλγορίθμων για την επίλυσή τους. Μεγάλη συνεισφορά αυτού του τομέα είναι η ιδέα ότι μπορεί να εκμεταλλευτούν ενεργά τους περιορισμούς, δηλαδή, οι περιορισμοί μπορούν να χρησιμεύσουν στην μείωση των μεταβλητών, πριν φτάσουμε στο σημείο να ψάχνουμε για τις πιθανές τιμές αυτών των μεταβλητών. Η επίδραση αυτής της μείωσης βοηθάει και στην περαιτέρω μείωση, μέσω άλλων περιορισμών, των πιθανών λύσεων, οδηγώντας σε έναν αντικειμενοστραφή τρόπο εξασφαλίζοντας την συνέπεια. Το συνολικό αποτέλεσμα μπορεί να είναι μια μεγάλη μείωση του δειγματικού χώρου, που εξαρτάται φυσικά από την φύση των υπαρχόντων περιορισμών. Αυτή η μέθοδος υποστηρίζεται από τον τομέα προγραμματισμού με περιορισμούς, που ερευνάται την άρδην την τελευταία δεκαετία με σκοπό να ασχοληθεί και να επιλύσει συνδυαστικά προβλήματα εύρεσης του πραγματικού κόσμου. Αρχικά, η ιδέα του προγραμματισμού με περιορισμούς είναι μια επέκταση του λογικού προγραμματισμού και της γλώσσας Prolog, δίνοντας γέννηση στον λογικό προγραμματισμό με περιορισμούς constraint logic programming (CLP). Στις μέρες μας, όμως μεταφράζεται και σε άλλα προγραμματιστικά προβλήματα όπως αυτά του αντικειμενοστραφή προγραμματισμού, κ.α.

Ο χρονοπρογραμματισμός ιπτάμενου προσωπικού των αεροπορικών εταιριών είναι ένα πολύ δύσκολο συνδυαστικό πρόβλημα, δεδομένης της πολυπλοκότητας των περιορισμών που πρέπει να ικανοποιηθούν και τον τεράστιο δειγματικό χώρο που πρέπει να εξεταστεί. Το πρόβλημα είναι συχνό φαινόμενο να διασπάται σε δύο υποπροβλήματα: α) ταίριασμα προσωπικού και β) ανάθεση προσωπικού τα οποία είναι και αυτά δύσκολα προβλήματα. Το ταίριασμα προσωπικού εξετάζεται εντατικά με πολλές διαφορετικές προσεγγίσεις όπως γενετικοί αλγόριθμοι, νευρωνικά δίκτυα κ.α. Πολύ δουλειά έχει γίνει για το πρόβλημα της ανάθεσης προσωπικού αντίστοιχα με πολλές προσεγγίσεις όπως επιχειρησιακή έρευνα, προγραμματισμός με περιορισμούς και υβριδικούς που συνδυάζουν και τους δύο.

Σε αυτή την διπλωματική, συζητάμε το πρόβλημα της ανάθεσης προσωπικού και προτείνουμε μια μοντελοποίηση ως πρόβλημα ικανοποίησης περιορισμών που μπορεί να λυθεί από μια αντικειμενοστραφή γλώσσα (C++) και το αντίστοιχο περιβάλλον (Ubuntu Linux).

Πρόβλημα ανάθεσης προσωπικού (Crew assignment problem).

Το πρόβλημα της ανάθεσης προσωπικού για τις αερογραμμές αναφέρεται στην δέσμευση προσωπικού για το πιλοτήριο και για την καμπίνα στα ταιριάσματα πτήσεων κατά την διάρκεια ενός προκαθορισμένου χρονικού διαστήματος, συνήθως ενός μήνα. Ένα ταίριασμα (pairing) είναι μία ακολουθία πτήσεων. Αρχίζει από μια βάση της εταιρίας και τελειώνει πάλι σε κάποια βάση και καταρτίζεται έτσι ώστε να μην παραβιάζονται νόμοι και κανόνες. Ένα ταίριασμα μπορεί να έχει διάρκεια από μια έως και πολλές μέρες. Το σύνολο των πτήσεων αποτελεί το πτητικό καθήκον. Ένα σύστημα ανάθεσης προσωπικού είναι υπεύθυνο για την δέσμευση προσωπικού σε προκαθορισμένα ταιριάσματα που καλύπτουν όλες τις πτήσεις μιας αεροπορικής εταιρίας για μια

δοσμένη χρονική περίοδο. Σύστημα τέτοιου είδους πρέπει να εγγυάται ότι κανένας κανονισμός δεν θα παραβιάζεται.

Μια διαδικασίας πλήρης ανάθεση γίνεται ξεχωριστά για την καμπίνα και για το πιλοτήριο, αφού τα καθήκοντα επηρεάζονται από διαφορετικούς περιορισμούς και κανόνες. Η ανάθεση στο πλήρωμα της καμπίνας μπορεί να διαιρεθεί ακόμα σε μικρότερα ανεξάρτητα υποπροβλήματα που αφορούν διαφορετικούς στόλους και ομάδες των μελών (κυβερνήτης, μηχανικός πτήσης κ.α.). Αυτό βέβαια δεν είναι πάντα εφικτό αν υπάρχουν περιορισμοί που συσχετίζουν διαφορετικές βαθμίδες εργασίας ή ακόμα αν υπάρχουν καθολικοί περιορισμοί που πρέπει να ικανοποιηθούν. Τέτοιοι είναι κανονισμοί που συσχετίζουν χαρακτηριστικά από άλλα μέλη του πληρώματος.

Εκτός από την διασφάλιση της εγκυρότητας των κανόνων και των κανονισμών, ένα σύστημα ανάθεσης προσωπικού πρέπει να ακολουθεί μια συγκεκριμένη μεθοδολογία. Τρεις βασικές μεθοδολογίες υπάρχουν:

α) Δίκαιη ανάθεση: Ο φόρτος εργασίας είναι (σχεδόν) ίσα κατανεμημένος στους εργαζομένους της εταιρίας, δηλαδή με δίκαιο τρόπο. Ρεπό, χρόνος καθήκοντος, χρόνος πτήσης, πτήσεις νωρίς/αργά και όλες οι άλλοι παράμετροι της δουλειάς πρέπει να είναι το δυνατόν ίσα.

β) Επιλογή με βάση την προσφορά: Αναρτάται από το σύστημα το πρόγραμμα των πτήσεων και κάθε πιλότος κάνει μόνος του κράτηση των αντίστοιχων πτήσεων σεβόμενο συνήθως την λογική FCFS (first-come first-served).

γ) Ανάθεση βάση προτίμησης: Οι εργαζόμενοι εκφράζουν γενικά και ειδικά τις προτιμήσεις τους (π.χ. να αποφεύγουν πρωινές πτήσεις, προτίμηση ανάθεσης μιας συγκεκριμένης πτήσης κ.α.) και το σύστημα προσπαθεί να πραγματοποιήσει τις προτιμήσεις αυτές, είτε ακολουθώντας τες απευθείας, είτε παράγοντας ατομικά πλάνα εργασίας και επιτυγχάνοντας την ανάθεση όλου του φόρτου εργασίας με την μεγαλύτερη ικανοποίηση του προσωπικού συνολικά.

Εμείς ακολουθούμε την πρώτη μεθοδολογία και προσπαθούμε να έχουμε μοιρασμένο φόρτο εργασίας σε όλους τους πιλότους.

Ακολουθούν κάποιοι απαραίτητοι ορισμοί για την κατανόηση της παρούσας διπλωματικής:

Χρόνος καθήκοντος (duty time): Κάθε συνεχόμενη περίοδος που ένα μέλος του πληρώματος (πιλότος) εκπονεί κάποια εργασία (πτήση).

Χρόνος πτητικού καθήκοντος (flying time)

Ρεπό (days off): Ελεύθερος χρόνος, δηλαδή χρόνο που ο πιλότος δεν κάνει κάποια πτήση για την εταιρία.

1.3 Τύποι μοντελοποίησης του προβλήματος της ανάθεσης προσωπικού

Υπάρχουν οι ακόλουθες προσεγγίσεις που προσπαθούν με διαφορετική προσέγγιση να μοντελοποιήσουν το πρόβλημα της ανάθεσης προσωπικού και να το επιλύσουν αντίστοιχα με τρόπο που επιλύονται τα προβλήματα των αντίστοιχων κλάδων της επιστήμης των υπολογιστών. Εκτός από τις δύο βασικές προσεγγίσεις που ακολουθούν υπάρχουν και άλλες που επιλύουν

με ικανοποιητικό τρόπο το παρόν πρόβλημα και θα αναφερθούν επιγραμματικά.

1.3.1. Γραμμικός προγραμματισμός - Linear programming

Μια μέθοδος/μοντελοποίηση που προσπαθεί να λύσει το παρόν πρόβλημα με χρήση γραμμικού προγραμματισμού είναι αυτή που έχει δημοσιεύσει ο Padberg σε μια εργασία του το 1993. Η λύση που προτείνει είναι μια προσέγγιση branch-and-cut που επιλύει με αποδεδειγμένα προβλήματα με την βέλτιστη λύση σε μεγάλου μεγέθους προβλήματα συνόλων που προέκυψαν εκείνη την δεκαετία. Η branch-and-cut μέθοδος παράγει επίπεδα βασισμένα σε βοηθητικές δομές πολυτόπων ορισμένα από την κλειστή καμπύλη εφικτών ακέραιων λύσεων και συμπεριλαμβάνει έναν αλγόριθμο αναζήτησης δέντρου που χρησιμοποιεί αυτοματοποιημένες διαδικασίες αναδιάρθρωσης, ευρετικούς αλγορίθμους και τεχνολογία γραμμικού προγραμματισμού για συνεργαστούν στην εύρεση της βέλτιστης λύσης. Αυτά τα προβλήματα περιλαμβάνουν απλά προβλήματα διαχωρισμού συνόλων και προβλήματα διαχωρισμού συνόλων ταιριασμένα με περιορισμούς. Αυτοί οι περιορισμοί αναπαριστούν τις απαιτήσεις του περιβάλλοντος οι οποίες συχνά είναι σε αντιπαράθεση με αποτέλεσμα να μην ξέρουμε πόσο γρήγορα προσεγγίζουμε την βέλτιστη λύση. Ένα ενδιαφέρον αποτέλεσμα του να παράγουμε την λύση με το μικρότερο κόστος στα προγράμματα της αεροπορικής εταιρίας είναι ότι οι εργαζόμενοι της εταιρίας είναι πιο ικανοποιημένοι διότι περνάν περισσότερες ώρες καθήκοντος σε πτήσεις παρά περιμένοντας στο έδαφος.

Ο στόχος των αεροπορικών είναι να ελαχιστοποιήσει το κόστος του προσωπικού ικανοποιώντας πολλούς περιορισμούς και κανόνες εργασίας. Το πρώτο βήμα για να μοντελοποιήσουμε ένα τέτοιο πρόβλημα είναι να βρούμε τις διαδοχικές πτήσεις (ταιριάσματα) που όμως συμμορφώνονται στους νόμους και του κανονισμούς του FAA, στα συμβόλαια που έχουν υπογραφεί με το προσωπικό κ.α. Την ίδια στιγμή μια τιμή κόστους ανατίθεται σε κάθε ταίριασμα. Αυτή η τιμή αναπαριστά όλα τα κόστη που έχουν να κάνουν με διάρκεια πτήσεων, ποσότητα και είδος καυσίμων, πόσο μακριά από την χώρα του κάθε εργαζομένου είναι οι πτήσεις κ.α.

Δοσμένου ενός εφικτού συνόλου ταιριασμάτων, το πρόβλημα μπορεί να αναδιατυπωθεί στο να βρούμε την βέλτιστη συλλογή ταιριασμάτων ώστε κάθε πτήση να είναι καλυμμένη από μόνο ένα ταίριασμα ως πρόβλημα ταιριάσματος συνόλου/set partitioning problem (SPP) ως εξής:

$\min \sum_{j=1}^{n} ci \times xj$

subject to: Ax = em (SPP),

xj ανήκει {0,1} για j = 1, … ,n,

όπου em είναι ο πίνακας με m εισόδους να ισούνται με 1 και n να είναι το νούμερο των θεωρημένων ταιριασμάτων. Κάθε γραμμή του mxn πίνακα A αναπαριστά μια πτήση, και αποτελούνται από δυαδικές μεταβλητές (0,1), συνδεδεμένες με το κάθε ταίριασμα j, με xj=1 αν το ταίριασμα j είναι επιλεγμένο και 0 διαφορετικά. Το ci είναι το συνδεδεμένο κόστος με το j ταίριασμα. Ο πίνακας A είναι δομημένος με μια στήλη την φορά όπου

$aij = \begin{cases} 1 \ εαν \ η \ πτήση \ καλύπτεται \ από \ το \ j \ ταίριασμα, \\ 0 \ αλλιώς. \end{cases}$

Ένα ιδιαίτερο χαρακτηριστικό της μεθόδου είναι το γεγονός ότι μπορεί να επιλύσει προβλήματα με οποιοδήποτε πλήθος περιορισμών πχ. περιορισμοί της μορφής:

α^0 ≤$\sum_{j\varepsilon B} aj \times xj$ ≤ α^1,

όπου B υποσύνολο του {1,2,..,n}, aj > 0 για κάθε j που ανήκει στο B και 0 < α^0 < α^1. Τέτοιοι περιορισμοί διασφαλίζουν ότι οι αερογραμμές συμμορφώνονται με τους κανόνες εργασίας του εξής είδους: ο συνολικός αριθμός των ωρών πτήσης να μην ξεπεράσει τις 32 ώρες σε μια βδομάδα. Τέτοιοι περιορισμοί περιορίζουν σημαντικά την δέσμευση των διαθέσιμων πληρωμάτων για τις πτήσεις και συνεπώς είναι ένα δομικό στοιχείο του κόστους για το ιπτάμενο προσωπικό.

Υπάρχουν τέσσερα συστατικά στον βελτιστοποιητή branch-and-cut: ένας προεπεξεργαστής που περιορίζει τον παρεχόμενο από τον χρήστη σχεδιασμό. Έναν ευριστικό αλγόριθμο που βρίσκει γρήγορα καλές εφικτές λύσεις. Μια διαδικασία παραγωγής αποκλεισμού – ο βασικός μηχανισμός όλης της προσέγγισης – που σφίγγει την χαλάρωση του γραμμικού προγραμματισμού και μια στρατηγική διακλάδωσης που επιλέγει την επόμενη μεταβλητή διακλάδωσης και καθορίζει την έρευνα του δέντρου. Προηγούμενες έρευνες έχουν δείξει ότι ενσωματώνοντας αυτά τα συστατικά στον αλγόριθμο επίλυσης έχουν επιλυθεί κλάσεις δύσκολων συνδυαστικών προβλημάτων βελτιστοποίησης που θεωρούνταν αδύνατο να επιλυθούν βέλτιστα. Η μέθοδος αυτή μελετά ουσιαστικά τον δειγματικό χώρο των λύσεων προσεγγίζοντας την βέλτιστη λύση με την βοήθεια της χρήσης πολυτόπων.

1.3.2. Λογικός προγραμματισμός με περιορισμούς - Constraint logic programming (CLP)

Ο λογικός προγραμματισμός με περιορισμούς (CLP) αναφέρεται σε μια κλάση γλωσσών προγραμματισμού που υποστηρίζουν μια υβριδική λογική που συνδυάζει τα πλεονεκτήματα των παραδοσιακών γλωσσών προγραμματισμού και την αποδοτικότητα της επίλυσης προβλημάτων με περιορισμούς. Ο CLP επωφελείται από όλα τα πλεονεκτήματα του λογικού προγραμματισμού όπως ο μη ντετερμινισμός, ενώ υπερπηδά τους περιορισμούς χάρις την μη αποδοτικότητα στην εξερεύνηση του δειγματικού χώρου των συνδυαστικών προβλημάτων.

Ο CLP είναι βασισμένος στην ιδέα ότι πολλά συνδυαστικά προβλήματα εύρεσης του πραγματικού κόσμου από διαφορετικούς τομείς μπορούν να μοντελοποιηθούν ως προβλήματα ικανοποίησης περιορισμών (CSP). Σε αυτά τα προβλήματα υπάρχει ένα σύνολο μεταβλητών, όπου η καθεμία είναι συνδυασμένη με έναν τομέα, το σύνολο των τιμών που μπορούν πιθανώς να ανατεθούν σε αυτήν.

Ένας περιορισμός cj εφαρμόζεται σε ένα σύνολο Vcj των μεταβλητών V. Εάν το μέγεθος του Vcj είναι n και κάθε μεταβλητή έχει τομέα με μέγεθος m, τότε το σύνολο SVcj των διαφορετικών πιθανών αναθέσεων τιμών στις μεταβλητές Vcj περιέχει m^n στοιχεία. Ένας περιορισμός χωρίζει αυτό το σύνολο πιθανών αναθέσεων σε συνεπείς και μη ασυνεπείς. Οι ασυνεπείς δεν συνάδουν με τους περιορισμούς και δεν είναι αποδεκτές.

Στα CSP, υπάρχει ένα σύνολο C περιορισμών, από τους οποίους ο καθένας εφαρμόζεται σε ένα διαφορετικό υποσύνολο του δειγματικού χώρου V. Μια λύση S είναι κάθε ανάθεση τιμών σε μεταβλητές που σέβεται όλους τους περιορισμούς. Με άλλα λόγια, η ανάθεση S για να είναι μια λύση, για κάθε περιορισμό cj του C, οι αναθέσεις στο S των μεταβλητών στο Vcj πρέπει να είναι συνεπείς.

Δοσμένου ενός CSP, ο στόχος είναι να βρούμε μια λύση, όλες τις λύσεις ή την βέλτιστη λύση σύμφωνα με μια δοσμένη αντικειμενική συνάρτηση. Η αναπαραγωγή περιορισμών (constraint propagation) είναι ο μηχανισμός που ελέγχει την αλληλεπίδραση των περιορισμών. Κάθε περιορισμός μπορεί να συμπεράνει απαραίτητες προϋποθέσεις στους τομείς των μεταβλητών. Όποτε ο τομέας μιας μεταβλητής διαφοροποιείται, η αναπαραγωγή περιορισμών θα πυροδοτήσει όλους τους σχετικούς με τις μεταβλητές περιορισμούς, με σκοπό να ανιχνεύσει επιπλέον συνέπειες.

Η δομή ενός CLP προγράμματος είναι η ακόλουθη:

λύσε(Λίστα): -

αρχικοποίηση_τομέα(Λίστα),

περιόρισε(Λίστα),

απαρίθμησε(Λίστα).

Το όρισμα Λίστα είναι μια λίστα μεταβλητών τομέα που αντιπροσωπεύει την λύση του προβλήματος. Στο βήμα της αρχικοποίηση_τομέα, κάθε μεταβλητή της λίστας είναι περιορισμένη σε έναν αρχικό τομέα. Στο βήμα του περιορισμού, οι περιορισμοί που αντιμετωπίζονται στο πρόβλημα υποβάλλονται στην λίστα μεταβλητών τομέα. Στο βήμα απαρίθμησης, κάθε μεταβλητή τομέα παίρνει μια τιμή με τυχαίο ή συστηματικό τρόπο. Κάθε φορά που μια τιμή ανατίθεται σε μια μεταβλητή, ο μηχανισμός αναπαραγωγής πυροδοτείται και ο λύτης περιορισμών φιλτράρει τις μεταβλητές τομέα, με σκοπό να ικανοποιήσει το σύνολο των περιορισμών. Στο τέλος του βήματος απαρίθμησης, κάθε άλλη μεταβλητή περιορίζεται σε μοναδική τιμή (εφικτή λύση) ή επιστρέφεται αποτυχία.

Η μοντελοποίηση βάση του CLP αφορά πρώτα στον ορισμό των μεταβλητών τομέα. Το πρόβλημα ανάθεσης προσωπικού παίρνει ως είσοδο ένα σύνολο ταιριασμάτων (pairings) και τα τυποποιούν όπως αναφέρεται στην επόμενη ενότητα. Έπειτα ορίζονται οι περιορισμοί οι οποίοι περιγράφονται αναλυτικά στην αναλυτική παρουσίαση του προβλήματος στην επόμενη ενότητα. Αφού έχουν είσοδο από το input file και έχουν επιλεγεί οι αντίστοιχοι περιορισμοί γίνεται βελτιστοποίηση στην αντικειμενική συνάρτηση που έχουμε ορίσει. Όπως έχουμε πει, αντικειμενικός στόχος του προβλήματος δεν είναι η εύρεση εφικτής λύσης αλλά η εύρεση της βέλτιστης λύσης. Το κριτήριο βελτιστοποίησης είναι η δικαιοσύνη στην κατανομή του χρόνου στους πιλότους. Η αντικειμενική συνάρτηση πρέπει να μετράει τον χρόνο πτήσης σύμφωνα με κριτήρια δικαιοσύνης. Μια πιθανή συνάρτηση είναι η εξής:

$Z = \sum_{i=1}^{N} |Fi - Fav|$

Fi είναι ο χρόνος πτήσης του εργαζομένου i. Αυτή η συνάρτηση δεν αντιπροσωπεύει επαρκώς το κριτήριο βελτιστοποίησης, επειδή δεν “τιμωρεί” μεγάλες αποκλίσεις από τον μέσο χρόνο πτήσης Fav. Η αντικειμενική συνάρτηση που χρησιμοποιούμε για ελαχιστοποίηση είναι η ακόλουθη:

$Z = \sum_{i=1}^{N} (Fi - Fav)^2$

Ένα σύνολο από περιορισμούς τυπικά μειώνει το τομέα μεταβλητών, αλλά μερικές φορές το ενοποιεί με μια τιμή. Η απαρίθμηση λαμβάνει χώρα στο στάδιο ονοματοδοσίας ενός προγράμματος λογικού προγραμματισμού. Ένα γενικό σχέδιο αλγορίθμου που υλοποιεί την απαρίθμηση της λίστας L των πεπερασμένων μεταβλητών τομέα είναι το εξής:

απαρίθμηση(L): -

συνθήκη_τερματισμού(L),

!.

απαρίθμηση(L): -
επιλογή_μεταβλητών(L, X),
επιλογή_τιμών(X, M),
απαρίθμηση(L).

Η δήλωση συνθήκη_τερματισμού/1 επιτυγχάνει όταν κάθε τομέας μεταβλητών είναι ενοποιημένος σε μια μοναδική λύση. Η δήλωση επιλογή_μεταβλητών(L, X) επιλέγει την επόμενη μεταβλητή X του L που θα της ανατεθεί τιμή από τον τομέα. Η φάση επιλογής μεταβλητών είναι σημείο κλειδί της έρευνας. Διαφορετική επιλογή στρατηγικής επηρεάζει την αποδοτικότητα της ανάθεσης και την τιμή της αντικειμενικής συνάρτησης. Η επιλογή μεταβλητών είναι η εξής: Διάλεξε μια μεταβλητή που ανταποκρίνεται στο ταίριασμα με τον μεγαλύτερο χρόνο πτήσης. Η δήλωση επιλογή_τιμών (X, M) αναθέτει την μεταβλητή τομέα X την τιμή M. Η επιλογή της τιμής είναι σημαντική. Η στρατηγική επιλογής που χρησιμοποιείται είναι η εξής: Διάλεξε την τιμή που απευθύνεται στον πιλότο με τον μικρότερο τρέχον χρόνο πτήσης. Αυτοί οι άπληστοι αλγόριθμοι εύρεσης δουλεύουν αρκετά καλά με το παρόν πρόβλημα, όπως έχει αποδειχθεί με πολλά πειράματα. Η διαίσθηση που υπάρχει για αυτούς είναι ότι θέλουμε να ξεφορτωθούμε νωρίς τα μεγάλα ταιριάσματα που είναι δύσκολα στην διαχείριση, ενώ τα μικρά είναι πιο ευέλικτα. Ολόκληρη η διαδικασία απαρίθμησης είναι ενσωματωμένη σε μια επαναληπτική διαίρει και βασίλευε διαδικασία, η οποία όποτε βρίσκει μια λύση με κάποιο κόστος, έστω C, επαναλαμβάνεται και αρχίζει να ψάχνει από την αρχή προσπαθώντας να βρει λύση με κόστος μικρότερο από C.

Άλλες προσεγγίσεις του λογικού προγραμματισμού είναι η παραγωγή στηλών / column generation approach και η ευριστική μέθοδος αναζήτησης δέντρου / Heuristic tree search.

1.4 ΑΝΤΙΚΕΙΜΕΝΟ ΔΙΠΛΩΜΑΤΙΚΗΣ ΕΡΓΑΣΙΑΣ

Κάθε αεροπορική εταιρεία αντιμετωπίζει το πρόβλημα της στελέχωσης των πτήσεων που πρέπει να φέρει σε πέρας με το κατάλληλο ιπτάμενο προσωπικό (κυβερνήτη, συγκυβερνήτη, συνοδούς και φροντιστές), σεβόμενη τόσο τους διεθνείς και εθνικούς κανονισμούς όσο και συλλογικές συμβάσεις που έχουν υπογραφεί με τους εργαζομένους. Η διαδικασία αυτή ονομάζεται χρονοπρογραμματισμός πληρωμάτων (crew scheduling) και, ως επί το πλείστον, γίνεται σε μηνιαία βάση. Μεταξύ των πολλών νόμιμων προγραμμάτων εργασίας που μπορούν να κατασκευασθούν, η εταιρεία επιθυμεί να υιοθετήσει το καλύτερο δυνατό, ή έστω ένα πολύ καλό, σύμφωνα με κάποιο κριτήριο, ή συνδυασμό κριτηρίων, που έχει αποφασιστεί.

Συνήθως, το πρόβλημα του χρονοπρογραμματισμού πληρωμάτων αντιμετωπίζεται σε δύο φάσεις. Αρχικά, ομαδοποιούνται οι πτήσεις που πρέπει να καλύψει η εταιρεία σε συνδυασμούς πτήσεων (pairings), όπου κάθε συνδυασμός πτήσεων είναι μια ακολουθία από τις πτήσεις που αρχίζει από την βάση της εταιρείας και καταλήγει επίσης στην βάση και είναι έτσι επιλεγμένες ώστε να μπορούν να στελεχωθούν νομίμως από το ίδιο πλήρωμα. Αυτή είναι η φάση της γέννησης συνδυασμών πτήσεων (pairing generation), τους οποίους ας ονομάσουμε στο εξής ΣΠ. Η δεύτερη φάση είναι αυτή της ανάθεσης των ΣΠ σε συγκεκριμένα μέλη πληρώματος και η οποία ονομάζεται ανάθεση πληρωμάτων (crew assignment).

Αντικείμενο της άσκησης αυτής είναι να υλοποιηθεί ένα πρόγραμμα σε C++ που να φέρει σε πέρας μία εκδοχή της δεύτερης φάσης του χρονοπρογραμματισμού πληρώματος, δηλαδή να αναθέτει ένα σύνολο από ΣΠ σε έναν αριθμό από κυβερνήτες. Κάθε ΣΠ πρέπει να ανατεθεί σε έναν ακριβώς κυβερνήτη, αλλά, προφανώς, κάθε κυβερνήτης μπορεί να αναλάβει πολλούς ΣΠ, εφ' όσον η ανάθεση αυτή είναι νόμιμη και εφικτή.

Για παράδειγμα, εφικτή δεν είναι μία ανάθεση όταν ένας πιλότος βρίσκεται σε δύο πτήσεις συγχρόνως (χρονική επικάλυψη).

Νόμιμη για εμάς είναι η ανάθεση που δεν παραβιάζει τους κανόνες που θέτουμε μέσα από το εξής σύνολο κανόνων:

1)Σε κάθε συνεχόμενο διάστημα 7 ημερολογιακών ημερών (από μεσάνυχτα σε μεσάνυχτα) πρέπει κάθε κυβερνήτης να έχει υο μέρες ανάπαυσης (ρεπό). Μία ημερολογιακή ημέρα θεωρείται ρεπό όταν ο ιπτάμενος δεν έχει κανένα πτητικό καθήκον κατά την διάρκεια της ημέρας αυτής (-daysoff7).

2)Μεταξύ δύο συνεχόμενων ΣΠ που ανατίθενται σε κάποιο κυβερνήτη, πρέπει να μεσολαβούν τουλάχιστον 11 ώρες ανάπαυσης, ή ημέρα ρεπό (-resttime).

3)Μέσα σε μία ημερολογιακή ημέρα δεν επιτρέπεται να εκτελούνται από τον ίδιο πιλότο τμήματα δύο συνδυασμών πτήσεων (-nosameday). Για παράδειγμα, αν ένας συνδυασμός πτήσεων Α τελειώνει στις 3:00 την ημέρα Χ και ο συνδυασμός πτήσεων Β αρχίζει στις 16:00 την ίδια ημέρα Χ, αν και είναι πάνω από 11 ώρες από τη λήξη του Α μέχρι την έναρξη του Β, δεν επιτρέπεται να ανατεθούν οι συνδυασμοί Α και Β στον ίδιο πιλότο.

4)Σε κάθε συνεχόμενο διάστημα 30 ημερών, πρέπει να υπάρχουν 9 ημέρες ανάπαυσης (-daysoff30) - παραλλαγή του περιορισμού 1.

5)Σε κάθε συνεχόμενο διάστημα 7 ημερών, ο συνολικός FT δεν πρέπει να υπερβαίνει τις 32 ώρες (-flighttime7).

6) Σε κάθε συνεχόμενο διάστημα 30 ημερών, ο συνολικός FT δεν πρέπει να υπερβαίνει τις 80 ώρες (-flighttime30).

7)Σε κάθε συνεχόμενο διάστημα 7 ημερών, ο συνολικός DT δεν πρέπει να υπερβαίνει τις 40 ώρες (-dutytime7).

8)Σε κάθε συνεχόμενο διάστημα 30 ημερών, ο συνολικός DT δεν πρέπει να υπερβαίνει τις 160 ώρες (-dutytime30).

Το πρόγραμμα πρέπει να παράγει νόμιμες αλλά και όσο το δυνατόν “πυκνές” αναθέσεις. Θα πρέπει να γίνει ανάθεση και για πλήρεις ημερολογιακούς μήνες, τουλάχιστον.

Μεταξύ των διαφόρων εναλλακτικών αναθέσεων που είναι νόμιμες, αλλά και “πυκνές”, οπότε χρησιμοποιούν τον ελάχιστο αριθμό κυβερνητών, πρέπει να δοθεί σαν

αποτέλεσμα εκείνη που έχει την καλύτερη, ή έστω κάποια αρκετά καλή, ισοκατανομή χρόνου πτήσης μεταξύ όλων των κυβερνητών, μέσα στην περίοδο χρονοπρογραμματισμού. Σαν μέτρο της ισοκατανομής V, που πρέπει να ελαχιστοποιηθεί, ορίζεται το εξής:

$V = \sum_{i=0}^{C} (FTi - IFT)^2$

όπου C είναι το πλήθος των κυβερνητών, FTi είναι ο συνολικός χρόνος πτήσης κάθε κυβερνήτη, που αντιστοιχεί στην απόλυτη ισοκατανομή. Δηλαδή:

$FTi = \sum_{j=1}^{P} (pij \times FTPj)$

όπου P είναι το πλήθος των ΣΠ της περιόδου, το pij δείχνει αν ο ΣΠ υπ' αριθμόν j ανατέθηκε στον κυβερνήτη i και FTPj είναι ο συνολικός χρόνος πτήσης του ΣΠ j. Συγκεκριμένα:

$pij = \begin{cases} 1, αν\ ο\ ΣΠ\ ανατέθηκε\ στον\ κυβερνήτη\ i \\ 0, αλλιώς \end{cases}$

και

$FTPj = \sum_{\kappa=1}^{Fj} (ATkj - DTkj)$

όπου Fj είναι το πλήθος των πτήσεων του ΣΠ j, ATkj και DTkj οι χρονικές στιγμές άφιξης και αναχώρησης, αντίστοιχα, της πτήσης k του ΣΠ j. Τέλος:

$IFT = (\sum_{m=1}^{P} FTPm)/ C$

1.5 ΔΟΜΗ ΔΙΠΛΩΜΑΤΙΚΗΣ ΕΡΓΑΣΙΑΣ

Η εργασία αποτελείται από τα εξής 5 κεφάλαια:

Το κεφάλαιο 1 αποτελεί το εισαγωγικό κεφάλαιο, όπου εξηγήθηκε το γενικό πρόβλημα της ανάθεσης πληρωμάτων και η χρησιμότητα της παρούσας διπλωματικής εργασίας.

Το κεφάλαιο 2 περιγράφει την μοντελοποίηση που έγινε και αναλύει τα πεδία και τις μεθόδους των κλάσεων καθώς και την χρησιμότητά τους.

Το κεφάλαιο 3 περιγράφει τις λεπτομέρειες υλοποίησης, τα δεδομένα εισόδου, τα ορίσματα, τα περιβάλλοντα ανάπτυξης και αποσφαλμάτωσης καθώς και το περιβάλλον για το οποίο έχει δημιουργηθεί η προσομοίωση.

Το κεφάλαιο 4 περιέχει την αξιολόγηση του εκτελέσιμου προγράμματος, βάσει των εκτελέσεων και των αποτελεσμάτων τους, ως προς την εγκυρότητα και την ταχύτητα εκτέλεσης.

Το κεφάλαιο 5 περιέχει την σύνοψη, δηλαδή μια συγκεντρωτική ματιά στην παρούσα διπλωματική εργασία και αναφέρει κάποιες προτάσεις για το πώς μπορεί να επεκταθεί/χρησιμοποιηθεί η παρούσα εργασία ως βάση για άλλες μελλοντικές εργασίες/προσομοιώσεις.

2. Περιγραφή κλάσεων

2.1 ΓΕΝΙΚΑ

Σε αυτό το κεφάλαιο θα περιγράψουμε αναλυτικά τις κλάσεις, τις συναρτήσεις και τις διαδικασίες που χρησιμοποιούνται προκειμένου να λυθεί το πρόβλημα που εξετάζουμε.

2.2 ΑΝΑΛΥΤΙΚΗ ΠΕΡΙΓΡΑΦΗ ΚΛΑΣΕΩΝ

2.2.1 Κλάση DateAndTime

```
class DateAndTime
{
        int day;
        int month;
        int year;
        int hour;
        int min;
public:
        DateAndTime();
        DateAndTime(int Year, int Month, int Day, int Hour, int Min);
        DateAndTime(const char*);
        time_t getAge();
        void setAge(time_t);
        void setDateTime(int Year, int Month, int Day, int Hour, int Min);
        void setStartDay() { hour = 0; min = 0; }
        void setEndDay() { hour = 23; min = 59; }
        bool operator< (const DateAndTime&) const;
};
```

Χρησιμότητα της κλάσης:

Η κλάση DateAndTime δημιουργήθηκε για να χειρίζεται ο συγγραφέας του κώδικα και όποιος θέλει να τον αναπτύξει ή να τον τροποποιήσει με ευκολία, δεδομένου ότι προσφέρει πολύ χρήσιμες διαδικασίες χειρισμού της ημερομηνίας και της ώρας απαραίτητες για την υλοποίηση αυτή. Αυτό επιτυγχάνεται διότι αυξάνεται το επίπεδο αφαίρεσης (abstraction) και ο συγγραφέας του κώδικα χειρίζεται ένα πολύ πιο εύχρηστο σύνολο διαδικασιών. Η σκέψη κλειδί για την υλοποίηση αυτή είναι η χρήση του τύπου δεδομένων time_t που μετατρέπει την ημερομηνία σε δευτερόλεπτα που έχουν περάσει από τις 00:00, 1 Ιανουαρίου, 1970. Με αυτόν τον τύπο δεδομένων καθίσταται πιο εύκολη η σύγκριση ημερομηνιών, η κύλιση χρονικού διαστήματος ή χρονικής στιγμής αλλά και η ενσωμάτωση κάποιων χρήσιμων χαρακτηριστικών στις συναρτήσεις (πχ. Εάν ένα έτος είναι δίσεκτο δεν χρειάζεται διαφορετική μεταχείριση για τις 29 Φεβρουαρίου) λόγω της φύσης του γενικού προβλήματος και των περιορισμών.

Πεδία κλάσης:
day: Η μέρα της εβδομάδας (1-31)
month: Μήνας του χρόνου (1-12)
year: Έτος (default).
hour: Ώρα της ημέρας (00:00 – 23:59)
min: Λεπτά της ώρας (0-59)

Μέθοδοι κλάσεις:
DateAndTime(): 1ος Constructor της κλάσης. Αρχικοποιεί με 0 όλα τα πεδία.

DateAndTime(int Year, int Month, int Day, int Hour, int Min): 2ος Constructor της κλάσης. Βάζει τις τιμές από τα αντίστοιχα ορίσματα στα πεδία της κλάσης. (Πχ. year = Year).

DateAndTime(const char*): 3ος Constructor της κλάσης. Παίρνει το κατάλληλο μέρος της κάθε γραμμής του αρχείου για να περάσει τις τιμές αυτές στα κατάλληλα πεδία. (input format: 2011-11-01 03:20)

time_t getAge(): Μετατρέπει την ώρα που έχει αποθηκευμένη η κλάση DateAndTime σε τύπο time_t. Ο τύπος time_t είναι ο αριθμός των δευτερολέπτων από τις 00:00, 1 Ιανουαρίου , 1970.

void setAge(time_t): Ουσιαστικά κάνει το αντίστροφο από την getAge() δηλαδή παίρνει ως είσοδο μια δομή τύπου time_t και συμπληρώνει τα πεδία της κλάσης ώστε να αναφέρονται στην ίδια χρονική στιγμή.

void setDateTime(int Year, int Month, int Day, int Hour, int Min): Βάζει τις τιμές από τα αντίστοιχα ορίσματα στα πεδία της κλάσης όπως ο 2ος Constructor της κλάσης . (Πχ. year = Year).

void setStartDay() { hour = 0; min = 0; }: Θέτει σαν ώρα στην κλάση την αρχή της ημέρας.

void setEndDay() { hour = 23; min = 59; }: Θέτει σαν ώρα στην κλάση το τέλος της ημέρας.

bool operator< (const DateAndTime&): Κάνουμε symbol overloading για να συγκρίνουμε ποια από δύο χρονικές στιγμές προηγείται της άλλης ανάμεσα σε δύο κλάσεις τύπου DateAndTime.

Βοηθητική συνάρτηση:
int diffDays(DateAndTime d1, DateAndTime d2): Παίρνει ως είσοδο δύο χρονικές στιγμές και επιστρέφει πόσες μέρες διαφορά έχουν αυτές οι στιγμές μεταξύ τους.

2.2.2 Κλάση Flight

```
class Flight
{
		int fp_id;
		int flight_num;
		string ap_start, ap_end;
		DateAndTime time_start, time_end;
public:
		Flight(string&);
		int get_flight_plan_id() { return fp_id; }
		DateAndTime& get_time_start() { return time_start; }
		DateAndTime& get_time_end() { return time_end; }
		time_t get_flight_duration();
		bool operator< (const Flight&) const;
};
```

Χρησιμότητα της κλάσης:
Με απλή και κατανοητή από τον καθένα μοντελοποίηση περιγράφει μια πτήση καθώς και προσφέρει ένα σύνολο κανόνων για την εξαγωγή πρόσθετης πληροφορίας (πχ. get_flight_duration()).

Πεδία κλάσης:
fp_id: Αύξων αριθμός ΣΠ.
flight_num: Αριθμός πτήσης.
ap_start: Αεροδρόμιο αναχώρησης.
ap_end: Αεροδρόμιο άφιξης.
time_start: Ώρα αναχώρησης.
time_end: Ώρα άφιξης.

Μέθοδοι κλάσεις:
Flight(string&): Constructor της κλάσης flight. Παίρνει σαν όρισμα την τρέχουσα γραμμή εισόδου από το αρχείο και με το κατάλληλο parsing βάζει τις τιμές των αντίστοιχων strings σαν τιμές στα πεδία της κλάσης.

int get_flight_plan_id() { return fp_id; }: Επιστρέφει την τιμή του fp_id (αύξων αριθμός ΣΠ).

DateAndTime& get_time_start() { return time_start; }: Επιστρέφει την τιμή της ώρας αναχώρησης.

DateAndTime& get_time_end() { return time_end; }: Επιστρέφει την τιμή της ώρας άφιξης.

time_t get_flight_duration(): Επιστρέφει την διάρκεια της πτήσης.

bool operator< (const Flight&): Κάνουμε symbol overloading για να συγκρίνουμε ποια πτήση αρχίζει πρώτη και άρα ποια προηγείται χρονικά. Πρακτικά συγκρίνουμε πτήσεις στον ίδιο ΣΠ και γι αυτό η σύγκριση αρκεί να ελέγχει την αρχή των πτήσεων(δεν υπάρχει χρονική επικάλυψη ανάμεσα σε δύο πτήσεις του ίδιου ΣΠ).

2.2.3 Κλάση FlightPlan

```
class FlightPlan
{
        int fp_id;
        list<Flight> flights;
        DateAndTime time_start, time_end;
public:
        FlightPlan(int id);
        bool addFlight(Flight &);
        DateAndTime get_time_start() { return time_start; }
        DateAndTime get_time_end() { return time_end; }
        int get_fp_id() { return fp_id; }
        int get_flights_count() { return flights.size(); }
        int get_flight_plan_id() { return fp_id; }
        time_t get_ftp();
        time_t get_ft();
        time_t get_dt();
        bool operator< (const FlightPlan&) const;
};
```

Χρησιμότητα της κλάσης:
Οργανώνουμε το ΣΠ (FlightPlan) χρησιμοποιώντας μια λίστα από δεδομένα τύπου flight και κρατώντας το id του ΣΠ και κάποιες χρήσιμες πληροφορίες, όπως το time_start και το time_end, για να μην κάνουμε περιττή προσπέλαση στην λίστα. Με αυτήν την κλάση στοχεύουμε στην μείωση του μεγέθους του κώδικα και την αύξηση της αναγνωσιμότητας σε κάποια κομμάτια κώδικα (πχ. Αντιγραφή ενός ΣΠ αντί για αντιγραφή μίας μίας των πτήσεων).

Πεδία κλάσης:
fp_id: Αύξων αριθμός ΣΠ(πλάνο πτήσης)
list<Flight> flights: Είναι μια λίστα από πτήσεις η οποίες αποτελούν τον ΣΠ με αύξων αριθμός ΣΠ fp_id.

time_start: Ώρα έναρξης του ΣΠ ο οποίος είναι ουσιαστικά το time_start της πρώτης πτήσης από την λίστα flights.

time_end: Ώρα λήξης του ΣΠ ο οποίος είναι ουσιαστικά το time_end της τελευταίας πτήσης από την λίστα flights.

Μέθοδοι κλάσεις:

FlightPlan(int id): Constructor της κλάσης FlightPlan. Αρχικοποιεί το πεδίο fp_id με την τιμή που παίρνει σαν όρισμα.

bool addFlight(Flight &): Εισάγει την πτήση flight στην λίστα flights αν έχει το ίδιο fp_id. Εάν είναι η πρώτη πτήση του ΣΠ τότε το time_start του FlightPlan είναι ίσο με το time_start του flight. Τέλος θέτουμε το time_end του FlightPlan ίσο με το time_end του flight.

DateAndTime get_time_start() { return time_start; }: Επιστρέφει την τιμή της ώρας αναχώρησης της πρώτης πτήσης, δηλαδή την ώρα που αρχίζει να εκτελείται το FlightPlan.

DateAndTime get_time_end() { return time_end; }: Επιστρέφει την τιμή της ώρας άφιξης της τελευταίας πτήσης, δηλαδή την ώρα που έρχεται σε πέρας το FlightPlan.

int get_fp_id() { return fp_id; }: Επιστρέφει την τιμή του fp_id (αύξων αριθμός ΣΠ).

int get_flights_count() { return flights.size(); }: Επιστρέφει το πλήθος των πτήσεων που αποτελούν το FlightPlan.

int get_flight_plan_id() { return fp_id; }: Επιστρέφει την τιμή του fp_id (αύξων αριθμός ΣΠ).

time_t get_ftp(): Επιστρέφει την χρονική διάρκεια του ΣΠ, δηλαδή το time_end του FlightPlan μείον το time_start του FlightPlan.

time_t get_ft(): Επιστρέφει την τιμή του flying time στο τρέχον ΣΠ. Το flying time είναι ο χρόνος που το ΣΠ βρίσκεται σε πτήση δηλαδή είναι ίσο με το άθροισμα της διάρκειας του συνόλου των πτήσεων του ΣΠ.

time_t get_dt(): Επιστρέφει την τιμή του duty time στο τρέχον ΣΠ. Το duty time είναι ο χρόνος που μεσολαβεί από την έναρξη της πρώτης πτήσης του συνδυασμού μέχρι τη λήξη της τελευταίας πτήσης του.

bool operator< (const FlightPlan&): Κάνουμε symbol overloading για να συγκρίνουμε ποιος ΣΠ αρχίζει πρώτος και άρα ποιος προηγείται χρονικά.

Βοηθητικές συνάρτησεις:

list<FlightPlan> get_list_within_days (list<FlightPlan> lfp, DateAndTime start, int days): Παίρνει ως είσοδο μία λίστα από ΣΠ, την χρονική αφετηρία και τον αριθμό των ημερών και επιστρέφει το μέρος της λίστας που είναι μέσα σε αυτή την χρονική περίοδο.

time_t get_ft_from_list(list<FlightPlan> lfp): Από τη λίστα ΣΠ που παίρνει ως όρισμα, επιστρέφει το flying time που είναι το άθροισμα των flying time των ΣΠ.

time_t get_dt_from_list(list<FlightPlan> lfp): Από τη λίστα ΣΠ που παίρνει ως όρισμα, επιστρέφει το duty time που είναι το άθροισμα των duty time των ΣΠ.

time_t get_max_ft(list<FlightPlan> &lfp, int days): Η συνάρτηση παίρνει ως είσοδο μια λίστα με ΣΠ και επιστρέφει το μέγιστο flying time σε διάστημα days ημερών (πχ. 7 ή 30).

time_t get_max_dt(list<FlightPlan> &lfp, int days): Η συνάρτηση παίρνει ως είσοδο μια λίστα με ΣΠ και επιστρέφει το μέγιστο duty time σε διάστημα days ημερών (πχ. 7 ή 30).

2.2.4 Κλάση Captain

```
class Captain
{
        int captain_id;
        list<FlightPlan> fps;
public:
        Captain(int id) : captain_id(id) {}
        bool has_valid_fp(list<int> &fp, Assignments& a);
        bool assign_flight_plan(FlightPlan &fp);
        list<FlightPlan>& get_fps() { return fps; }
        bool check_rule_1(FlightPlan &fp);
        bool check_rule_2(FlightPlan &fp);
        bool check_rule_3(FlightPlan &fp);
        bool check_rule_4(FlightPlan &fp);
        bool check_rule_5(FlightPlan &fp);
        bool check_rule_6(FlightPlan &fp);
        bool check_rule_7(FlightPlan &fp);
        bool check_rule_8(FlightPlan &fp);
        void print_fps();
        time_t get_FT();
};
```

Χρησιμότητα της κλάσης:

Με αυτήν την κλάση μοντελοποιούμε τον πιλότο και τις διαδικασίες που πρέπει να υλοποιεί. Αποτελείται από το μοναδικό id του και από την λίστα ΣΠ που του έχουν ανατεθεί. Για να του γίνει ανάθεση ΣΠ σύμφωνα με τους περιορισμούς που είναι ενεργοί

πρέπει ο ίδιος να ελέγχει αν παραβιάζεται κάποιος κανόνας με τις κλήσεις check_rule_i όπου i το αντίστοιχο νούμερο των περιορισμών. Τέλος, σε αυτή την κλάση υλοποιούνται και συναρτήσεις που κάνουν ανάθεση και έλεγχο της εγκυρότητας για τα ΣΠ προς ανάθεση στον πιλότο.

Πεδία κλάσης:
captain_id: Αναγνωριστικός αριθμός του κάθε πιλότου. Είναι μοναδικός για κάθε πιλότο.

list<FlightPlan> fps: Λίστα από FlightPlans. Είναι ο φόρτος εργασίας(αναθέσεις πτήσεων) του κάθε πιλότου.

Μέθοδοι κλάσεις:
Captain(int id) : captain_id(id) {}: Constructor της κλάσης Captain. Αρχικοποιεί το id του πιλότου, δηλαδή τον αναγνωριστικό αριθμό του.

bool assign_flight_plan(FlightPlan &fp): Για όποιους περιορισμούς από τους παρακάτω είναι ενεργοί ελέγχει αν μπορεί να κάνει ανάθεση στον πιλότο τον ΣΠ (fp). Αν όλοι οι ενεργοί περιορισμοί ικανοποιούνται τότε μπορεί να κάνει ανάθεση του ΣΠ κάνοντας fps.push_back(fp), δηλαδή εισαγωγή στο τέλος της λίστας και επιστρέφει true. Αν παραβιάζεται έστω και ένας επιστρέφει false και δεν προσθέτει τον ΣΠ στην λίστα.

bool has_valid_fp(list<int> &fp, Assignments& a): Αντίστοιχα με την assign_flight_plan() για όποιους περιορισμούς από τους παρακάτω είναι ενεργοί ελέγχει αν μπορεί να κάνει ανάθεση στον πιλότο τον ΣΠ (fp). Αν όλοι οι ενεργοί περιορισμοί ικανοποιούνται τότε μπορεί να κάνει ανάθεση του ΣΠ κάνοντας fps.push_back(fp), δηλαδή εισαγωγή στο τέλος της λίστας και επιστρέφει true. Αν παραβιάζεται έστω και ένας επιστρέφει false και δεν προσθέτει τον ΣΠ στην λίστα και τυπώνει στην οθόνη ποιος περιορισμός παραβιάζεται από αυτήν την ανάθεση.

list<FlightPlan>& get_fps() { return fps; }: Επιστρέφει την λίστα με τα FlightPlans.

bool check_rule_1(FlightPlan &fp): Ελέγχει τον περιορισμό 1: Σε κάθε συνεχόμενο διάστημα 7 ημερολογιακών ημερών (από μεσάνυχτα σε μεσάνυχτα) πρέπει κάθε κυβερνήτης να έχει δυο μέρες ανάπαυσης (ρεπό). Μία ημερολογιακή ημέρα θεωρείται ρεπό όταν ο ιπτάμενος δεν έχει κανένα πτητικό καθήκον κατά την διάρκεια της ημέρας αυτής (-daysoff7). Ακολουθεί την εξής συλλογιστική πορεία: Βρίσκει την πρώτη και την τελευταία μέρα και δημιουργεί bool πίνακα που αντιστοιχούν στις μέρες (αρχικοποιημένες με false). Αν κάποια μέρα έχει πτητικό καθήκον κάνουμε το αντίστοιχο πεδίο true. Στο τέλος, με την βοήθεια της check_working_days() ελέγχουμε αν παραβιάζεται ο κανόνας 1. Αν παραβιάζεται ο κανόνας επιστρέφει false αλλιώς επιστρέφει true.

bool check_rule_2(FlightPlan &fp): Ελέγχει τον περιορισμό 2: Μεταξύ δύο συνεχόμενων ΣΠ που ανατίθενται σε κάποιο κυβερνήτη, πρέπει να μεσολαβούν τουλάχιστον 11 ώρες ανάπαυσης, ή ημέρα ρεπό (-resttime). Έτσι ελέγχουμε αν τα ΣΠ του πιλότου απέχουν από το fp τουλάχιστον 11 ώρες, δηλαδή 3600*11 δευτερόλεπτα. Αν παραβιάζεται ο κανόνας επιστρέφει false αλλιώς επιστρέφει true.

bool check_rule_3(FlightPlan &fp): Ελέγχει τον περιορισμό 3: Μέσα σε μία ημερολογιακή ημέρα δεν επιτρέπεται να εκτελούνται από τον ίδιο πιλότο τμήματα δύο συνδυασμών πτήσεων (-nosameday). Για παράδειγμα, αν ένας συνδυασμός πτήσεων Α τελειώνει στις 3:00 την ημέρα Χ και ο συνδυασμός πτήσεων Β αρχίζει στις 16:00 την ίδια ημέρα Χ, αν και είναι πάνω από 11 ώρες από τη λήξη του Α μέχρι την έναρξη του Β, δεν επιτρέπεται να ανατεθούν οι συνδυασμοί Α και Β στον ίδιο πιλότο. Έτσι ελέγχουμε αν το fp είναι άλλη μέρα απ' ότι τα άλλα ΣΠ του πιλότου. Αν παραβιάζεται ο κανόνας επιστρέφει false αλλιώς επιστρέφει true.

bool check_rule_4(FlightPlan &fp): Ελέγχει τον περιορισμό 4: Σε κάθε συνεχόμενο διάστημα 30 ημερών, πρέπει να υπάρχουν 9 ημέρες ανάπαυσης (-daysoff30) - παραλλαγή του περιορισμού 1. Ακολουθεί την εξής συλλογιστική πορεία: Βρίσκει την πρώτη και την τελευταία μέρα και δημιουργεί bool πίνακα που αντιστοιχούν στις μέρες (αρχικοποιημένες με false). Αν κάποια μέρα έχει πτητικό καθήκον κάνουμε το αντίστοιχο πεδίο true. Στο τέλος, με την βοήθεια της check_working_days() ελέγχουμε αν παραβιάζεται ο κανόνας 3. Αν παραβιάζεται ο κανόνας επιστρέφει false αλλιώς επιστρέφει true.

bool check_rule_5(FlightPlan &fp): Ελέγχει τον περιορισμό 5: Σε κάθε συνεχόμενο διάστημα 7 ημερών, ο συνολικός FT δεν πρέπει να υπερβαίνει τις 32 ώρες (-flighttime7). Έτσι ελέγχουμε με την βοήθεια της get_max_ft(), βάζοντας στην λίστα τον fp, αν το max flying time είναι μεγαλύτερο των 32 ωρών (3600*32 δευτερολέπτων) για 7 μέρες. Αν παραβιάζεται ο κανόνας επιστρέφει false αλλιώς επιστρέφει true.

bool check_rule_6(FlightPlan &fp): Ελέγχει τον περιορισμό 6: Σε κάθε συνεχόμενο διάστημα 30 ημερών, ο συνολικός FT δεν πρέπει να υπερβαίνει τις 80 ώρες (-flighttime30). Έτσι ελέγχουμε με την βοήθεια της get_max_ft(), βάζοντας στην λίστα τον fp, αν το max flying time είναι μεγαλύτερο των 80 ωρών (3600*80 δευτερολέπτων) για 30 μέρες. Αν παραβιάζεται ο κανόνας επιστρέφει false αλλιώς επιστρέφει true.

bool check_rule_7(FlightPlan &fp): Ελέγχει τον περιορισμό 7: Σε κάθε συνεχόμενο διάστημα 7 ημερών, ο συνολικός DT δεν πρέπει να υπερβαίνει τις 40 ώρες (-dutytime7). Έτσι ελέγχουμε με την βοήθεια της get_max_dt(), βάζοντας στην λίστα τον fp, αν το max duty time είναι μεγαλύτερο των 40 ωρών (3600*40 δευτερολέπτων) για 7 μέρες. Αν παραβιάζεται ο κανόνας επιστρέφει false αλλιώς επιστρέφει true.

bool check_rule_8(FlightPlan &fp): Ελέγχει τον περιορισμό 8: Σε κάθε συνεχόμενο διάστημα 30 ημερών, ο συνολικός DT δεν πρέπει να υπερβαίνει τις 160 ώρες (-dutytime30). Έτσι ελέγχουμε με την βοήθεια της get_max_dt(), βάζοντας στην λίστα τον fp, αν το max duty time είναι μεγαλύτερο των 160 ωρών (3600*160 δευτερολέπτων) για 30 μέρες. Αν παραβιάζεται ο κανόνας επιστρέφει false αλλιώς επιστρέφει true.

void print_fps(): Τυπώνει τους ΣΠ του πιλότου κάνοντας προσπέλαση σειριακά στην λίστα των FlightPlan.

time_t get_FT(): Επιστρέφει το flying time του πιλότου που είναι ίσο με το σύνολο των flying time των ΣΠ που έχει αναλάβει.

2.2.5 Κλάση Assignments

```
class Assignments
{
		time_t IFT;
		time_t *FT;
		time_t V;
		list<FlightPlan> flight_plans;
		int N;
		Captain **c;
public:
		Assignments(int n);
		~Assignments();
		void calculate_IFT();
		void calculate_V();
		bool make_assignments(double threshold);
		bool check_assignments();
		bool parse_input_file(bool print, string&, DateAndTime *start=NULL, DateAndTime *end=NULL);
		FlightPlan getFP_withID(int id);
};
```

Χρησιμότητα της κλάσης:

Η πιο σύνθετη από τις κλάσεις που συνδυάζοντας τις παραπάνω κλάσεις και κάποια άλλα δεδομένα, υλοποιεί τις αναθέσεις (έλεγχο εγκυρότητας και παράδοση καθηκόντων). Δημιουργήθηκε για να έχουν εύκολη επικοινωνία οι παραπάνω κλάσεις μεταξύ τους και προσφέροντας μια εύκολη διεπαφή που μπορεί κάποιος νέος χρήστης του κώδικα να κατανοήσει ή και να τροποποιήσει εύκολα. Εκτός των ελέγχων και της ανάθεσης στους πιλότους διαβάζει το αρχείο εισόδου για να εξάγει τα δεδομένα των πτήσεων, υπολογίζει τα απαιτούμενα στατιστικά και κάνει δυναμική χρήση της μνήμης (με new() και delete()) για πιο αποδοτική χρήση της μνήμης.

Πεδία κλάσης:

time_t IFT: IFT είναι ο ιδανικός συνολικός χρόνος πτήσης κάθε κυβερνήτη, που αντιστοιχεί στην απόλυτη ισοκατανομή. Αναλυτικά εκφράζεται με τον τύπο:

IFT = ($\sum_{m=1}^{P} FTPm$)/ C

time_t *FT: Το FTi είναι ο συνολικός χρόνος πτήσης του i κυβερνήτη μέσα στην περίοδο. Αναλυτικά εκφράζεται με τον τύπο:

FTi = $\sum_{j=1}^{P} (pij \times FTPj)$

Έτσι δημιουργούμε το FT ως πίνακα από δεδομένα τύπου time_t που κρατάει τα FTi των n πιλότων.

time_t V: V είναι το μέτρο ισοκατανομής που πρέπει να ελαχιστοποιηθεί για να είναι το δυνατών δίκαιες οι αναθέσεις ΣΠ στους πιλότους και δίνεται από τον τύπο:

$V = \sum_{i=0}^{C} (FTi - IFT)^2$

Μέθοδοι κλάσεις:

Assignments(int n): Constructor της κλάσης Assignments. Αρχικοποιεί τα στατιστικά της κλάσης με 0(IFT, FTi, V) και έπειτα αφού κάνει allocation στην κατάλληλη μνήμη καλεί τον conctructor της κλάσης Captain για τους N πιλότους.

~Assignments(): Destructor της κλάσης Assignments. Διαγράφει (με χρήση delete) τα μέρη μνήμης που έγιναν δυναμικά allocated (με χρήση new), τα οποία είναι ο πίνακας από πιλότους και ο πίνακας από FTi. Επίσης οι δείκτες παίρνουν την τιμή NULL για να εκμηδενίσουμε την πιθανότητα προσπέλασης λάθος κομματιού της μνήμης.

void calculate_IFT(): Υπολογίζει το IFT. Με την βοήθεια της get_ftp() βρίσκει το άθροισμα των flying time των ΣΠ, ψάχνοντας την λίστα των ΣΠ, και το διαιρεί με το πλήθος των πιλότων.

void calculate_V(): Υπολογίζει το V σύμφωνα με τον παραπάνω τύπο για το V, βρίσκοντας το FTi του κάθε με πιλότου με την βοήθεια της get_FT().

bool make_assignments(double threshold): Κάνει assignments των ΣΠ σύμφωνα με τα εξής βήματα:

Για όλα τα ΣΠ

Για όλους τους πιλότους

αν ξεπεράσουμε το thr_value με την ανάθεση

μην την κάνεις και πήγαινε στο επόμενο πιλότο.

αν μπορώ να κάνω ανάθεση ΣΠ στον πιλότο

κάνε την ανάθεση και πήγαινε στο επόμενο ΣΠ

Να επισημανθεί ότι το thr_value είναι το επαυξημένο ποσοστό του IFT που βάζουμε ως όριο για τις αναθέσεις για να υπάρχει ένα όριο που εξασφαλίζει παρόμοιο φόρτο εργασίας στους πιλότους ανάλογα με την επιλογή μας (thr_value = (threshold+1) * IFT). Έπειτα τύπωσε όλες τις αναθέσεις για κάθε πιλότο(print_fps()), με κάθε γραμμή να αντιστοιχεί σε ένα πιλότο και υπολόγισε τον V (calculate_V()). Επιστρέφει true αν έγινε ανάθεση όλων των ΣΠ.

bool check_assignments(): Διαβάζει από το stdin γραμμή γραμμή τις αναθέσεις ΣΠ και βλέπει αν είναι εφικτές: Δημιουργεί έναν πιλότο και ελέγχει αν είναι νόμιμη και εφικτή η ανάθεση με την χρήση της has_valid_fp(). Αν δεν είναι νόμιμη ή εφικτή τυπώνει την εσφαλμένη ανάθεση και επιστρέφει false.

bool parse_input_file(bool print, string&, DateAndTime *start=NULL, DateAndTime *end=NULL):

FlightPlan getFP_withID(int id): Προσπελαύνει σειριακά τη λίστα με τα ΣΠ και με την βοήθεια της get_fp_id(), αν κάποιο έχει το ίδιο id το επιστρέφει. Αν δεν βρεθεί κάποιο επιστρέφει με casting το 0 (return (FlightPlan)0;).

3. Λεπτομέρειες υλοποίησης

3.1 ΓΕΝΙΚΑ

Σε αυτό το κεφάλαιο θα δούμε αναλυτικά τις λεπτομέρειες υλοποίησης, τα δεδομένα εισόδου, τα ορίσματα, τα περιβάλλοντα ανάπτυξης και αποσφαλμάτωσης καθώς και το περιβάλλον για το οποίο έχει δημιουργηθεί η προσομοίωση.

3.2 ΑΡΧΕΙΟ ΕΙΣΟΔΟΥ

Στο αρχείο εισόδου αναγράφονται με αυστηρή δομή οι πτήσεις που αποτελούν τα ΣΠ και άρα τα πτητικά καθήκοντα τα οποία θα αναθέσουμε στους πιλότους με κάθε γραμμή του αρχείου να αντιστοιχεί σε μια πτήση. Η αυστηρή αυτή δομή (format) των γραμμών είναι η εξής:

Στήλες 1 - 4: Αύξων αριθμός ΣΠ

Στήλες 6 – 8: Αριθμός πτήσης

Στήλες 10 – 12: Αεροδρόμιο αναχώρησης

Στήλες 14 – 16: Αεροδρόμιο άφιξης

Στήλες 18 – 27: Ημερομηνία αναχώρησης (της μορφής YYYY-MM-DD)

Στήλες 29 – 33: Ώρα αναχώρησης (της μορφής HH:MM)

Στήλες 35 – 44: Ημερομηνία άφιξης (της μορφής YYYY-MM-DD)

Στήλες 46 – 50: Ώρα άφιξης (της μορφής HH:MM)

3.3 ΕΞΟΔΟΣ - ΟΡΙΣΜΑΤΑ - ΠΑΡΑΜΕΤΡΟΙ ΕΚΤΕΛΕΣΗΣ

Σχετικά με την έξοδο που θα παράγει το πρόγραμμά αυτό, πρέπει οπωσδήποτε, για λόγους ελέγχου της ορθότητας των αποτελεσμάτων, να εκτυπώνεται στο stdout μια γραμμή για κάθε κυβερνήτη, στην οποία θα περιέχονται οι αύξοντες αριθμοί ΣΠ που του έχουν ανατεθεί. Το πρόγραμμα θα πρέπει επίσης να είναι σε θέση να κάνει, εκτός από την ανάθεση ΣΠ σε κυβερνήτες, και τον έλεγχο αν μια ανάθεση που του δίνεται στο stdin είναι νόμιμη. Τέλος, θα πρέπει να δέχεται από την γραμμή εντολών τιμές για κάποιες παραμέτρους της ανάθεσης, ορισμένες από τις οποίες είναι υποχρεωτικές, ενώ οι υπόλοιπες μπορεί να μην δοθούν, οπότε θα ισχύουν οι default τιμές. Οι επιλογές τους προγράμματος στη γραμμή εντολής, που μπορεί να είναι με οποιαδήποτε σειρά, είναι οι εξής:

-p <filename> : Το αρχείο εισόδου με τους ΣΠ είναι το <filename> (υποχρεωτικό).

-n <N> : Οι ΣΠ θα ανατεθούν σε <N> κυβερνήτες (υποχρεωτικό).

-s <starttime> : Θα ανατεθούν ΣΠ που η χρονική στιγμή έναρξης της πρώτης πτήσης τους είναι από το <starttime> (στο format YYYY-MM-DD/HH:MM) και μετά. Αν δεν δοθεί η παράμετρος, να χρησιμοποιηθεί η τιμή 2001-01-01/00:00.

-e <endtime> : Θα ανατεθούν ΣΠ που η χρονική στιγμή έναρξης της πρώτης πτήσης τους είναι μέχρι και το <endtime> (στο format YYYY-MM-DD/HH:MM) και μετά. Αν δεν δοθεί η παράμετρος, να χρησιμοποιηθεί η τιμή 2020-12-31/23:59.

-r <seed> : Σε περίπτωση που το πρόγραμμα χρησιμοποιεί την rand() για να κάνει τυχαίες επιλογές σε διάφορα σημεία του, η γεννήτρια τυχαίων αριθμών να αρχικοποιηθελί με το <seed>, ώστε να είναι δυνατόν να αναπαραχθούν τα αποτελέσματά του. Αν δεν δοθεί η παράμετρος, να χρησιμοποιείται ως φύτρο ο τρέχων χρόνος. Αν το πρόγραμμά δεν έχει τυχαιότητα στην εκτέλεσή του, η επιλογή αυτή δεν είναι υποχρεωτικό να υλοποιηθεί.

-i : Το πρόγραμμα να ελέγξει την εγκυρότητα της ανάθεσης που δίνεται από το stdin. Υπενθυμίζεται ότι μια ανάθεση αποτελείται από μια γραμμή για κάθε κυβερνήτη, η οποία περιλαμβάνει τους αριθμούς ΣΠ που έχουν ανατεθεί στον κυβερνήτη. Αν η ανάθεση είναι έγκυρη, να εκτυπώνεται το μέτρο της ισοκατανομής V. Αν όχι, να εκτυπώνεται κατάλληλο διαγνωστικό μήνυμα με τους λόγους για τους οποίους η ανάθεση δεν είναι έγκυρη. Αν δεν δοθεί η επιλογή αυτή, στην περίπτωση που έχει δοθεί η -r, η τελευταία αγνοείται.

3.4 ΠΕΡΙΒΑΛΛΟΝ ΥΛΟΠΟΙΗΣΗΣ – ΑΠΟΣΦΑΛΜΑΤΩΣΗΣ

Το πρόγραμμα αναπτύχθηκε με την βοήθεια του κειμενογράφου gedit των linux. Αποσφαλμάτωση και έλεγχος ορθής χρήσης της μνήμης έγινε με τα εργαλεία GDB (GNU debugger) και Valgrind (ελέγχει memory leaks). Το περιβάλλον στο οποίο τρέχει το πρόγραμμα/προσομοίωση είναι οι περισσότερες standar διανομές ubuntu (linux, solaris κ.α.).

3.5 MAKEFILE – ΑΡΧΕΙΑ

Το makefile για την κομματιαστή μεταγλώτιση του κώδικα και την σύνδεση των αντικειμενικών αρχείων είναι το εξής:

```
OBJS = main.o source.o
SOURCE = main.cpp source.cpp
HEADER = source.h
OUT = crewas
CC = g++
FLAGS = -g -c
DIR = .
all: $(OBJS)
	$(CC) $(OBJS) -g -o $(OUT)
main.o: main.cpp source.h
	$(CC) $(FLAGS) main.cpp
source.o: source.cpp source.h
	$(CC) $(FLAGS) source.cpp
clean:
	rm -f $(OBJS) $(OUT)
```

Βλέπουμε ότι μεταγλωτίζουμε ξεχωριστά τα main.cpp και source.cpp σύμφωνα με τον ενδεδειγμένο τρόπο δημιουργίας ενός σύγχρονου makefile χρησιμοποιώντας τον μεταγλωτιστή g++ και κάνοντας ξανά μεταγλώτιση μόνο στα αρχεία που είναι outdated ως εξής: <prompt> make all

και αντίστοιχα καθαρίζουμε τα παραγόμενα αρχεία (εκτελέσιμο και αντικειμενικά) ως εξής: <prompt> make clean.

Τα αρχεία περιέχουν αντίστοιχα:

main.cpp : Τον κορμό της υλοποίησης, δηλαδή την main που χειρίζεται τα ορίσματα και καλεί αντίστοιχα τις κατάλληλες συναρτήσεις για την περάτωση της προσομοίωσης.

source.cpp : Την υλοποίηση των κλάσεων και όλων των βοηθητικών συναρτήσεων που χρησιμοποιούνται.

source.h : Αρχείο επικεφαλίδας (header file) που γίνεται include στα παραπάνω αρχεία με δηλώσεις των κλάσεων και των βοηθητικών μεθόδων καθώς και των global μεταβλητών που χρησιμοποιούμε για τον έλεγχο των ενεργών περιορισμών.

4. Αξιολόγηση

4.1 ΓΕΝΙΚΑ

Σε αυτό το κεφάλαιο θα κάνουμε μια αναλυτική αξιολόγηση του προγράμματος αυτού, βάσει των εκτελέσεων και των αποτελεσμάτων τους, ως προς την εγκυρότητα, την ταχύτητα εκτέλεσης και το κατά πόσο οι λύσεις που παράγονται είναι τουλάχιστον ανεκτές ή ακόμα οι βέλτιστες (κοντά στις βέλτιστες).

4.2 ΕΚΤΕΛΕΣΕΙΣ – ΑΠΟΤΕΛΕΣΜΑΤΑ

Παραθέτουμε αναλυτικά τα αποτελέσματα των εκτελέσεων από το terminal των ubuntu linux στις δύο διαφορετικές χρήσεις του εκτελέσιμου: α) Ανάθεση ΣΠ στους πιλότους και β) Έλεγχος εγκυρότητας αναθέσεων.

α)Παραδείγματα εκτέλεσης ανάθεσης ΣΠ στους πιλότους:

1) > crewas -p Pairings.txt -n 3 -s 2011-11-01/00:00 -e 2011-11-02/23:59 -daysoff7

Έξοδος στο stdin:

Inserted 51 flight plans into the list.

0001 0002 0003 0004 0005 0006 0007 0008 0009 0010 0011 0012 0013 0014 0015 0016 0018

0017 0025 0026 0027 0028 0029 0030 0031 0032 0033 0034 0035 0036 0037 0051 0052 0053 0054 0055 0056 0057 0058

0059 0060 0061 0062 0063 0064 0065 0080 0081 0082 0089 0105

2) > crewas -p Pairings.txt -n 5 -s 2011-11-01/00:00 -e 2011-11-02/23:59 -daysoff7

Έξοδος στο stdin:

Inserted 51 flight plans into the list.

0001 0002 0003 0004 0005 0006 0007 0008 0009 0012

0010 0011 0013 0014 0015 0016 0017 0018 0025 0026 0029

0027 0028 0030 0031 0032 0033 0034 0035 0036 0037 0051 0052 0053 0054 0059

0055 0056 0057 0058 0060 0061 0062 0063 0064 0065 0080 0082

0081 0089 0105

3) > crewas -p Pairings.txt -n 10 -s 2011-11-01/00:00 -e 2011-11-02/23:59 -daysoff7 -resttime

Έξοδος στο stdin:

Inserted 51 flight plans into the list.

0001 0011 0059

0002 0013

0003 0014

0004 0015

0005 0016

0006 0017

0007 0018

0008 0057

0009 0060

0010 0082

4) > crewas -p Pairings.txt -n 5 -s 2011-11-01/00:00 -e 2011-11-04/23:00 -daysoff7 -flighttime7

Έξοδος στο stdin:

Inserted 104 flight plans into the list.

0001 0002 0003 0004 0005 0006 0007 0008 0009 0010 0011 0012 0013 0014 0015 0016 0017 0018 0019 0021 0023 0025

0022 0024 0026 0027 0028 0029 0030 0031 0032 0033 0034 0035 0036 0037 0038 0039 0040 0041 0042 0043 0044 0045 0046 0047 0052

0048 0049 0050 0051 0053 0054 0055 0056 0057 0058 0059 0060 0061 0062 0063 0064 0065 0066 0067 0068 0069 0070 0071

0072 0073 0080 0081 0082 0083 0084 0085 0086 0087 0088 0089 0090 0091 0092 0093 0094 0105 0106 0107 0108 0109 0110 0111 0112 0113 0114 0115 0117

0116 0131 0132 0140 0155

5) > crewas -p Pairings.txt -n 5 -s 2011-11-02/11:00 -e 2011-11-04/23:59 -daysoff7 -flighttime30

Έξοδος στο stdin:

0019 0021 0022 0023 0024 0038 0039 0040 0041 0042 0043 0044

0045 0046 0047 0048 0049 0050 0057 0058 0059 0060 0064 0065 0066 0067 0070

0068 0069 0071 0072 0073 0081 0082 0083 0084 0085 0086 0087 0088 0092 0093

0090 0091 0094 0105 0106 0107 0108 0109 0110 0111 0112 0113 0114 0115 0116 0117 0131

0132 0140 0155

6) > crewas -p Pairings.txt -n 6 -s 2011-11-01/00:00 -e 2011-11-04/23:00 -daysoff7 -flighttime7 -dutytime7

Έξοδος στο stdin:

0001 0002 0003 0004 0005 0006 0007 0008 0009 0010 0011 0012 0013 0014 0015 0016 0017 0018

0019 0021 0022 0023 0024 0025 0026 0027 0028 0029 0030 0031 0032 0033 0034 0035 0036 0037 0038 0039 0040 0052

0041 0042 0043 0044 0045 0046 0047 0048 0049 0050 0051 0053 0054 0055 0056 0057 0058 0059 0060 0061

0062 0063 0064 0065 0066 0067 0068 0069 0070 0071 0072 0073 0080 0081 0082 0083 0084 0085 0087 0092

0086 0088 0089 0090 0091 0093 0094 0105 0106 0107 0108 0109 0110 0111 0112 0113 0114 0115 0116 0117 0131 0132 0155

0140

7) > crewas -p Pairings.txt -n 5 -s 2011-11-02/20:00 -e 2011-11-04/23:00 -daysoff7 -flighttime7 -dutytime30

Έξοδος στο stdin:

0021 0022 0023 0024 0038 0039 0040 0041 0042 0043 0092

0044 0045 0046 0047 0048 0049 0050 0066 0067 0068 0070

0069 0071 0072 0073 0083 0084 0085 0086 0087 0088 0090 0093 0094

0091 0106 0107 0108 0109 0110 0111 0112 0113 0114 0115 0116 0117 0131

0132 0140 0155

8) > crewas -p Pairings.txt -n 6 -s 2011-11-02/00:00 -e 2011-11-06/00:00 -daysoff30 -resttime

Έξοδος στο stdin:

0013 0024 0047 0074 0100

0014 0023 0045 0071 0096

0015 0038 0066 0079 0139

0016 0039 0049 0073 0099

0017 0040 0067 0095

0018 0041 0068 0097

9) > time ./crewas -p Pairings.txt -n 10 -s 2011-11-01/00:00 -e 2011-12-01/00:00 -daysoff7 -flighttime7 -dutytime30 -resttime

Έξοδος στο stdin:

Inserted 814 flight plans into the list.

0001 0011 0019 0045 0071 0096

0002 0013 0024 0047 0074 0100 0178

0003 0014 0023 0046 0072 0099 0170 0200

0004 0015 0038 0066 0079 0139

0005 0016 0039 0049 0073 0119 0171 0207

0006 0017 0040 0067 0095 0172

0007 0018 0041 0068 0097 0175 0208

0008 0020 0144 0173 0202

0009 0021 0048 0076 0174 0199 0210 0237 0265

0010 0022 0050 0077 0176 0204 0228 0257 0286

real 0m6.908s

user 0m4.282s

sys 0m2.603s

β)Παραδείγματα εκτέλεσης ελέγχου εγκυρότητας αναθέσεων:

1) > ./crewas -n 10 -p Pairings.txt -s 2011-12-01/00:00 -e 2011-12-31/23:59 | ./crewas -p Pairings.txt -n 10 -s 2011-12-01/00:00 -e 2011-12-31/23:59 -i

Έξοδος στο stdin:

Inserted 838 flight plans into the list.

Valid assignment

V = 49881202281000

2) > crewas -p Pairings.txt -n 6 -s 2011-11-02/00:00 -e 2011-11-06/00:00 -daysoff30 -resttime | crewas -p Pairings.txt -n 6 -s 2011-11-02/00:00 -e 2011-11-06/00:00 -daysoff30 -resttime -i

Έξοδος στο stdin:

Inserted 104 flight plans into the list.

Valid assignment

V = 1161072060000

3) > ./crewas -p Pairings.txt -n 1 -s 2011-11-01/00:00 -e 2011-12-01/00:00 -daysoff7 -daysoff30 -flighttime7 -flighttime30 -dutytime30 -nosameday -dutytime7 -resttime |

./crewas -p Pairings.txt -n 1 -s 2011-11-01/00:00 -e 2011-12-01/00:00 -daysoff7 -daysoff30 -flighttime7 -flighttime30 -dutytime30 -nosameday -dutytime7 -resttime -i

Έξοδος στο stdin:

4) > time crewas -p Pairings.txt -n 10 -s 2011-11-01/00:00 -e 2011-12-10/00:00 -daysoff7 -daysoff30 -flighttime7 -flighttime30 -dutytime30 -nosameday -dutytime7 -resttime | crewas -p Pairings.txt -n 10 -s 2011-11-01/00:00 -e 2011-12-10/00:00 -daysoff7 -daysoff30 -flighttime7 -flighttime30 -dutytime30 -nosameday -dutytime7 -resttime -i

Έξοδος στο stdin:

Inserted 1059 flight plans into the list.

Valid assignment

V = 79327849104000

real 0m19.497s

user 0m12.798s

sys 0m6.720s

5) ./crewas -p Pairings_2n+0.txt -n 19 -s 2011-11-17/13:32 -e 2012-02-08/07:52 -daysoff7 -resttime | ./crewas -i -p Pairings_2n+0.txt -n 19 -s 2011-11-17/13:32 -e 2012-02-08/07:52 -daysoff7 -resttime

Έξοδος στο stdin:

Inserted 1127 flight plans into the list.

Rule 1 is invalid

Invalid assignment:

426 --> 454 --> 482 --> 506 --> 528 --> 610 --> 640 --> 668 --> 676 --> 698 --> 722 --> 802 --> 812 --> 838 --> 864 --> 888 --> 910 --> 992 --> 1022 --> 1050 --> 1058 --> 1080 --> 1104 --> 1184 --> 1194 --> 1220 --> 1246 --> 1270 --> 1292 --> 1374 --> 1404 --> 1432 --> 1440 --> 1462 --> 1486 --> 1566 --> 1576 --> 1602 --> 1628 --> 1652 --> 1674 --> 1756 --> 1786 --> 1814 --> 1822 --> 1844 --> 1868 --> 1948 --> 1958 --> 1984 --> 2010 --> 2034 --> 2056 --> 2138 --> 2168 --> 2196 --> 2204 --> 2226 --> 2250 --> 2330 --> 2340 --> 2366 --> 2392 --> 2416 --> 2438 --> 2520 --> 2550 --> 2578 --> 2586 --> 2608 --> 2632 --> NULL

4.3 ΑΞΙΟΛΟΓΗΣΗ ΑΠΟΤΕΛΣΜΑΤΩΝ

Με τα παραπάνω αποτελέσματα της ενότητας 4.2 διαπιστώνουμε ότι το πρόγραμμα λειτουργεί αποτελεσματικά και για την ανάθεση εργασίας και για τον έλεγχο εγκυρότητας. Σε επίπεδο ορθότητας αποτελεσμάτων βλέπουμε αν ελέγξουμε τα αρχεία εισόδου ότι γίνονται αναθέσεις που υπακούν στους αντίστοιχους περιορισμούς της κάθε εκτέλεσης, θέτοντας τους περιορισμούς είτε ξεχωριστά έναν έναν είτε σε συνδυασμό (από 2 έως και 8 περιορισμοί συγχρόνως). Σε επίπεδο πολυπλοκότητας και χρόνου εκτέλεσης βλέπουμε ότι η προσομοίωση εκτελείται με μεγάλη ταχύτητα για το μέγεθος του προβλήματος και κάνει τους μόνο τους αναγκαίους υπολογισμούς που απαιτούνται για την επίλυση του προβλήματος. Έχουμε και παράδειγμα προσομοίωσης που γίνεται ανάθεση και εγκυρότητα για ενάμιση μήνα, χρονικό διάστημα που είναι πάνω και από τις συνήθεις προσδοκίες κάποιων αεροπορικών, δεδομένου ότι κάνουν κυρίως βραχυπρόθεσμες αναθέσεις προσωπικού.

5. Επίλογος

5.1 ΣΥΝΟΨΗ

Στην παρούσα εργασία είδαμε αναλυτικά το πρόβλημα της ανάθεσης εργασίας / crew assignment από τις σκοπιές του σύγχρονου προγραμματισμού και επιλύσαμε αποτελεσματικά ένα ειδικό υποπρόβλημα αναθέτοντας τους καθήκοντα (πτήσεις) σύμφωνα με τους παραπάνω περιορισμούς. Αναπτύχθηκε κώδικας στην γλώσσα προγραμματισμού C++ που υλοποιεί την ανάθεση και τον έλεγχο εγκυρότητας σε ικανοποιητικό βαθμό από άποψη πολυπλοκότητας και ταχύτητας εκτέλεσης. Αυτή η εργασία όπως αναφέρεται και παρακάτω μπορεί να χρησιμοποιηθεί ως αναφορά για περαιτέρω έρευνα.

5.2 ΜΕΛΛΟΝΤΙΚΕΣ ΕΡΓΑΣΙΕΣ

Η παρούσα διπλωματική θα μπορούσε να αποτελέσει τη βάση για ένα σύνολο εργασιών ή εφαρμογών που έχουν σχέση με την ανάθεση εργασίας και τον έλεγχο της εγκυρότητας στον ήδη κατανεμημένο φόρτο εργασίας. Τέτοιες πιθανές εργασίες / εφαρμογές είναι οι εξής:

1) Στην πραγματικότητα το περιβάλλον μεταβάλλεται πολύ γρήγορα και με απρόβλεπτο τρόπο και έτσι δημιουργούνται και άλλοι περιορισμοί. Μια περίπτωση επέκτασης, λοιπόν, είναι η συγγραφή ακόμα παραπάνω περιορισμών οι οποίοι θα ενεργοποιούνται αντίστοιχα ώστε να υπάρχει πιο πλήρες και δυναμικό σύνολο κανόνων. Η δομή και η υλοποίηση των περιορισμών είναι άκρως αυτόνομη και αποδοτική και έτσι ενθαρρύνονται τέτοιες αλλαγές

2) Άλλη επέκταση της προσομοίωσης είναι η αύξηση των τύπων των εργαζομένων. Για παράδειγμα, μια πτήση δεν χρειάζεται μόνο πιλότο, αλλά συγκυβερνήτη, αεροσυνοδούς κ.α. Έτσι μπορούμε να κάνουμε μια υπερκλάση (πχ. worker) για όλους τους εργαζόμενους που θα έχει τα στοιχεία του εργαζομένου, τον τύπο του και όποιες πληροφορίες κριθούν αναγκαίες από τον συγγραφέα. Έτσι λόγω της κληρονομικότητας στη C++ (επειδή είναι αντικειμενοστραφής γλώσσα έχει πλεονέκτημα σε τέτοιου είδους μοντελοποιήσεις) θα μπορεί να στελεχωθεί σωστά ένα σύνολο ΣΠ μιας εταιρείας.

3) Η τρίτη πιθανή κατεύθυνση είναι περισσότερο στραμμένη στις τάσεις της εποχής που θέλει online real time ενημέρωση των φυσικών ατόμων, είτε μέσω online server, είτε μέσω application, είτε μέσω e-mail το οποίο ουσιαστικά θα αυτοματοποιήσει την ανάθεση καθηκόντων στους εργαζόμενους. Μια ακόμα πιο προχωρημένη προσέγγιση είναι η εφαρμογή ανάδρασης ώστε να λαμβάνονται υπ' όψιν οι προτιμήσεις των εργαζομένων και η χρήση ιστορικού για τον φόρτο εργασίας του καθένα τους, τα οποία θα δημιουργήσουν την πιο δίκαιη ανάθεση και ισοκατανομή σε βάθος χρόνου.

4) Η επόμενη πιθανή προέκταση είναι ίσως πιο θεωρητική και είναι μια προσπάθεια μετατροπής του μοντέλου ανάθεσης μιας εταιρείας σε ένα Partially Observable Markov Decision Process (POMDP) όπου σε κάθε βάση (αφετηρία) της εταιρείας, με την βοήθεια ενός συντονιστή θα γίνεται η ανάθεση προσωπικού λαμβάνοντας υπ' όψιν την

διαμοιραζόμενη πληροφορία με την μέθοδο της κοινής πληροφόρισης βελτιστοποιώντας έτσι την πολιτική ανάθεσης. Γενικά ο κλάδος αποκεντρωμένης λήψης αποφάσεων είναι υπό ανάπτυξη και αρκετά απαιτητικός και γι αυτό μια τέτοια εργασία θα είχε πολύ μεγάλη δυσκολία αλλά και ακαδημαϊκό ενδιαφέρον.

Βιβλιογραφία

[1] Barnhart, C. and Shenoi, R.G. (1998). An approximate model and solution approach for the long-haul crew pairing problem, Transportation Science.

[2] Boubaker, K., Desaulniers, G. and Elhallaoui, I., (2010). Bidline scheduling with equity by heuristic dynamic constraint aggregation. Transportation Research Part B.

[3] Campbell, K.W., Durfee, R.B. and Hines, G.S., (1997). FedEx generates bid lines using Simulated Annealing. Interfaces.

[4] Christou, I.T., Zakarian, A., Liu, J. and Carter, H., (1999). A two-phase genetic algorithm for large-scale bidline-generation problems at Delta Air Lines. Interfaces.

[5] Desaulniers, G., Desrosiers, J., Dumas Y., Marc, S., Rioux, B., Solomon, M.M. and Soumis, F. (1997). Crew pairing at Air France. European Journal of Operational Research.

[6] Desaulniers, G., Desrosiers, J., Gamache M. and Soumis, F. (1998). Crew scheduling in air transportation.

[7] 7 .Elhallaoui, I., Villeneuve, D., Soumis, F. and Desaulniers, G., (2005). Dynamic aggregation of set partitioning constraints in column generation. Operations Research.

[8] Elhallaoui, I., Metrane, A., Soumis, F., and Desaulniers, G. (2008a). Multiphase dynamic constraint aggregation for set partitioning type problems. Mathematical Programming A.

[9] Gopalakrishnan, B. and Johnson, E.L. (2005). Airline crew scheduling: State-of-the-art. Annals of Operations Research.

[10] Jarrah, A.I.Z., and Diamond, J.T., (1997). The problem of generating crew bidlines, Interfaces.

[11] Klabjan, D. (2005). Large-scale models in the airline industry. In G. Desaulniers, J. Desrosiers and M.M. Solomon (eds), Column Generation.

[12] Klabjan, D., Johnson, E. and Nemhauser, G.L., (2001). Solving large airline crew scheduling problems: random pairing generation and strong branching. Computational Optimization and Applications.

[13] Saddoune, M., Desaulniers, G., and Soumis, F. (2009). Aircrew pairing with possible repetitions of the same flight number. Technical report G-2009-76, Les Cahiers du GERAD.

[14] Saddoune, M., Desaulniers, G., Elhallaoui, I., and Soumis, F., 2010). Integrated airline crew pairing and crew assignment by dynamic constraint aggregation. Technical report G-2010-05, Les

[15] Cahiers du GERAD.

[16] Vance, P.H., Barnhart, C., Johnson, E.L. and Nemhauser, G.L. (1997). Airline crew scheduling: A new formulation and decomposition algorithm. Operations Research.

[17] Zeghal F.M, and Minoux, M., (2006). Modeling and solving a crew assignment problem in air transportation. European Journal of Operational Research.

[18] G. Christodoulou and P. Stamatopoulos, Crew assignment by logic programming,2nd Hellenic Conf. On AI, SETN-2002, 11-12 April 2002, Thessaloniki, Greece, Proceedings, Companion Volume, pp. 117-128.

[19] Manfred Padberg, Karla L. Hoffman, Solving Airline Crew Scheduling Problems by Branch-and-Cut, Operations Research Department, George Mason University

[20] http://www.cplusplus.com

[21] http://www.gnu.org/software/gdb/

[22] http://www.valgrind.org/