OPERATIONAL AIRCRAFT MAINTENANCE ROUTING PROBLEM WITH REMAINING TIME CONSIDERATION

by

Mehmet Başdere

B.S., Industrial Engineering, Boğaziçi University, 2010

Submitted to the Institute for Graduate Studies in
Science and Engineering in partial fulfillment of
the requirements for the degree of
Master of Science

Graduate Program in Industrial Engineering

Boğaziçi University

2012

ACKNOWLEDGEMENTS

First of all, I would like to thank Prof. Ümit Bilge for her endless support, patience and sensibility throughout my entire graduate study. This work would not be completed without her wise advices and suggestions. Secondly, I would like to express my gratitudes to Prof. Necati Aras and Assoc. Prof. Deniz Aksen for taking part in my thesis jury.

I would like to thank Gökalp, Umut, Erinç and Serkan for their friendship, unignorable support and the great times we spent together in BUFAIM Laboratory. I also want to thank my colleagues Ezgi, Turgut and Burak for their patience in listening to the difficulties I encountered during this study.

I am grateful to my childhood friends, Bilal and Sibel, for trusting and encouraging me at times when I feel doubtful about my decisions and for being the source of my energy when I feel tired.

I will never be able to find correct words to express my gratitudes to my family. I would like to thank my sister for making me remember the fact that I am still her little brother no matter what I achieve. I thank my mother for her unlimited encouragement and patience and my father for being the light of the path I am following. Thank you for loving me more than everything you have.

Finally, I would like to thank my love, Tuğçe, for being so supportive, caring and patient. Your love and support make every problem I face easier to solve. Without the warmth of your love, I will definitely be lost.

I also thank TUBITAK for their financial support during my graduate study.

ABSTRACT

OPERATIONAL AIRCRAFT MAINTENANCE ROUTING PROBLEM WITH REMAINING TIME CONSIDERATION

Aircraft maintenance routing problem is one of the most studied problems in airline industry. However, most of these studies focus on finding a unique rotation that will be repeated by each aircraft in the fleet with a certain lag. In practice, using a single rotation for the entire fleet may not be applicable due to the stochastic environment and operational considerations in the airline industry. In this study, our aim is to develop a fast responsive methodology which provides maintenance feasible routes for each aircraft in the fleet for the planning horizon with the objective of maximizing utilization of the total remaining flying time of fleet. For this purpose, we formulate an integer linear programming (ILP) model by modifying the connection network representation. ILP model can be considered as an operational aircraft maintenance routing model since it provides routes for each aircraft with respect to initial states. The proposed model is solved using B&B under different priority settings for variables to branch on. A simulated annealing based heuristic method (compressed annealing) is utilized for the same problem and results are compared with those of B&B. Additionally, a rolling horizon based routing methodology is introduced and maintenance capacity constraints are discussed. Finally, compressed annealing method with maintenance capacity consideration is tested on the operational routing problem of a domestic airline company.

ÖZET

KALAN SÜRE DEĞERLENDİRMELİ OPERASYONEL UÇAK BAKIM ROTALAMA PROBLEMİ

Uçak bakım rotalama problemi hava yolu endüstrisinde sıklıkla çalışılmış problemlerden biridir. Bu alandaki birçok çalışmada odak noktası filodaki bütün uçakların belli zaman farkları ile senkronize bir şekilde tekrar edebileceği tek bir rotasyon bulmaktır. Pratik olarak bakıldığında, tek bir rotasyonu bütün filodaki uçaklara tekrarlatmak hava yolu endüstrisindeki rasgelelik ve operasyonel kısıtlar düşünüldüğünde pek de mümkün olmayabilir. Bu çalışmada amacımız, filodaki bütün uçaklara bakım açısından olurlu rotalar sağlayabilecek, filonun kalan toplam uçuş süresinin verimini en büyükleyecek ve karşılaşılabilecek değişikliklere hızlı yanıt veren bir çözüm yöntemi geliştirmektir. Bu amaçla, ilgili yazında yer alan uçuş bağlantı ağı gösterimini değiştirerek oluşturulan bir ağ üzerinde tamsayılı doğrusal programlama modeli (TDP) geliştirilmiştir. Geliştirilen TDP modeli, uçaklara başlangıç durumlarına göre rotalar ataması sebebiyle, operasyonel uçak bakım rotalama problemidir. Model dallanma sırasında farklı karar değişkenlerine öncelik veren farklı dal-sınır yöntemleri ile çözülmüştür. Bunun yanı sıra, benzetimli tavlama tabanlı sezgisel bir method olan basınçlı tavlama yöntemi de aynı problemin çözümü için kullanılmış ve basınçlı tavlama yönteminin sonuçları dal-sınır yöntemininkilerle karşılaştırılmıştır. Bunlara ek olarak, dönen planlama ufku tabanlı bir çözüm yöntemi oluşturulmuş ve bakım kapasitesi kısıtları tartışılmıştır. Son olarak, hava alanı bakım kapasitelerini hesaba katan basınçlı tavlama tekniği, yerel bir hava yolu şirketinin rotalama problemi üzerinde denenmiştir.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF SYMBOLS

c_{aw}	the maintenance capacity of airport a during time window w
d_i	duration of flight leg i
e_{awkl}	binary parameter that takes value 1 if l^{th} maintenance opportunity of aircraft k take place in airport a during time window w and 0 otherwise
$f_s(.)$	objective function of compressed annealing solution s
g_s	the amount of unused legal flying time for compressed annealing solution s
h_s	the number of maintenance infeasible aircrafts for compressed annealing solution s
m_{kl}	the l^{th} maintenance opportunity of aircraft k
m_{kl}^{p}	a binary variable denoting the maintenance opportunities in set $\overline{M}_p$
o	artificial source node of flight network
$ord(i,j)$	ordinary arc between flights i and j
r_k	binary variable which takes value of 1 if aircraft k undergoes a maintenance operation within its route 0 otherwise
r_{kl}	the amount of remaining time incurred by aircraft k when it selects the l^{th} maintenance opportunity in its route
$rep(i,j)$	replenishment arc between flights i and j
t	artificial sink node of flight network
u_s	the number of unsatisfied booked maintenance slots for compressed annealing solution s
x_{ijk}	binary variable which takes value of 1 if aircraft k flies flight legs i and j consecutively before replenishment and 0 otherwise
y_{ijk}	binary variable which takes value of 1 if aircraft k flies flight legs i and j consecutively and undergoes maintenance in between and 0 otherwise

y^p_{ijk}	a binary variable denoting the replenishment arc $rep(i,j)$ of aircraft k in set R_p
z_{ijk}	binary variable which takes value of 1 if aircraft k flies flight legs i and j consecutively either after replenishment or in a route with no replenishment and 0 otherwise
$A(i,j)$	index set consisting of p indices of R_p sets that contain $rep(i,j)$ as an element
$\mathscr{A}$	set of airports
$B(k)$	set of initial flight legs that aircraft k can fly
C	set of critical aircrafts
C'	set of non-critical aircrafts
C''	the set of aircrafts that need to undergo maintenance with respect to the route assignments of solution s
D_k	remaining time of aircraft k
$E(k,l)$	index set of p indices of $\overline{M}_p$ sets that contain m_{kl} as an element
I	set of flight legs
K	set of aircrafts
L	length of the longest route on connection network in terms of the accumulated flight duration
M_k	the set of feasible maintenance opportunities for aircraft k
$\overline{M}_p$	set of maintenance opportunities which can take place in slot p
O	set of ordinary arcs
O_{am}	set of ordinary arcs after maintenance
O_{bm}	set of ordinary arcs before maintenance
$P(i)$	set of flight legs that immediately precede flight leg i
$P'(i)$	set of flight legs that immediately precede flight leg i and that allow maintenance in between
$\mathscr{P}$	the set of booked maintenance slots
R	set of replenishment arcs

$\overline{R}_p$	set of replenishment arcs that represent maintenance opportunities which can take place within the booked maintenance slot p
R_{aw}	the replenishment arcs that represent maintenance opportunities in airport a during time window w
$S(i)$	set of flight legs that immediately succeed flight leg i
$S'(i)$	set of flight legs that immediately succeed flight leg i and that allow maintenance in between
T	set of time windows
U	underutilization percentage of total remaining time
$\mathscr{W}$	set of time windows
X_0	acceptance ratio at τ_0
Z	objective value of a solution
Z_f	objective value of the first feasible solution found during compressed annealing replication
β	temperature decrease parameter
γ	pressure increase parameter
$\|\Delta f\|_p$	absolute objective difference of solution pair p
κ	the ratio of penalty term to objective function value when pressure is λ_{max}
λ	variable pressure parameter for maintenance infeasible aircrafts
λ_{max}	maximum pressure for maintenance infeasible aircrafts
λ^*	pressure cap
μ	variable pressure parameter for unsatisfied booked maintenance slots
μ_{max}	maximum pressure for unsatisfied booked maintenance slots
ν	constant penalty multiplier in simulated annealing
σ_Z	standard deviation of compressed annealing replication results for a certain case
τ	temperature parameter of compressed annealing
τ_0	initial temperature of compressed annealing

LIST OF ACRONYMS/ABBREVIATIONS

AMR	Aircraft Maintenance Routing
AMRP	Aircraft Maintenance Routing Problem
B&B	Branch-and-Bound
CA	Compressed Annealing
CMA	Capacitated Maintenance Assignment Model
COAMRP-ILP	Integer Linear Programming formulation of Capacitated Operational Aircraft Maintenance Routing Problem
CPU	Central Processing Unit
GB	Gigabyte
GRASP	Greedy Randomized Adaptive Search Procedure
ILP	Integer Linear Programming
LOF	Lines of Flight
NP	Nondeterministic Polynomial Time
OAMRP	Operational Aircraft Maintenance Routing Problem
OAMRP-ILP	Integer Linear Programming formulation of Operational Aircraft Maintenance Routing Problem
OD	Origin Destination
RATSP	Asymmetric Traveling Salesman Problem
RHCMA	Rolling Horizon Based Capacitated Maintenance Assignment Model
RTN	Rotation Tour Network
SA	Simulated Annealing
TSP	Traveling Salesman Problem

1. INTRODUCTION

Operations research methods are widely used for different planning and scheduling problems in airline industry. These problems can be divided into four major classes: Flight scheduling problem, fleet assignment problem, aircraft maintenance routing problem and crew scheduling problem (Liang and Chaovalitwongse , 2009). In general terms, flight scheduling problem deals with scheduling of flights so that market demand is met. Fleet assignment problem tries to sort out the assignments of fleets to predetermined flights with the aim of maximizing the total profit. Crew scheduling problem, on the other hand, tries to handle the assignments of crew to each aircraft and aircraft maintenance routing problem (AMRP) deals with arranging routes for aircrafts so that the maintenance regulation constraints are not violated. Although all of these problems are widely studied for the last few decades, the challenge remains fresh due to high complexity of airline networks and increasing size of industry.

In addition to these four major problems, airline companies must also deal with tail assignment problem as the last stage of planning and scheduling operations. In simple terms, tail assignment is the operational decision of assigning aircrafts (with certain tail numbers) to already scheduled routes to perform the corresponding flights (Grönkvist , 2005). The nature of tail assignment problem is different from those that are discussed above because at tail assignment stage, all the operational restrictions must be considered and satisfied. It is not realistic to assume AMRP and tail assignment problem as two separate stages of airline scheduling since the routes must be determined by considering the current states of the aircrafts.

Particularly, in AMRP, the aim is to find a maintenance feasible rotation. Each aircraft needs to undergo certain type of maintenance after operating for certain hours or flying for certain consecutive days without maintenance. There are various types of maintenance checks and the variation stems from location, duration and frequency requirements of different checks (Clarke *et al.* , 1997). For example, A-check must be

repeated very frequently (every 65 hours of flying or every week) and it involves visual inspection of major systems for a few hours. On the other hand, D-Check must be repeated for every four or five years and it can be only completed at specialized hangars in about one month (Sriram and Haghani , 2003). It must be noted that AMRP deals with short-term or mid-term checks with shorter maintenance intervals. The reason behind this rationale is that less frequent checks generally last longer and they directly affect fleet capacity; hence, such maintenance checks must be considered while solving the fleet assignment problem. Therefore, it can be claimed that AMRP cannot be considered as pure-tactical or pure-operational problem because it is strongly related to both fleet assignment and tail assignment problems.

The problems in airline planning and scheduling have been studied very immensely for the last few decades. Particularly, there are many studies which deal with AMRP focusing on either operational (short-term planning) or tactical (long-term and aggregate planning) level. Most of the early studies focus on tactical level, ignoring some of the operational level constraints. Unlike such studies, our aim is to develop a fast and responsive methodology to deal with dynamic and operational changes in airline environment, namely to solve the operational aircraft maintenance routing problem (OAMRP) with the objective of maximizing the utilization of *remaining times* of the aircrafts. Remaining time of an aircraft is the difference between legal flying hour limit, which is the amount of time allowed between consecutive maintenance operations, and the accumulated flight duration from the last maintenance operation. A solution to OAMRP is the assignment of *maintenance feasible routes* to each aircraft in the fleet where the assigned routes cover all flights in the corresponding flight schedule. In a maintenance feasible route, there cannot be a maintenance-free segment of flights whose accumulated duration is larger than the remaining time of the corresponding aircraft. There are some recent studies which are related with OAMRP; however, this study differs from the existing ones in terms of the proposed integer linear programming (ILP) formulation and solution methodology.

The paper is organized as follows: In Chapter 2, we present a literature review about AMRP in general. In Chapter 3, the OAMRP model is described and a new ILP formulation is proposed. In Chapter 4, details on variable prioritization in branch-and-bound (B&B) are given and the heuristic method (compressed annealing) is presented. Chapter 5 covers the numerical experiments and comparisons of proposed methods. In Chapter 6, a rolling horizon based routing methodology is introduced and maintenance capacity related modifications are discussed. In Chapter 7, concluding remarks are provided.

2. LITERATURE REVIEW

There is a rich literature on airline planning and scheduling problems. In this chapter, we briefly describe and discuss some of the major studies in AMRP; however, one can refer to the surveys of Etschmaier and Mathaisel (1985); Gopalakrishnan and Johnson (2005); Sherali *et al.* (2006) for flight scheduling, fleet assignment and crew scheduling problems, respectively.

Similar to the other planning and scheduling problems in airline industry, there is a vast literature on AMRP. It must be noted that these studies differ with respect to their approaches to AMRP: Operational level or tactical level. In aircraft maintenance routing (AMR) studies, where tactical approach is used, the aim is to find a unique route which can be repeated by the aircrafts in the fleet. In such studies, initial conditions of aircrafts and dynamic changes (cancellations or delays) in flight environment are ignored. On operational level side, every restriction must be taken into consideration because the solutions of operational level problems determine the plans to be followed in real life. Additionally, many studies have been focusing on integrated models in which at least two of the four major problems of airline scheduling are considered simultaneously. The classification of aircraft maintenance related studies can be seen in Table 2.1.

There are two widely used network representations for airline network modeling: Time-space network and connection network (Liang and Chaovalitwongse , 2009). In time-space network, basically, nodes represent arrival or departure event of aircrafts on a timeline and there are three different types of arcs which represent flights, overnights and ground. In connection network, on the other hand, a simpler representation is used where flight arcs are transformed into flight nodes. The arcs in the connection network represent the possible connections (and replenishments) between flight nodes. Examples of time-space network and connection network for the same set of flight legs can be seen in figures 2.1 and 2.2 respectively.

Table 2.1. Classification of Aircraft Maintenance Related Studies Based on Their Approaches.

Aircraft Routing Focused		**Integrated**
Tactical Approach	**Operational Approach**	
Kabbani & Patty (1992)	Argüello & Bard (1997)	Barnhart et al. (1998)
Clarke et al. (1997)	Gopalan & Talluri (1998)	Sosnowska & Rolim (2001)
Gopalan and Talluri (1998)	Sriram & Haghani (2003)	Cordeau et al. (2001)
Talluri (1998)	Afsar et al. (2006, 2009)	Klabjan et al. (2002)
Mak & Boland (2000)	Sarac et al. (2006)	Cohn & Barnhart (2003)
Liang et al. (2011)	Orhan et al. (2011)	Mercier et al. (2005)
		Mercier & Soumis (2007)
		Haouari et al. (2009)
		Weide et al. (2010)

Daskin and Panayotopoulos (1989) present an ILP formulation to assign aircrafts to routes in a hub-and-spoke network with the objective of maximizing profits. They propose a Lagrangean relaxation procedure with a repair heuristic to solve the problem. Desaulniers *et al.* (1997) ignore maintenance constraints and formulate two models for integrated fleet assignment and routing problem.

Feo and Bard (1989) present a model that aims to both locate maintenance stations and develop flight schedules which better meet the cyclical demand for maintenance with a minimum cost. They formulate the problem as a min-cost, multi-commodity flow network with integrality constraints and propose a two-phase heuristic solution procedure. Their study is not related with maintenance routing problem directly; however, they try to create conditions which can yield better routes for cyclical maintenance demand.

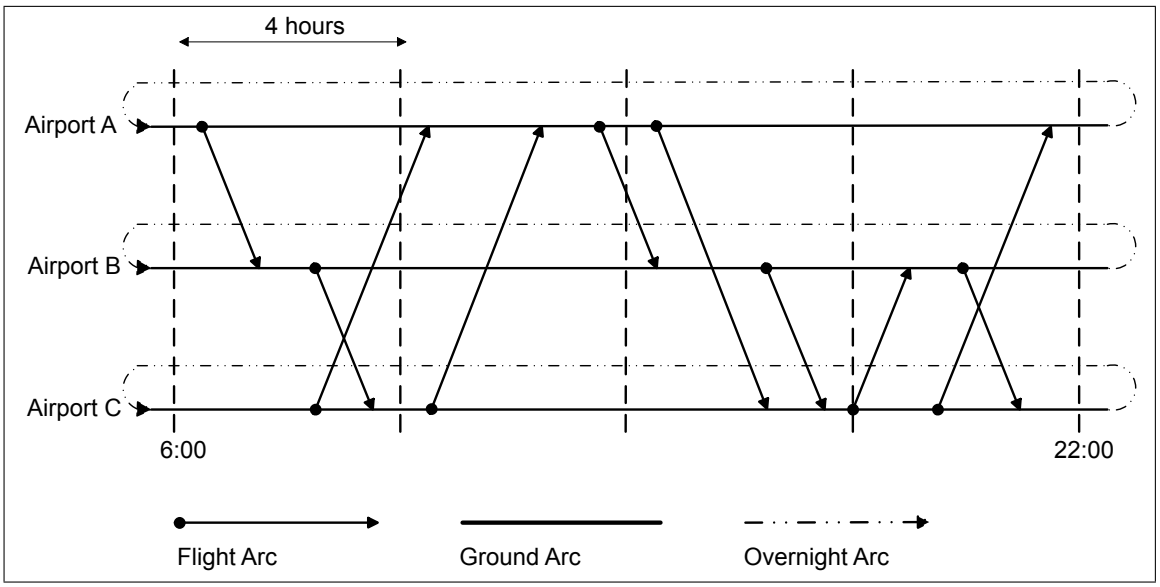

Figure 2.1. An Example of Time-space Network.

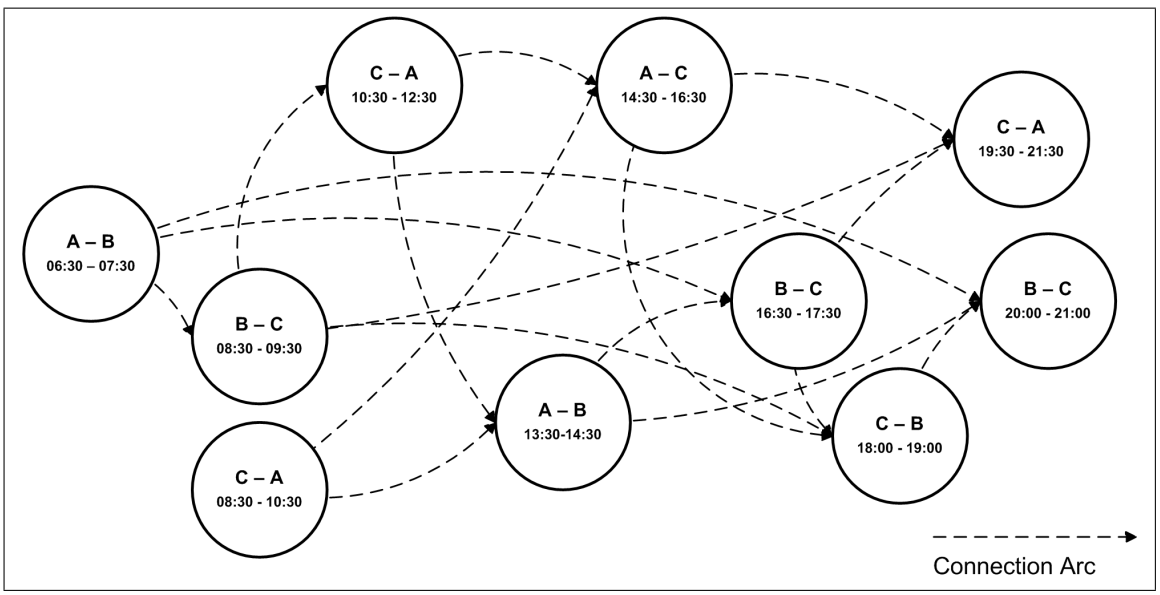

Figure 2.2. An Example of Connection Network.

Clarke *et al.* (1996) try to generalize the fleet assignment model of Hane *et al.* (1995) by considering maintenance and crew restrictions. They state that a basic fleet assignment without such considerations may lead to excessive costs at operational level. From maintenance perspective, they are trying to eliminate the shortcomings of the original fleet assignment model by enforcing sufficient number of maintenance opportunities during fleet assignment phase. In their study, the flight network is a

domestic, hub-and-spoke network and it is assumed that each flight is repeated every day. It must be noted that their solutions do not guarantee feasible intervals for maintenance operations of individual aircrafts at operational level. Such a guarantee can only be given if the corresponding routing problem is solved.

On tactical side of AMR studies, Kabbani and Patty (1992) propose a set partitioning model to solve maintenance routing problem where each aircraft needs to be checked every three days. In their set partitioning model, columns represent possible weekly routings and rows represent flights. The formulation becomes very large to solve for their case; therefore, they divide the problem into two subproblems. Assuming that maintenances are performed overnight, they first generate *over-the-day* routings (also known as *lines of flight*). Over-the-day routings are fixed daily route segments with a certain starting station in the morning and with a certain ending station in the evening. After fixing over-the-day routings, they try to connect those routings to solve their original problem.

Gopalan and Talluri (1998) use *lines of flight (LOF)* logic similar to over-the-day routings of Kabbani and Patty (1992), assuming that maintenance checks can only be done as overnight operations. They focus on *k-day maintenance routing problem* and develop a polynomial time algorithm for $k = 3$. In k-day maintenance routing problem, each aircraft has to visit a maintenance station after at most k days of flying without maintenance; in other words, an aircraft cannot select LOFs that end in a non-maintenance station for k consecutive days. In order to find 3-day maintenance routing, they first fix LOFs and construct a graph $G(V, E)$ where the set of vertices, V, represents overnight stations and the set of edges, E, represents LOFs. Then, on a transformed graph of G, G', they search for an Euler tour which satisfies 3-day maintenance requirement. In this study, 3-day maintenance routing problem is solved for both infinite-horizon and finite-horizon. The infinite-horizon version of the problem can be classified as a tactical one, on the other hand, the finite-horizon version is more operational. Additionally, they prove that without fixing LOFs, even 2-day maintenance routing problem becomes NP-Complete and Talluri (1998) shows that

even with fixed LOFs beforehand, 4-day maintenance routing problem is NP-Hard.

Clarke *et al.* (1997) state that fixing daily routes beforehand reduces the options to form rotations in hub-and-spoke networks. In their study, the aim is to find a unique, maintenance feasible rotation which yields the largest total *through value* where through value can be described as the total benefit obtained by connecting certain flights. In brief, they aim to find the Euler tour that satisfies maintenance constraints of different type and maximizes the total through value. For this purpose, they first simplify their flight network by using node and arc aggregation during preprocessing and formulate an integer programming model. As a solution procedure, a Lagrangean relaxation technique is used where subset elimination and service violation (maintenance related) constraints are relaxed.

Mak and Boland (2000) formulate AMRP as an asymmetric traveling salesman problem with *replenishment arcs* (RATSP) where a replenishment arc denotes a connection during which a maintenance operation can take place. In their study, they construct a connection network-like graph $G(V, A)$, where the set of vertices, V, represents flights with corresponding weights and the set of arcs, A, represents possible flight connections (either replenishment or ordinary) with corresponding costs. Each aircraft has a positive legal flying hour limit and a routing is feasible if it does not consist of a segment which violates that limit. Similar to Clarke *et al.* (1997), they use Lagrangean relaxation by relaxing subset elimination and maintenance violation constraints to obtain lower bounds and simulated annealing (SA) method to obtain upper bounds. Their SA method allows infeasible flight connections in solutions, but penalizes them with a large penalty cost.

Liang *et al.* (2011) consider daily check (k-day maintenance routing problem) assuming that maintenances are overnight operations and flight schedule is repeated every day. They construct a compact representation of time-space network, *Rotation Tour Network (RTN)* and propose a new integer programming formulation based on their new representation.

On operational side, Argüello and Bard (1997) propose a greedy randomized adaptive search procedure (GRASP) to reconstruct aircraft routings in response to dynamic changes in airline environment with the objective of minimizing costs incurred by flight cancellations and delays. To remain feasible, they do not change the routes of aircrafts with scheduled maintenance; however, try to change other routes (even by cancelling some flights) with minimum cost. It must be noted that the focus of the paper is not directly on maintenance.

Sriram and Haghani (2003) uses *Origin Destination (OD) pairs* similar to LOFs. Assuming that OD pairs are already given and maintenances are overnight operations, their aim is to determine individual aircraft assignments to the routes that yield the minimum total cost. In their study, they deal with two different types of maintenance checks and consider airport maintenance capacities. They propose two models: The first one assumes that a check must be done once within corresponding period (type A check must be done once within 4 days) and the second one tries to keep track of the remaining time before the next maintenance (type A check must be done after 65 hours of flying). They solve the first model by using a search technique which consists of depth-first and random search. They claim that the second model becomes too large to solve and they do not attempt to solve it. Notice that daily routes in this study are predetermined as OD pairs and such an approach results in sub-optimal solutions. They also conclude that considering flight legs rather than predetermined OD pairs yields an improvement in solution quality at a cost of increasing the problem size significantly.

Afsar *et al.* (2006) try to determine routes for aircrafts to satisfy the already scheduled long-term maintenance checks over an acyclic horizon. Their objective is a composite one: Maximization of *critical* aircraft utilization and utilization balancing. Critical aircrafts are those that have to undergo maintenance soon. Due to stochasticity, they do not fix routes several weeks ago; instead they prepare the plan based on a rolling horizon of one week. They propose a two-step heuristic approach by dividing the problem into two sub-problems where first sub-problem deals with critical aircrafts

and second deals with non-critical ones. In Afsar *et al.* (2009) different heuristic methods, including SA, and priority rules for non-critical aircraft selection are applied to the same problem. We must note that the maintenance operations of critical aircrafts are assumed to be already scheduled in these studies and with such an assumption the problem becomes easier.

Sarac *et al.* (2006) propose a set partitioning based formulation to minimize total daily maintenance costs without violating the flying hour limit of each aircraft. Their model considers maintenance resource availability (in terms of man-hour and airport slot) and legal remaining times of aircrafts. Additionally, it can handle different types of maintenance requirements. Since the set partitioning based formulation contains exponentially many variables each of which represents a feasible route, column generation technique is used in solution procedure. The time horizon of the model is one day and as stated in the study, longer time horizon can better utilize remaining times of aircrafts.

Orhan *et al.* (2011) propose an integer programming model for daily concurrent aircraft routing and maintenance scheduling. In their model, critical aircrafts that need maintenance at the end of the day are already determined by a decision maker. Their objective is maximizing the utilization of remaining time while providing daily routes ending at maintenance stations for critical aircrafts. Similar to studies of Afsar *et al.* (2006, 2009), their model does not schedule maintenance operations; it just maximizes the utilization of remaining times with respect to given maintenance information.

Gabteni and Grönkvist (2008) solve the tail assignment problem by using a combined technique which consists of column generation and constraint programming. Since tail assignment problem deals with assigning aircrafts to flights at operational level, maintenance requirements and crew considerations cannot be ignored. They start with a simple path (route) based formulation, similar to that of Sarac *et al.* (2006), and they enforce maintenance feasibility while solving the pricing problem in their column generation iterations. They try to eliminate the disadvantage of slow

convergence in column generation approach by using quick feasible solution generation ability of constraint programming.

There are also some studies which focus on solving different phases of airline scheduling simultaneously. Barnhart *et al.* (1998) construct a string-based model to solve fleet assignment and aircraft routing problem simultaneously. In their model, they consider operational constraints such as maintenance and equal aircraft utilization and propose a B&B solution approach involving column and cut generation. Sosnowska and Rolim (2001) assume that maintenances are fixed beforehand and use SA method to solve integrated fleet assignment and aircraft routing problem. Their SA move is a limited swap operation which only allows swapping segments that do not include scheduled maintenances. In a similar study, Haouari *et al.* (2009) model the integrated problem as a multi-commodity network flow problem where the planned time-window for maintenance of each aircraft is known beforehand as in Sosnowska and Rolim (2001) and they propose a two-phase network flow based heuristic to solve it. In some studies, crew scheduling and aircraft maintenance routing problems are solved simultaneously to increase cost benefits (Cohn and Barnhart , 2003; Cordeau *et al.* , 2001; Klabjan *et al.* , 2002; Mercier *et al.* , 2005; Mercier and Soumis , 2007; Weide *et al.* , 2010).

In this study, our focus will be on OAMRP. By using a modified connection network, we propose a new ILP formulation similar to multi-commodity network flow models; however, the proposed model differs from existing ones in the sense that it utilizes arc duplications in order to keep track of remaining time for each aircraft. In other words, our modified connection network contains two types of ordinary arcs and arc type indicates whether an ordinary arc is used before or after maintenance. Many heuristic methods, including SA, are applied to similar problems in literature; however, our SA based method (compressed annealing) has some structural differences when compared to traditional ones. Additionally, most of the studies focus on either short-term (daily) route assignment or long-term (weekly or longer) route assignment; however, short-term route assignments may reduce the opportunities for better utilization of remaining time and long-term ones are not very realistic due to dynamic airline

environment. In order to eliminate the disadvantages of both approaches, we propose a solution methodology based on rolling horizon.

3. OAMRP FORMULATION

In our OAMRP formulation, the aim is to find maintenance feasible routes for each aircraft in the fleet with the objective of maximizing the utilization of remaining times to decrease costs incurred by maintenance operations. Right after undergoing a maintenance operation, the remaining time of corresponding aircraft is set to legal flying hour limit imposed by the maintenance regulations and as it covers the flights on its route, the remaining time decreases by the durations of covered flights. In other words, the remaining time at the beginning of a planning horizon is the difference between legal flying hour limit and accumulated flight durations after the previous maintenance. Notice that, if an aircraft with a positive remaining time undergoes a maintenance operation, then the remaining time of that aircraft is not fully utilized. We call the amount of unused remaining time just before the maintenance as *unused legal flying time* of corresponding aircraft. Therefore, maximizing the utilization of remaining times becomes equivalent to minimization of unused legal flying times.

3.1. Modified Connection Network Structure

In order to minimize total unused legal flying time, we need to formulate a model which can keep track of the accumulated flight durations of aircrafts. Additionally, the model must be able to distinguish whether a flight is covered before a maintenance operation or after it because if a flight is covered after a certain maintenance operation, then it does not have any effect on the remaining time calculations prior to that maintenance. For these purposes, connection network representation is slightly modified by using ordinary arc duplication. The modified network still includes sets of ordinary arcs, O, and replenishment arcs, R, representing possible connections and replenishments between flight leg nodes; however, ordinary arcs of original network are duplicated so that each connection in the modified network is represented with a pair of ordinary arcs with different labels: *before maintenance* and *after maintenance*. In other words, O is duplicated, resulting in two ordinary arc sets both of which represent

exactly the same connections; however, elements in one set, O_{bm}, mean that corresponding flight connection is done before maintenance whereas those in the other set, O_{am}, mean that the connection is done after maintenance.

Other than ordinary arc duplication, the construction of modified network representation is the same as that of connection network representation. There is an ordinary arc, $ord(i, j)$, between flight leg i and flight leg j if arrival airport of flight leg i and departure airport of flight leg j are the same and there is a sufficient amount of time to prepare aircraft for the next flight between the arrival time of flight leg i and departure time of flight leg j. If the corresponding airport is a maintenance airport and the amount of time between flight legs i and j is sufficient for a maintenance operation, then there is also a replenishment arc, $rep(i, j)$, between flight legs i and j.

3.2. ILP Model Formulation

In this section, we propose a multi-commodity network flow based ILP formulation for OAMRP with remaining time consideration. The summary for definitions of sets, parameters and decision variables used in the formulation is as follows:

Sets:

I	set of flight legs
K	set of aircrafts
C	set of critical aircrafts
C'	set of non-critical aircrafts
$B(k)$	set of initial flight legs that aircraft k can fly
$S(i)$	set of flight legs that immediately succeed flight leg i
$S'(i)$	set of flight legs that immediately succeed flight leg i and that allow maintenance in between
$P(i)$	set of flight legs that immediately precede flight leg i
$P'(i)$	set of flight legs that immediately precede flight leg i and that allow maintenance in between
$\{o, t\}$	artificial source and sink nodes of flight network

Parameters:

- d_i duration of flight leg i
- D_k remaining time of aircraft k
- L length of the longest route in terms of the accumulated flight duration

Decision variables:

- x_{ijk} binary variable which takes value of 1 if aircraft k flies flight legs i and j consecutively before replenishment and 0 otherwise
- y_{ijk} binary variable which takes value of 1 if aircraft k flies flight legs i and j consecutively and undergoes maintenance in between and 0 otherwise
- z_{ijk} binary variable which takes value of 1 if aircraft k flies flight legs i and j consecutively either after replenishment or in a route with no replenishment and 0 otherwise
- r_k binary variable which takes value of 1 if aircraft k undergoes a maintenance operation within its route 0 otherwise

Sets $S(i)$, $S'(i)$, $P(i)$ and $P'(i)$ are defined as immediate successor and predecessor flight legs sets of a flight leg i by using the connections in the connection network. For instance, if $ord(i,j) \in O$ then $j \in S(i)$ and if $rep(i,j) \in R$ then $j \in S'(i)$. Decision variables x_{ijk}, y_{ijk} and z_{ijk} are simply the arcs of the modified connection network for each aircraft. Namely, x_{ijk}, z_{ijk} and y_{ijk} variables represent arcs in sets O_{bm}, O_{am} and R for each aircraft respectively. In order to decrease the number of variables in the formulation, aircrafts in the fleet are divided as *critical* (C) and *non-critical* (C') aircrafts. An aircraft k is considered to be critical if its initial remaining time, D_k, is less than the length of longest route on connection network in terms of the accumulated flight duration, L, and non-critical otherwise. In other words, a critical aircraft is an aircraft that may undergo a maintenance operation within the current planning horizon. It is obvious that a non-critical aircraft does not need a maintenance within the current planning horizon even if it is assigned to the longest route in terms of accumulated duration; therefore, it is not necessary to define x_{ijk} and y_{ijk} variables for such an aircraft. The mathematical model of OAMRP with remaining time consideration (OAMRP-ILP) is as follows:

OAMRP-ILP:

$$min \sum_{k \in C} \left(r_k D_k - \sum_{i \in I} d_i \left(\sum_{j \in S(i)} x_{ijk} + \sum_{j \in S'(i)} y_{ijk} \right) \right) \tag{3.1}$$

s.t.

$$\sum_{j \in B(k)} (x_{ojk} + y_{ojk} + z_{ojk}) = 1 \qquad k \in C \tag{3.2}$$

$$\sum_{j \in B(k)} z_{ojk} = 1 \qquad k \in C' \tag{3.3}$$

$$\sum_{i \in I} z_{itk} = 1 \qquad k \in K \tag{3.4}$$

$$\sum_{j \in P(i) \cup \{o\}} x_{jik} = \sum_{j \in S(i)} x_{ijk} + \sum_{j \in S'(i)} y_{ijk} \qquad i \in I, k \in C \tag{3.5}$$

$$\sum_{j \in P'(i) \cup \{o\}} y_{jik} + \sum_{j \in P(i) \cup \{o\}} z_{jik} = \sum_{j \in S(i) \cup \{t\}} z_{ijk} \qquad i \in I, k \in C \tag{3.6}$$

$$\sum_{j \in P(i) \cup \{o\}} z_{jik} = \sum_{j \in S(i) \cup \{t\}} z_{ijk} \qquad i \in I, k \in C' \tag{3.7}$$

$$\sum_{k \in K} \left(\sum_{j \in P(i) \cup \{o\}} (x_{jik} + z_{jik}) + \sum_{j \in P'(i) \cup \{o\}} y_{jik} \right) = 1 \qquad i \in I \tag{3.8}$$

$$\sum_{i \in I} d_i \left(\sum_{j \in S(i)} x_{ijk} + \sum_{j \in S'(i)} y_{ijk} \right) \leq D_k \qquad k \in C \tag{3.9}$$

$$\sum_{i \in I} d_i \left(\sum_{j \in S(i) \cup \{t\}} z_{ijk} \right) \leq D_k \sum_{i \in I} z_{oik} + L \left(1 - \sum_{i \in I} z_{oik} \right) \qquad k \in C \tag{3.10}$$

$$r_k = \sum_{i \in I \cup \{o\}} \sum_{j \in S(i)} y_{ijk} \qquad k \in C \tag{3.11}$$

$$x_{ijk}, y_{ijk}, z_{ijk} \in \{0, 1\} \qquad i, j \in I, k \in K \tag{3.12}$$

$$r_k \in \{0, 1\} \qquad k \in K \tag{3.13}$$

The objective function (3.1) is the minimization of total unused legal flying time of critical aircrafts. It must be noted that noncritical aircrafts and critical aircrafts that do not undergo maintenance within current planning horizon do not contribute to objective function because unused legal flying time can only occur when aircrafts undergo a maintenance operation. Constraints (3.2) and (3.3) ensure that each aircraft in the fleet starts its route by flying one of the possible initial flight legs. Similar to (3.2) and (3.3), constraints (3.4) ensure the completion of routes.

Before discussing balance constraints (3.5), (3.6) and (3.7), it must be noted that a feasible route in this network can be constructed in two different ways. Firstly, if a critical aircraft has a maintenance on its route, it must use ordinary arcs in O_{bm} to connect the flight legs until the maintenance, then a replenishment arc in R and then ordinary arcs in O_{am} to connect the remaining flight legs after maintenance (note that a critical aircraft can choose to start directly with a replenishment arc; in this case none of the ordinary arcs in O_{bm} is used). Secondly, if an aircraft does not have a maintenance within current planning horizon (not necessarily a non-critical aircraft), then the entire route can be constructed by using ordinary arcs in O_{am}. The reason why the ordinary arcs in O_{am} are used becomes apparent when our remaining time calculation in the objective function is examined. The ordinary arcs in O_{am} are represented by z_{ijk} variables in the model and these variables have no effect in remaining time calculations. This property serves to our aim perfectly because aircrafts with no maintenance operations within current planning horizon have no effect on remaining time calculations.

Constraints (3.5) and (3.6) are the balance constraints for critical aircrafts. Constraints (3.5) indicate that if an aircraft covers flight leg i before maintenance, then the next flight leg covered by that aircraft must be either before maintenance again or right after maintenance. In other words, knowing that flight leg i is connected to its predecessor flight leg with some $ord(j, i) \in O_{bm}$, it must be connected to its successor flight leg with either some $ord(i, j') \in O_{bm}$ or with some $rep(i, j') \in R$. Similar to (3.5), constraints (3.6) indicate that if an aircraft covers flight leg i after maintenance,

then the next flight covered by that aircraft must be after maintenance again. It must be noted that these balance constraints do not enforce a critical aircraft to undergo a maintenance operation, it can still choose not to undergo maintenance by setting x_{ijk} and y_{ijk} variables to 0. Constraints (3.7) are the balance constraints for non-critical aircrafts. Since non-critical aircrafts do not undergo maintenance within the planning horizon, x_{ijk} and y_{ijk} variables are not defined for them and balance constraints (3.5) and (3.6) reduce to constraints (3.7).

Constraints (3.8) are flight coverage constraints, indicating that each flight must be covered by an aircraft. Constraints (3.9) and (3.10) are legal flying hour limit constraints for critical aircrafts. Constraints (3.9) are for critical aircrafts that undergo maintenance, indicating that the accumulated flight durations until maintenance cannot be larger than the remaining time of the aircraft, D_k. For critical aircrafts that have no maintenance operations on their route, these constraints are redundant ones. Constraints (3.10) are for critical aircrafts that do not undergo maintenance. For such an aircraft, the accumulated flight duration for the entire route should be less than D_k. This constraint becomes redundant for aircrafts with a maintenance operation on their routes because in this case, the right hand side of the constraint becomes L, which is the length of longest route in terms of accumulated duration for given network. There is no need to define such constraints for non-critical aircrafts because by definition their remaining times cannot be violated within the current planning horizon. Constraints (3.11) are coupling constraints between variables r_k and y_{ijk}; namely if an aircraft uses a replenishment arc within its route then this means that it undergoes maintenance. Note that r_k variable can be removed from the model by replacing the r_k term in the objective with the right hand side of (3.11); however, r_k has a physical meaning in variable prioritization for B&B; therefore, we decide to keep it in the model. Constraints (3.12) and (3.13) are integrality constraints on variables.

In our formulation, a critical aircraft can undergo a maintenance operation at most once within the planning horizon. Therefore, legal flying hour limit, which is imposed by maintenance regulation, can be considered as an upper bound on L because it

is not possible to guarantee the existence of a feasible solution with single maintenance for each aircraft in networks where L is larger than legal flying hour limit.

4. SOLUTION METHODS

Before introducing the solution methods, we must comment on the hardness of OAMRP. Basically, OAMRP includes the partition problem, where $|I|$ many flight legs are covered by $|K|$ many routes exactly once, with some additional constraints. The partition problem is known to be NP-Complete; hence, OAMRP is NP-Hard (Sarac *et al.* , 2006).

4.1. Exact Method: B&B with Variable Prioritization

Although OAMRP is NP-Hard and exact methods are not likely to find effective solutions for large scaled instances, it is important to solve moderate instances to optimality in order to have a base for evaluation of heuristic method performance. In this study, the optimal solution to OAMRP is obtained by B&B using a general-purpose integer programming solver like IBM ILOG CPLEX . During branching, CPLEX uses a combination of predefined rules in an automated manner to decide on the variable to branch. However, giving higher branching priorities to the variables in certain sets may reduce the solution times of some ILP problems significantly.

Assigning higher priority order to some set of variables means that the variables in that set must be decided before the other variables of the problem. In other words, variables in sets with higher priority order are branched on (i.e. fixed) at higher levels of B&B tree. Therefore, assigning higher priority orders to the set of variables which activate the other sets of variables is a reasonable decision (User's Manual for CPLEX , 2009).

In the scope of OAMRP, r_k and y_{ijk} variables can be considered as *activating* variables. As discussed in Section 3.2, decision variable r_k indicates whether aircraft k undergoes maintenance during the planning horizon. Notice that if we branch on r_k variables at the higher levels of B&B tree, we gain important information regarding

x_{ijk} and y_{ijk} variables. For instance, at child nodes above which $r_{k'}$ variable is set to 0, values of $x_{ijk'}$ and $y_{ijk'}$ variables directly become 0 due to constraints (3.5) and (3.11); in addition, constraints (3.9) become redundant. Similar to r_k variables, y_{ijk} variables can also be branched on at higher levels of B & B tree because by doing so, we can gain information about when the maintenance operation of aircraft k is going to occur. For instance, when $y_{i'j'k'}$ is set to 1 at some branch, we know that aircraft k' is going to undergo maintenance between flight legs i' and j'. With this information, we know that we will be connecting flight legs by using $x_{ijk'}$ variables until flight leg i' and $z_{ijk'}$ variables after flight leg j' due to constraints (3.5) and (3.6). Hence, at child nodes which emanate from such a branch, values of $x_{ijk'}$ variables connecting flight legs that are after flight leg j' and $z_{ijk'}$ variables connecting flight legs that are before flight leg i' become 0 for aircraft k'.

For these purposes, three prioritization schemes are defined and tested under different instances. In our first prioritization scheme, we assign high priority to all r_k variables and call this scheme as $r - prio$. In the second one, we assign high priority to all r_k variables and $y_{ijk'}$ variables where $i, j \in I$ and k' being the index of *most critical* aircraft. By most critical, we simply mean that the aircraft with the least remaining time. The second scheme is called as $r - y_{crit} - prio$. Finally, we assign high priority to all r_k and all y_{ijk} variables; and we call this scheme as $r - y_{all} - prio$. These prioritization schemes are also compared with the default branching scheme of CPLEX.

4.2. Heuristic Method: Compressed Annealing

Frequently occurring stochastic events in airline industry may affect the states of the aircrafts in the fleet and such events generally increase operational costs or even create infeasibilities in the existing route assignments. We may need to solve OAMRP very frequently in order to respond to such changes; therefore, developing a fast and responsive solution methodology becomes an essential task. For this purpose, we propose a heuristic solution procedure based on compressed annealing for OAMRP.

Introduced by Kirkpatrick *et al.* (1983) and Cerny (1985), simulated annealing (SA) is a probabilistic metaheuristic analogous to annealing in metallurgy. In simple terms, SA is a probabilistic local search method that allows accepting worse neighboring solutions with a certain probability. The probability of accepting worse solutions is directly proportional to temperature of annealing. At high temperatures SA behaves as random search; however, as iterations go on, the temperature decreases and SA converges to local search. One can refer to Dowsland (1993) for further reading on simulated annealing.

During SA iterations for a minimization problem, infeasible solutions can be penalized to have significantly large objective values compared to feasible ones. The amount of penalty is generally determined by multiplying a constant penalty multiplier, ν, and the amount of infeasibility. However, determining the value of ν is not an easy task. Setting ν to very large values is a meaningful option to avoid infeasibilities; but such a decision makes it more difficult for SA algorithm to explore solution space. Setting ν to smaller values, on the other hand, can remedy the exploration problem; however, in this case, SA algorithm may end up with an infeasible solution. Therefore, in the scope of OAMRP; rather than using SA with a constant penalty multiplier, we can focus on an extension of it: *Compressed Annealing* (CA). CA is a metaheuristic based on SA where infeasibilities are penalized with a variable penalty multiplier which is called as *pressure* (Morse , 1997; Theodoracatos and Grimsley , 1995). Both temperature (τ) and pressure (λ) are adjusted simultaneously during the iterations of CA. At the beginning, the algorithm starts with a high temperature and a low pressure setting in order to move freely within the solution space. As the iterations go on, with decreasing temperature and increasing pressure values, CA starts to reject infeasible solutions and converges to local search. In a recent study, Ohlmann and Thomas (2007) use CA for TSP with time windows and show that CA generally performs better than SA with static penalty multiplier for their problem. Additionally, a detailed discussion on convergence in probability of CA can be found in the study of Ohlmann *et al.* (2004). During the construction of our CA procedure, we benefit from the results of these studies.

The details regarding the application of CA to OAMRP are as follows:

Solution Structure: The solution structure used in CA iterations consists of $|K|$ many flight strings, denoting the routes of aircrafts. Namely, k^{th} flight string is the route which is assigned to aircraft k. In our solution structure, the routes are always connection feasible. In other words, all the flights in the flight strings are connected to their immediate successors with arcs of any type.

Neighboring Solution Generation: For the generation of neighboring solutions, we utilize the *swap* move which basically swaps the tails of given routes without violating the connection feasibility of the routes. During the swap move, we first select two flight strings, route 1 and route 2, from the current solution randomly. Then, from route 1, we again randomly select successive flight legs i and j. After that, we try to find successive flight legs i' and j' of route 2 which satisfy the following conditions: $j' \in S(i) \cup S'(i)$ and $j \in S(i') \cup S'(i')$. If we can find such successive flights, we swap the tails of route 1 and 2, else we repeat this procedure until we find a neighboring solution. Notice that this swap operation always creates connection feasible routes. An example of swap move can be seen in Figure 4.1.

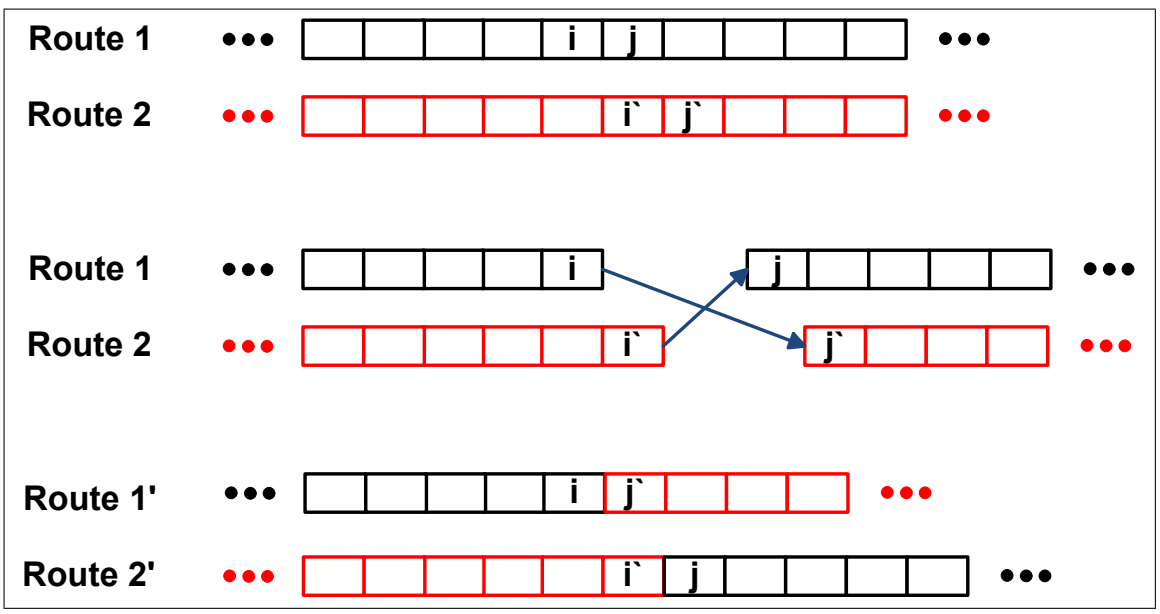

Figure 4.1. An Example of Swap Move.

Initial Solution: We can obtain a connection feasible initial solution by solving a simplified version of ILP model. For the simplified version, we ignore the objective, remove constraints (3.2), (3.5), (3.6), (3.9), (3.10), (3.11) and write constraints (3.3) and (3.7) for all aircrafts rather than non-critical ones. As a result of the simplified model, we obtain $|K|$ many connection feasible routes (not necessarily maintenance feasible) and use them as flight strings of the initial solution.

Determination of τ_0 and λ_{max}: Determination of *meaningful* initial temperature (τ_0) and maximum pressure (λ_{max}) values is an essential issue in CA because these values directly affect the way how the heuristic behaves while exploring the solution space. Since the initial conditions of aircrafts formalize the structure of the solution space in OAMRP, it is a wise decision to determine τ_0 and λ_{max} values with respect to them, rather than assigning constant values to these parameters for all cases. For this purpose, at the beginning of the algorithm, we generate n many solution pairs, $\mathscr{F}$, and calculate the absolute differences, $|\Delta f|_p$, between the objective values of solutions in each pair. Then we calculate initial temperature as follows (Dowsland , 1993):

$$\tau_0 = \frac{\overline{|\Delta f|}}{ln\left(\frac{1}{X_0}\right)} \tag{4.1}$$

In equation 4.1, $\overline{|\Delta f|}$ is the average of individual absolute differences and X_0 is the acceptance ratio at τ_0.

Before discussing the determination of λ_{max}, we must introduce the objective function structure of a solution s. The objective function, $f_s(\lambda)$, is a composite objective function with two components, where the first component, g_s, represents the amount of unused legal flying time and the second one, h_s, represents the number of maintenance infeasible aircrafts:

$$f_s(\lambda) = g_s + \lambda h_s \tag{4.2}$$

In their study, Ohlmann *et al.* (2004) discuss the existence of a pressure cap, λ^*. Basically, λ^* is the minimum amount of pressure which is sufficient to represent solution topography's features. However, it is not easy to find a tight upper bound for λ^*; hence, the approximation proposed by Ohlmann and Thomas (2007) can be used:

$$\lambda_{max} = \max_{s \in \mathscr{F}} \left\{ \frac{g_s}{h_s} \frac{\kappa}{(1-\kappa)} \right\} \tag{4.3}$$

In equation 4.3, κ denotes the ratio of penalty term to objective function value when pressure is λ_{max}. It must be noted that this approximation does not guarantee that λ_{max} is an upper bound on λ^*.

Cooling and Compression: The performance of CA heuristic is strongly dependent to temperature and pressure updating schedules, also known as *cooling* and *compression*. Ohlmann *et al.* (2004) state that the convergence to the set of global minima can be ensured by cooling and compression schedules that have decreasing derivatives. For this purpose, we utilize the geometric cooling and limited exponential compression schedule (Ohlmann and Thomas , 2007) with parameters $0 \leq \beta \leq 1$, $\gamma \geq 0$, $\lambda_0 \geq 0$ and $\lambda_{max} > 0$:

$$\tau_{c+1} = \beta \tau_c \tag{4.4}$$

$$\lambda_{c+1} = \lambda_{max}\left(1 - \frac{(\lambda_{max} - \lambda_0)}{\lambda_{max}} e^{-\gamma c}\right) \tag{4.5}$$

Cycle Length: Cycle length, which is the number of iterations per CA cycle, is another important determinant regarding the performance of CA. In our CA runs, we start iterations with a predetermined initial cycle length and at the end of each cycle, i.e. at each temperature and pressure update, we increase it by a certain percentage which we call as cycle length increment.

Calculation of a Solution's Objective Value: The way we should calculate the objective value of a solution strongly depends on the assumption about the airport maintenance capacities. The airport maintenance capacity is a restriction on the maximum number of aircrafts that can undergo maintenance simultaneously for a certain airport. Assuming that the airport maintenance capacities are ignored, individual unused legal flying times and infeasibilities of aircrafts can be computed independently to calculate g_s and h_s components of objective function. Under such an assumption, every aircraft can independently select the best maintenance option among all other options within its route without creating any infeasibilities regarding the airport maintenance capacities (note that the capacitated case will be discussed in Section 6.1).

The calculation of unused legal flying time for an aircraft is trivial. First of all, we know that there is a maintenance opportunity between successive flight legs i and j if set R contains the replenishment arc $rep(i, j)$ and by checking all successive flight legs within the route of corresponding critical aircraft, we determine all possible maintenance opportunities. Additionally, if the initial airport of the considered aircraft is a maintenance airport, then there is a maintenance opportunity at the beginning of the route (these kind of opportunities correspond to y_{ojk} variables in the ILP formulation). At this stage, two different cases can be observed: Either the maintenance opportunity which yields the least unused legal flying time is assigned to the corresponding aircraft

or no such opportunity can be found and the aircraft becomes maintenance infeasible. One must note that, even an aircraft with positive number of maintenance opportunities can become maintenance infeasible when the accumulated flight duration until the first opportunity is greater than the initial remaining time. Finally, the sum of maintenance feasible aircrafts' unused legal flying time constitutes g_s component, the number of maintenance infeasible aircrafts constitutes h_s component and composite objective function is calculated by using the current pressure value λ.

Termination Criterion: The CA algorithm stops when there is no improvement in the best feasible solution for a certain number of cycles and the cycle count exceeds a predetermined limit for minimum number of cycles to execute. The algorithm also stops if the objective value of a feasible solution turns out to be 0 because 0 is a lower bound for OAMRP and a feasible solution with this value is surely optimal.

5. NUMERICAL RESULTS AND PERFORMANCE OF SOLUTION METHODS

5.1. Experimental Setting

We perform computational experiments on a domestic flight network of a commercial airline company. The flight network consists of 354 flights per week and an 8-aircraft fleet is assigned to cover these flights. Technically, the flight network includes 354 nodes, 2852 ordinary arcs and 1900 replenishment arcs. Our aim is to determine weekly routes for the aircrafts in the fleet with minimum unused flying time. In terms of accumulated flight duration, the longest route of flight network lasts 4885 minutes; i.e. $L = 4885$ *mins*.

The test cases mainly differ from each other in terms of the number of critical aircrafts, $|C|$. In our test cases, $|C|$ takes on three different values: $|C| = 1$, $|C| = 3$ and $|C| = 5$. For each $|C|$ value, we create ten different test cases by assigning randomly determined remaining times to arbitrarily selected $|C|$ many aircrafts in the fleet. The remaining time, D_k, assigned to a critical aircraft $k \in C$, can take on values between 0 and 3000 minutes as multiples of five. We select such an increment value because all flight durations are multiples of five minutes. In addition, notice that any D_k value less than L could be used as remaining time of a critical aircraft; however, we do not allow D_k to be larger than 3000 minutes in order to create tighter instances.

To summarize, we have 30 test cases, where test cases 1-10, 11-20 and 21-30 have one, three and five critical aircrafts respectively. For each test case, critical aircrafts and their corresponding remaining times are determined randomly.

As discussed in Section 4.1, we employed IBM ILOG CPLEX 11.0 in our exact method runs and CA algorithm is coded in C# environment of Microsoft Visual Studio 2010. All the experiments are carried out on a PC with a 2.33 GHz CPU running under

64-bit Windows Vista operating system. Maximum memory usage of CPLEX is limited to 4.00 GB and a time limit of five hours is alloted for exact method runs.

5.2. Exact Method Results

The summary of exact method results for different prioritization schemes can be seen in Table 5.1. For each test case, we report total unused legal flying time (in minutes) as well as the required CPU time (in seconds) under all prioritization schemes. The table also provides optimal values for all test cases except for cases 22 and 30. For these two cases, the provided values are the best results obtained at the end of a 1-week B&B run. Z_{best} values in *Best* column are the best of the objective values found among four prioritization schemes. Notice that the exact methods do not necessarily find optimal solutions due to the time limit of five hours.

The optimality gaps are not reported in Table 5.1 because for most of the cases lower bound values are stuck at the initial lower bound value of 0. In order to obtain high quality lower bounds, we applied Lagrangean relaxation to OAMRP and obtained aircraft separable problems by relaxing the constraints (3.8). Unfortunately, Lagrangean relaxation procedure failed to provide high quality lower bounds at a reasonable amount of time.

Before discussing the individual performances of prioritization schemes, we must comment on the difficulty of test cases. By looking at required solution times, we can clearly see that OAMRP becomes harder to solve as the number of critical aircrafts increases. This is an expected behavior because the number of x_{ijk} and y_{ijk} variables are directly proportional to the number of critical aircrafts.

On the basis of the results presented in Table 5.1, we can compare the performances of prioritization schemes. First of all, when the averages of objective values are investigated, we can say that the default branching scheme of CPLEX performs better than all prioritization schemes on the average. However, the differences in av-

Table 5.1. Exact Run Summaries for 8-aircraft Fleet.

Case	Optimal[a]	Best	$default$		$r-prio$		$r-y_{crit}-prio$		$r-y_{all}-prio$	
	Z^*	Z_{best}	Z	CPU (s)	Z	CPU (s)	Z	CPU (s)	Z	CPU (s)
Case 1	0	0	0	1879.38	0	1762.7	0	765.64	0	770.61
Case 2	0	0	0	130.23	0	127.14	0	128.03	0	127.42
Case 3	0	0	0	62.83	0	54.23	0	55.2	0	55.34
Case 4	35	35	35	34.91	35	30.48	35	31.61	35	30.66
Case 5	0	0	0	114.64	0	96.92	0	96.27	0	96.08
Case 6	0	0	0	4047.61	0	210.95	0	871.05	0	873.16
Case 7	105	105	105	31.84	105	29.91	105	29.75	105	29.69
Case 8	0	0	0	707.06	0	674.31	0	799.88	0	802.66
Case 9	0	0	0	1382.81	0	1212.13	0	1897.86	0	1900.97
Case 10	0	0	0	432.44	0	362.52	0	360.5	0	362.8
Avg. 1-10	14	14	14	882.38	14	456.13	14	503.58	14	504.94
Case 11	0	0	0	1302.13	0	1249.8	0	1019.5	0	3663.88
Case 12	10	10	10	18000	10	18000	10	3550.58	10	16972.41
Case 13	0	0	5	18000	15	18000	0	15255.98	0	736.81
Case 14	50	50	50	802.81	50	5091.52	50	5069.69	50	881.2
Case 15	0	0	15	18000	0	3310.03	0	5621.36	15	18000
Case 16	95	95	95	3963.91	95	422.11	95	420.83	95	8097.67
Case 17	0	0	0	1293.66	5	18000	0	600.42	0	7799.61
Case 18	50	50	50	510.72	50	847.09	50	845.03	50	2201.56
Case 19	0	0	0	15004.03	5	18000	40	18000	5	18000
Case 20	0	10	15	18000	25	18000	10	18000	10	18000
Avg. 11-20	20.5	21.5	24	9487.73	25.5	10092.06	25.5	6838.34	23.5	9435.31
Case 21	35	50	65	18000	80	18000	50	18000	90	18000
Case 22	5[b]	70	70	18000	75	18000	130	18000	**2770**	18000
Case 23	80	80	90	18000	80	14295.95	95	18000	135	18000
Case 24	115	115	135	18000	115	17487.91	115	14881.5	125	18000
Case 25	0	30	85	18000	65	18000	30	18000	40	18000
Case 26	0	0	40	18000	90	18000	180	18000	0	17875.58
Case 27	130	130	135	18000	130	4464.69	140	18000	165	18000
Case 28	0	50	55	18000	**2100**	18000	**380**	18000	50	18000
Case 29	145	145	145	8679.08	145	7962.47	145	7978.45	145	10534.91
Case 30	125[b]	125	130	18000	150	18000	140	18000	125	18000
Avg. 21-30	64	79.5	95	17067.91	303	15221.1	140.5	16686	364.5	17241.05
Grand Avg.	32.83	38.33	44.33	9146	114.17	8589.76	60	8009.3	134	9060.43
Rev. Avg.	34.82	36.79	43.04	8513.57	44.64	7917.6	46.07	7295.68	42.86	8421.89

[a] The optimal values are obtained by using CPLEX without time limit. In these runs, the best solutions are provided to CPLEX as initial solutions.

[b] B&B procedure of CPLEX is terminated due to excessive memory usage after 1 week. Reported results are the best ones obtained at the end of a 1-week B&B run.

erages are mainly caused by cases 22 and 28 because all prioritization schemes we defined perform poorly in one of these two cases. Although this implies that the default branching scheme of CPLEX is more robust to changes in instances, we discard cases 22 and 28 and calculate the average of remaining 28 cases, which we call as *revised average*, for further comparison. The differences between revised averages become insignificant compared to previous differences; hence, we cannot claim that any prioritization scheme is significantly better than the default one. On the other hand, default branching scheme ends up with Z_{best} values for 19 of 30 cases whereas this number is 20 for $r - prio$, 22 for $r - y_{all} - prio$ and 23 for $r - y_{crit} - prio$ scheme. In fact, this supports our robustness claim about default branching scheme of CPLEX: Although it is the worst scheme in terms of the number of best solutions found, it is the best one when averages are considered.

In cases 1-10 all prioritization schemes find the optimum solutions; however, default branching scheme performs worse than the other three in terms of required CPU time. For the remaining test case sets (11-20 and 21-30), it is not very meaningful to compare required CPU times of different schemes because the best solution performances of different schemes vary for most of the test cases in these sets. On the other hand, when we calculate the averages of required CPU times of 16 cases for which the best solution performances of all prioritization schemes are the same, the average required CPU time for $default$, $r - prio$, $r - y_{crit} - prio$ and $r - y_{all} - prio$ schemes are 2630.15, 2383.39, 1494.99 and 2962.56 seconds respectively.

To summarize, the default branching scheme of CPLEX produces more stable results and generally requires longer CPU time than other branching schemes. Among the other schemes, $r - y_{crit} - prio$ provides the best solutions for many cases in shorter CPU times; however, its performance is poor on the average due to low quality solutions in cases with high number of critical aircrafts (cases 21-30). Performances of $r - prio$ and $r - y_{all} - prio$ are very similar to the default scheme and they are (especially $r - y_{all} - prio$) likely to find the best solutions for many cases; however, their robustness is arguable since they perform dramatically poor on cases 28 and 22 respectively.

5.3. CA Results

Determination of a good parameter setting is the key factor in the success of all heuristic methods. For this purpose, we carried out a preliminary analysis in order to determine the values of parameters for CA runs. The setting presented in Table 5.2 seems to perform well under different instances.

Table 5.2. Parameter Settings for CA.

Parameter	**Setting**
Cooling coeff. (β)	0.90
Compression coeff. (γ)	0.10
Initial acceptance ratio (X_0)	0.99
Pressure cap ratio (κ)	0.99
Initial cycle length	60
Cycle length increment	1%
Minimum number of cycles to execute	100

The summary of ten CA replications can be seen in Table 5.3 along with the optimal values and the results of the best prioritization schemes for the corresponding cases. In the table, Z_{best} and $\overline{Z}$ columns represent the best result found during replications and the average of these replications, respectively. The standard deviation of replication results are reported in σ_Z column. Additionally, $\overline{CPU}$ and $\overline{CPU}_f$ columns show the averages of total solution and first feasible solution times of replications.

Similar to the exact method runs, we observe the increasing trend in $\overline{CPU}$ and $\overline{CPU}_f$ as the number of critical aircrafts increases. We must also note that σ_Z values increase as the number of critical aircrafts increases. This is not an unexpected behavior because as the number of critical aircrafts increases, the search space becomes harder to explore due to higher number of infeasibilities. As a result, high quality solutions may not be observed as frequently as they are in easier test cases.

Table 5.3. Compressed Annealing Run Summaries for 8-aircraft Fleet.

Case	Optimal	Best of Exact Runs		Compressed Annealing Replication Summary				
	Z^*	Z	$\overline{CPU}$ (s)	Z_{best}	$\overline{Z}$	σ_Z	$\overline{CPU}(s)$	$\overline{CPU}_f(s)$
Case 1	0	0	765.64	0	1.5	3.37	40.7	2.22
Case 2	0	0	127.14	0	0	0	15.06	2.44
Case 3	0	0	54.23	0	0	0	25.4	3.14
Case 4	35	35	30.48	35	35	0	79.91	2.25
Case 5	0	0	96.08	0	0	0	20.41	2.1
Case 6	0	0	210.95	0	0	0	21.86	2.3
Case 7	105	105	29.69	105	105	0	80.6	2.18
Case 8	0	0	674.31	0	1.5	3.37	29.22	3.03
Case 9	0	0	1212.13	0	6	5.16	68.92	2.23
Case 10	0	0	360.5	0	0	0	5.66	2.29
Avg. 1-10	14	14	356.12	14	14.9	1.19	38.77	2.42
Case 11	0	0	1019.5	0	2	3.5	84.83	2.28
Case 12	10	10	3550.58	10	21.25	16.2	122.47	45.39
Case 13	0	0	736.81	0	3.5	4.12	96.69	3.04
Case 14	50	50	802.81	50	56.5	7.47	122.95	2.88
Case 15	0	0	3310.03	0	8	7.89	106.61	9.11
Case 16	95	95	420.83	95	99.5	5.5	117.09	2.53
Case 17	0	0	600.42	0	6.5	5.8	101.94	2.4
Case 18	50	50	510.72	50	50.71	1.89	107.21	25.66
Case 19	0	0	15004.03	0	13	13.51	108.4	35.16
Case 20	0	10	18000	5	25.5	23.15	118	8.78
Avg. 11-20	20.5	21.5	4395.57	21	28.65	8.90	108.62	13.72
Case 21	35	50	18000	40	55.71	12.05	156.64	36.71
Case 22	5	70	18000	10	91	67.32	167.56	3.6
Case 23	80	80	14295.95	95	105	10	118.92	21.92
Case 24	115	115	14881.5	115	129.29	16.94	161.83	11.03
Case 25	0	30	18000	20	42.5	24.64	143.77	3.23
Case 26	0	0	17875.58	0	57	38.96	159.78	3.92
Case 27	130	130	4464.69	130	135	6.61	154.87	33.82
Case 28	0	50	18000	0	34.5	28.33	151.01	2.73
Case 29	145	145	7962.47	145	156	10.84	141.79	42.38
Case 30	125	125	18000	140	178.75	46.04	144.72	24.26
Avg. 21-30	64	79.5	14948.02	69.5	98.48	26.17	150.09	18.36
Grand Avg.	32.83	38.33	6566.57	34.83	47.34	12.09	99.16	11.5

Before discussing and comparing the performances of the methods further, we must comment on the total solution time of CA runs. Note that $\overline{CPU}$ column in Table 5.3 reports the average duration per CA replication. If we are discussing performance of CA by considering the best results obtained at the end of ten replications, we must multiply the $\overline{CPU}$ column by ten in order to obtain the overall time required to find the best solution. For instance, for the test case 21, the best objective value of 40 is assumed to be found in $156.64 \times 10 = 1566.4$ seconds because even if we find 40 at the end of first replication, we have to wait until the end of tenth replication to claim that it is our best solution for the test case 21.

When we consider the best results found during replications, CA method finds optimal solutions for 26 of 30 test cases. Furthermore, it performs slightly worse than the result of the best prioritization scheme only on the cases 23 and 30. In addition to these, by looking at average solution quality, $\overline{CPU}$ and $\overline{CPU}_f$ times, we can claim that CA provides the responsiveness we need in the airline environment because it finds high quality solutions in a very short amount of time compared to exact methods we have tried. For further discussion, we report average objective values of different methods in Table 5.4 along with the average optimality gaps and required solution times. Apart from all cases, we provide the same statistics by excluding cases 22 and 28 since some branching schemes perform significantly poor in these two cases. Notice that in the table, *CA avg* row is the average performance of a compressed annealing replication whereas *CA best* row represents the best performance among ten replications; hence, overall solution time of *CA best* is exactly ten times that of *CA avg*.

CA performs (especially when *CA best* is considered) much better than all prioritization schemes that are tried in terms of both solution quality and required solution time. In these 30 test cases, as seen in *CA avg* row of Table 5.4, an average CA replication lasts 1.6 minutes and returns a solution which is almost the same as those of exact runs in terms of solution quality (25-30% optimality gap). We are able to decrease the optimality gaps to 5-6% in 16 minutes simply by replicating this procedure ten times. Additionally, CA seems to be robust under all different test cases although one

Table 5.4. Summarized Performances of the Methods.

		$\overline{\overline{Z}}$	*gap (%)*	$\overline{CPU}(s)$
all cases	*optimal*	32.83	-	-
	default	44.33	25.94	9146
	r-prio	114.17	71.24	8589.76
	r-crit-prio	60	45.28	8009.3
	r-yall-prio	134	75.5	9060.43
	CA avg	47.34	30.65	99.16
	CA best	34.83	5.74	991.61
except 22 & 28	*optimal*	34.82	-	-
	default	43.04	19.1	8513.57
	r-prio	44.64	22	7917.6
	r-crit-prio	46.07	24.42	7295.68
	r-yall-prio	42.86	18.76	8421.89
	CA avg	46.23	24.68	94.87
	CA best	36.96	5.79	948.66

or two replications out of ten may terminate without finding any solutions in some rare instances.

We must note that an 8-aircraft fleet is not large scaled enough to discuss the solution quality and speed of CA; however, with the help of such small or moderate scaled instances, we are able to compare our heuristic methods with exact methods and optimal solutions. In most of the large scaled instances, exact methods fail to find even feasible solutions due to difficulty of the problem and the performance gap between heuristic and exact methods becomes immeasurable. Therefore, comparing a heuristic method with an exact one in large scaled instances is not meaningful since it always favors the heuristic method (especially if the heuristic method is able to find feasible solutions in a short amount of time). On the other hand, testing the heuristic methods in large scaled instances is necessary in order to see their applicability in real life. For this purpose, in Section 6.1, we perform computational experiments on a 20-aircraft

fleet with additional constraints and comment on the scalability and applicability of CA.

6. MODIFICATIONS TO OAMRP

Since we are dealing with *operational* aircraft maintenance routing problem, operational level constraints related to airport maintenance capacities and satisfaction of already booked maintenances must be taken into account. Since the proposed ILP formulation of OAMRP and CA-based solution procedure ignore such restrictions, we propose some modifications to existing ILP formulation and heuristic solution method.

6.1. Capacity Considerations

The maintenance capacity of an airport is a restriction due to hangar availability and it limits the number of aircrafts that can undergo maintenance simultaneously at the corresponding airport. In most of the previous studies, maintenances are assumed to be overnight operations and with such an assumption maintenance capacity constraints can be constructed by limiting the number of replenishment arcs to be used during nighttime. Although this approach may result in underestimation of real hangar capacities in our case, we construct the capacity constraints in the same way due to various advantages it provides.

Since we are not enforcing specific time windows for maintenance operations, merely limiting the number of intersecting replenishment arcs may lead to underestimation of the real hangar capacities. When the duration of a maintenance operation is less than 24 hours (eight hours in our case) and these operations do not necessarily occur during night time, the same hangar can be used by different aircrafts during the same day. Since the available time window between the flights connected by a replenishment arc can be larger than the duration required by the maintenance operation, corresponding maintenances of replenishment arcs with intersecting time windows can be *scheduled* without violating the hangar capacities. However, our construction does not allow usage of intersecting replenishment arcs simultaneously even though a feasible *schedule* can exist, resulting in an underestimation of real hangar capacities.

Although limiting the number of intersecting replenishment arcs may result in underestimations in terms of hangar capacity, this approach provides various advantages. First of all, such capacity constraints can be added to our ILP formulation without further complicating the existing structure. Secondly, since these constraints underestimate the real hangar capacity, it is not likely to observe tight maintenance schedules in the solution. It is always better to avoid tight maintenance schedules because they tend to create operational problems in real life. For these reasons, we divide our planning horizon to daily time windows and allow at most *capacity many* replenishment arcs to be active during that day for the corresponding airport.

ILP formulation for *capacitated* OAMRP (COAMRP-ILP) can be written with the addition of capacity constraints to OAMRP-ILP:

$$\sum_{\substack{i,j: \\ rep(i,j)\in R_{aw}}} \sum_{k\in C} y_{ijk} \leq c_{aw} \qquad\qquad a \in \mathscr{A}, w \in \mathscr{W} \qquad\qquad (6.1)$$

$\mathscr{A}$ is the set of airports and $\mathscr{W}$ is the set of time windows. Set R_{aw} is a subset of R and it contains the replenishment arcs that represent maintenance opportunities in airport a during time window w. Constraints (6.1) ensure that the number of maintenance operations in airport a for time period w is not greater than the maintenance capacity, c_{aw}.

CA-based solution procedure for capacitated case differs from that of OAMRP in terms of objective calculation only. When we ignore the airport maintenance capacities, each aircraft can select the best maintenance opportunity in its current route independently and individual unused legal flying times and infeasibilities of aircrafts can be added to calculate g and h components of the objective function. However, in capacitated case, such an approach may create infeasibilities because independently determined maintenances of aircrafts can be capacity infeasible. For this reason, the

following capacitated maintenance assignment model (CMA) must be solved to determine the best maintenance assignments for a solution s of CA:

CMA:

$$min \sum_{k \in C''} \sum_{l}^{|M_k|} r_{kl} m_{kl} \tag{6.2}$$

$s.t.$

$$\sum_{l}^{|M_k|} m_{kl} = 1 \qquad k \in C'' \tag{6.3}$$

$$\sum_{k \in C''} \sum_{l}^{|M_k|} e_{awkl} m_{kl} \leq c_{aw} \qquad a \in \mathscr{A}, w \in \mathscr{W} \tag{6.4}$$

$$m_{kl} \in \{0, 1\} \qquad k \in C'', l = 1, ..., |M_k| \tag{6.5}$$

In this formulation, C'' is the set of aircrafts that need to undergo maintenance with respect to the route assignments of solution s. The critical aircrafts are not necessarily a member of set C'' because even the critical ones do not undergo maintenance when the total accumulated flight duration of their routes are less than their remaining times; hence, $C'' \subseteq C$. $M_k = \{m_{k1}, m_{k2}, ..., m_{k|M_k|}\}$ is the set of feasible maintenance opportunities for aircraft k where variable m_{kl} represents the l^{th} opportunity. m_{kl} is a binary variable which takes value 1 if aircraft k uses its l^{th} maintenance opportunity and 0 otherwise. r_{kl} parameter is the amount of remaining time incurred by aircraft k when it selects the l^{th} maintenance opportunity in its route. e_{awkl} is a binary parameter that takes value 1 if l^{th} maintenance opportunity of aircraft k take place in airport a during time window w and 0 otherwise.

Objective (6.2) is the minimization of total remaining time. Constraints (6.3) are maintenance satisfaction constraints which assure that every aircraft that needs maintenance utilizes one of its feasible maintenances opportunities. Constraints (6.4)

are similar to constraints (6.1). They ensure that the number of maintenance operations in airport a for time period w is not greater than the maintenance capacity, c_{aw}.

In our capacitated CA iterations, CMA is not solved for every route assignment we generate. We first assign the best maintenances to aircrafts independently, calculate objective components as we do in uncapacitated case and check whether the assignments we find are capacity feasible. CMA is only solved if the maintenance assignments we find are capacity infeasible.

6.1.1. Experiments for the Capacitated Case

The flight network on which we performed capacitated experiments contains more flights than the flight network we used in Chapter 5. There are two reasons why we choose a larger flight network: First of all, it is not very likely to create tight test cases in terms of capacity when the network and corresponding fleet is small. Secondly, larger test cases can provide us information regarding the real life applicability CA. The new flight network, which is another domestic flight network of the same commercial airline, consists of 667 flights per week and a 20-aircraft fleet is assigned to cover these flights. Technically, the flight network includes 667 nodes, 11987 ordinary arcs and 6670 replenishment arcs. The longest route in terms of accumulated duration, L, is 5140 minutes. The flights in the flight network visit 36 airports and five of these airports are maintenance airports. Among these maintenance airports, one has a maintenance capacity of five whereas the others have maintenance capacities of one. Our aim is, again, to determine weekly routes for the aircrafts in the fleet with minimum unused flying time.

The test cases are generated in a similar fashion to those in Section 5.1. The number of critical aircrafts, $|C|$, takes on values 8 and 12 and the remaining time of a critical aircraft k, D_k, is randomly generated between 0 to 2000 minutes as multiplies of five. Again, for each $|C|$ value, ten different test cases are created by assigning randomly determined remaining times to arbitrarily selected $|C|$ many aircrafts in the

fleet. In fact, the number of critical aircrafts are exaggerated and D_k parameters are limited to maximum of 2000 minutes to have sufficiently tight test cases in terms of capacity. In summary, we have 20 test cases, where the test cases 1-10 and 11-20 have 8 and 12 critical aircrafts respectively.

The parameter settings of capacitated CA runs are the same as those of 8-aircraft runs except the initial cycle length. Initial cycle length is increased from 60 to 100, parallel to increase in the number of flights in the flight network. The summary of ten capacitated CA replications can be seen in Table 6.1.

For the capacitated CA replications, we report the same statistics as in Table 5.3 along with the averages of objective values of first feasible solutions found during replications, $\overline{Z}_f$. Additionally, since the exact methods are unable to generate even feasible solutions in five hours, we do not report any results related to exact runs. In order to have an opinion about the quality of solutions, we report the underutilization percentage, U, which denotes the unused portion of total remaining flying time:

$$U = \frac{Z}{\sum_{k \in C} D_k} \times 100 \tag{6.6}$$

When we consider quality of solutions, we know that Z_{best} values for cases 1, 3, 4, 6, 7, 12 and 20 are optimal because 0 is a lower bound to OAMRP (i.e. we cannot have a negative unused flying time in a feasible routing). Since we do not have any optimal results for the rest of the cases, we can use underutilization percentages to discuss about the quality of solutions. In cases 1-10, averages of U_{best} and $\overline{U}$ are 0.14% and 0.46% respectively; in cases 11-20, these values increase to 0.79% and 1.34% leading to a grand average of 0.47% and 0.90%. In the capaciated CA replications, the underutilization decreases to less than 1% on average whereas first feasible solutions found fail to utilize more than a quarter of total remaining time. From this perspective,

Table 6.1. Capacitated Compressed Annealing Runs for 20-aircraft Fleet.

Case	Capacitated CA Replication Summary						Underutilization Percentage		
	Z_{best}	$\overline{Z}$	σ_Z	$\overline{CPU}(s)$	$\overline{Z}_f$	$\overline{CPU}_f(s)$	$U_{best}(\%)$	$\overline{U}(\%)$	$\overline{U}_f(\%)$
Case 1	0	2.5	5.4	983.44	2834.5	13.08	0	0.021	23.66
Case 2	5	21.5	13.55	1392.6	2195	83.76	0.061	0.263	26.867
Case 3	0	26	26.23	1331.82	2280.5	33.54	0	0.307	26.94
Case 4	0	18.5	23.34	1348.78	1909.5	163.61	0	0.208	21.443
Case 5	50	88.5	33.67	1302.29	3145	8.21	0.508	0.9	31.978
Case 6	0	26	19.12	1460.53	1826.5	82.14	0	0.381	26.742
Case 7	0	7.5	11.61	1242.65	2737.5	8.3	0	0.08	29.294
Case 8	35	47	14.38	1554.49	2089	37.38	0.494	0.663	29.464
Case 9	20	39	16.63	1240.49	1378.5	204.55	0.274	0.535	18.897
Case 10	5	74.5	74.22	1267.99	2551.5	31.65	0.085	1.27	43.504
Avg. 1-10	11.5	35.1	23.81	1312.51	2294.75	66.62	0.142	0.463	27.879
Case 11	90	149	34.86	1540.56	3608	24.11	0.707	1.17	28.342
Case 12	0	18.5	13.75	1498.05	3921.5	14.35	0	0.131	27.842
Case 13	110	132.7	23.86	1590.95	1236.6	328.58	1.129	1.362	12.69
Case 14	30	66.5	43.14	1994.68	4137.5	8.44	0.225	0.5	31.097
Case 15	25	39.5	13.22	1806.26	3161.5	35.24	0.229	0.362	28.991
Case 16	185	293	82.7	1523.97	2760	33.06	1.961	3.105	29.253
Case 17	165	282	74.84	1914.3	2171.5	123.62	2.124	3.629	27.947
Case 18	35	53.5	17	1589.88	3533	142.94	0.244	0.373	24.629
Case 19	110	223	107.11	1763.42	2364	140.77	1.275	2.584	27.393
Case 20	0	17.5	18.14	1669.64	3602.5	88.58	0	0.134	27.669
Avg. 11-20	75	127.52	42.86	1689.17	3049.61	93.97	0.789	1.335	26.585
Grand Avg	43.25	81.31	33.34	1500.84	2672.18	80.3	0.466	0.899	27.232

we can claim that capacitated CA replications perform well on 20-aircraft fleet runs in terms of solution quality. In addition to that, among total of $20 \times 10 = 200$ replications, only one replication failed to return a feasible solution.

The average solution time per replication is around 25 minutes; therefore, best solutions are obtained around four hours. Furthermore, we can still find feasible solutions to our problem in less than two minutes. In this sense, we can claim that the responsiveness of CA is not lost because in case of a sudden state change low quality

(underutilization around 25%) weekly routes for the entire fleet can be generated in two minutes and these routes can be improved significantly (underutilization around 1%) within 25 minutes.

6.1.2. Scalability and Applicability of Compressed Annealing

One of the most important factors that affects the total solution time of CA is the number of flights to be routed, i.e. the length of planning horizon. In the scope of this study, we aim to find weekly routes for the aircrafts; therefore, the flights in our flight network are simply the flights of the following seven days. When we switch from weekly to daily routing, the flight network becomes significantly smaller; therefore, the solution times decrease significantly. On the other hand, solving this problem in a daily fashion is a myopic approach and would reduce the quality of solutions.

Fleet size, which is strongly related to number of flights to be covered, is another important determinant of total solution time. As seen in 8-aircraft fleet and 20-aircraft fleet runs, increasing fleet size increase total solution time of CA remarkably. It is obvious that the problem we solve for 20-aircraft fleet is harder due to capacity constraints; however, the significant increase in total solution time of CA can be observed even in cases where CMA is not solved very frequently. Similar to total solution time, average first feasible solution time increases as the fleet size becomes larger; however, we must note that even for a weekly routing of a 20-aircraft fleet it is on the order of one to two minutes.

In order to have a better opinion about the scalability and applicability of CA, we duplicated the flight network of 20-aircraft fleet to obtain a 40-aircraft fleet with 1334 flights to be covered per week. In our experimentations with 40-aircraft fleet, we obtain promising results about the applicability of CA. In these cases, solution time per CA replication increases to five to six hours; and first feasible solutions are found within five minutes. Although the replications last considerably long, the solution quality seems to be unaffected. In other words, in most of the 40-aircraft cases underutilizations are

still less than 1%. According to our observations, it might be a good option to limit total execution time of CA in large scale cases when fast solutions are required because major improvements in objective function are observed within the first few hours of the CA replication. In general, first feasible solutions lead to an underutilization of 25-30% whereas this value decreases to 5% within one hour and to 1% within 2-2.5 hours. An example of underutilization change during a CA replication for 40-aircraft fleet can be seen in Figure 6.1.

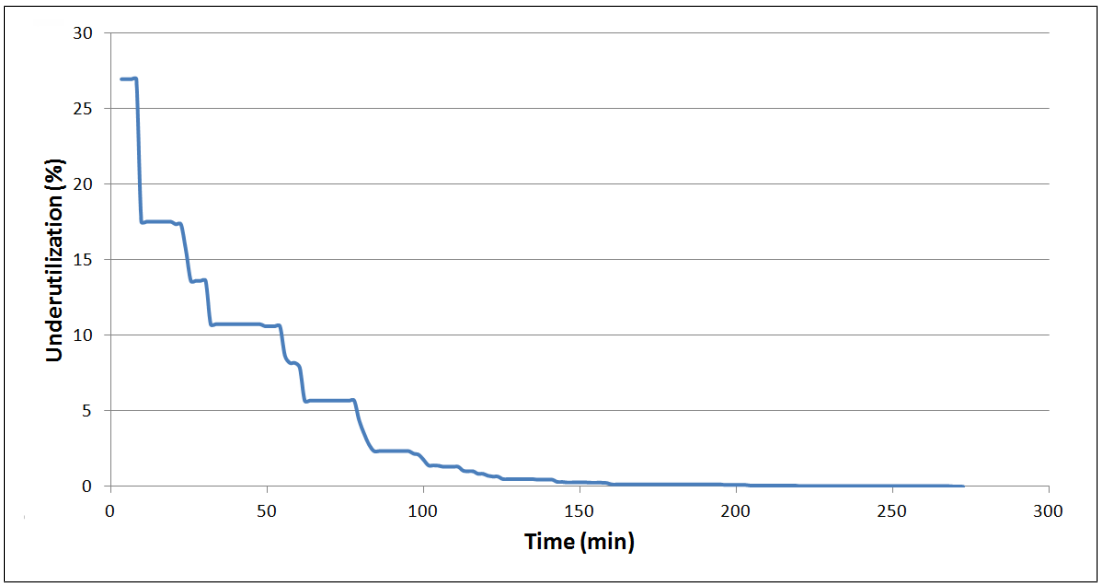

Figure 6.1. Underutilization Change During a CA Replication for 40-aircraft Fleet.

In summary, performance of CA depends on the size of fleet and the length of planning horizon. The experimental results show that CA yields high quality solutions for OAMRP even for large scaled instances within a reasonable amount of time. Furthermore, the swap move seems to be very effective in generating feasible solutions because first feasible solution times are around five minutes even for weekly route assignments of a 40-aircraft fleet.

6.2. Rolling Horizon Approach

Longer planning horizons yield high-quality routes from optimization perspective; however, such routes are not likely to be followed at operational level. Basically, long-term routing results in longer flight strings to be followed. Even if these strings are

feasible with respect to operational constraints, it is not logical to assign them to aircrafts in advance due to the stochasticity of airline environment. On the other hand, selecting shorter planning horizons results in routes that are more likely to be followed; however, it leads to a myopic perspective that may cause low-quality solutions. Therefore, a rolling horizon approach which combines quality of long-term horizon and applicability of short-term routing can remedy these problems.

In our rolling horizon approach, we basically optimize for a long planning horizon and fix a small portion of our solution. For instance, we solve OAMRP for the following seven days to determine week-long routes for each aircraft and from those routes, the flights of the following day are set as daily routes to be followed. Then, at the end of the day, we again solve OAMRP by reconstructing the flight network for the following seven days and repeat this procedure. Upon doing so, we are able to determine shorter routes, which are more likely to be followed at operational level, without ignoring the flights of the following days.

In this approach, we need to deal with the already booked maintenances. Booking a maintenance operation simply means reserving a maintenance interval at some hangar of a maintenance airport and we call these bookings as *booked maintenance slots*. Since airports must be informed about maintenance operations beforehand, we always have some booked maintenance slots that we have to consider within our current planning horizon. In other words, although we have the freedom of fixing the routes of the next day at the last moment, we are not free to plan for maintenance operations that late. As a result, we need to plan for our maintenance operations with respect to our long-term routing results and try to obey them as much as we can at operational level.

In the literature, when a maintenance operation is booked for some aircraft k, it must undergo that specific maintenance regardless of the situation. In other words, the aircraft which undergoes maintenance in a certain booked maintenance slot is determined during planning of maintenance and cannot be changed. For this purpose, such an aircraft is either forced to follow a fixed route until the booked maintenance

occurs or restricted to arrive to the airport of the booked maintenance at the planned time. In this study, different from the literature, we do not fix the specific aircraft which will use the booked maintenance slot in advance, instead, we only have a restriction which specifies that each booked maintenance slot must be utilized by an aircraft that needs to undergo maintenance. For instance, we may plan for a maintenance operation and book a maintenance slot, p, at a certain airport at a certain time by considering the long-term routing result of some aircraft k; however, p is not necessarily used by aircraft k because during our rolling horizon approach, routes of aircrafts keep changing and p may become more beneficial for another aircraft k'. We make this relaxation due to the fact that most of the fleets consist of similar types of aircrafts because maintenance slots can be used by any aircraft in such fleets.

We assume that a maintenance operation can take place within a booked maintenance slot p, if the replenishment arc of the corresponding maintenance and p share a common interval whose length is more than a user-defined specific proportion of the maintenance duration. Figure 6.2 shows an example where this proportion is set to 3/4 of 8-hour maintenance duration. In the figure, all replenishment arcs except the dotted ones can take place during p because they have sufficiently (more than six hours) long common intervals with p. Notice that in the figure, i_1, i_2, i_3 and i_4 are arriving and j_1, j_2, j_3 and j_4 are departing flight legs for a maintenance aircraft.

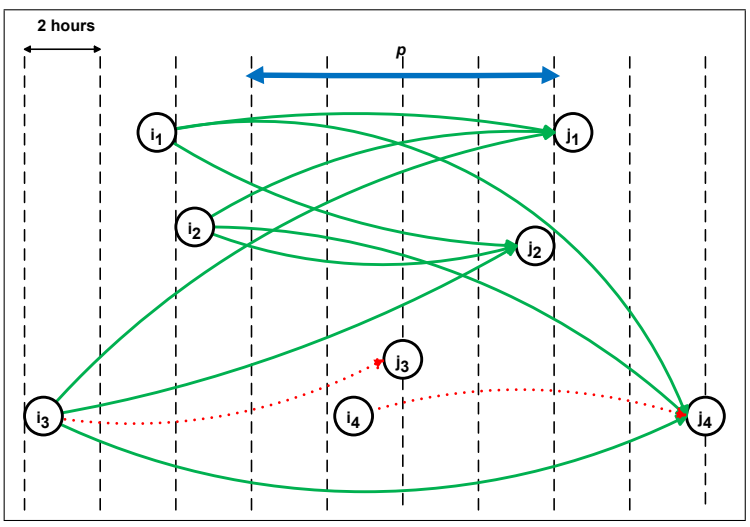

Figure 6.2. Maintenance Opportunities Which Can Take Place During p.

The maintenance slot satisfaction constraints must be added to OAMRP-ILP (or COAMRP-ILP) in order to generate feasible routes for rolling horizon approach:

$$\sum_{\substack{i,j: \\ rep(i,j)\in\overline{R}_p}} \sum_{k\in C} y_{ijk} \geq 1 \qquad\qquad p \in \mathscr{P} \qquad\qquad (6.7)$$

$\mathscr{P}$ is the set of booked maintenance slots, set $\overline{R}_p$ is a subset of R and it contains replenishment arcs that represent maintenance opportunities which can take place within the booked maintenance slot p. Therefore, constraints (6.7) ensure that there is at least one maintenance operation during booked maintenance slot p. We must note that these constraints do not work correctly when a replenishment arc $rep(i,j)$ belongs to more than one $\overline{R}_p$ sets. For instance, when there are two booked maintenance slots , say p_1 and p_2, within the same interval at the same airport (which may be the case in airports with maintenance capacities larger than one), then a replenishment arc $rep(i,j)$ can be element of both $\overline{R}_{p_1}$ and $\overline{R}_{p_2}$. In such a case, if we decide to use $rep(i,j)$ for some aircraft k, then we seem to satisfy both constraints by filling both slots with a single maintenance operation; however, this is not correct. Therefore, we need to define variables in order to distinguish replenishment arcs that belong to different $\overline{R}_p$ sets. y^p_{ijk} is a binary variable denoting the replenishment arc $rep(i,j)$ of aircraft k in set $\overline{R}_p$. Additionally, index set $A(i,j)$ consists of p indices of $\overline{R}_p$ sets that contain $rep(i,j)$ as an element. In order to correct our formulation, constraints (6.7) are replaced with the following ones:

$$\sum_{\substack{i,j:\\ rep(i,j)\in\overline{R}_p}} \sum_{k\in C} y^p_{ijk} \geq 1 \qquad p \in \mathscr{P} \tag{6.8}$$

$$\sum_{p\in A(i,j)} y^p_{ijk} = y_{ijk} \qquad i,j \in I, k \in C \tag{6.9}$$

$$y^p_{ijk} \in \{0,1\} \qquad i,j \in I, k \in C, p \in A(i,j) \tag{6.10}$$

Constraints (6.8), (6.9) and (6.10) ensure that there is at least one maintenance operation during each booked maintenance slot.

CA-based solution procedure for rolling horizon approach differs from that of OAMRP in terms of objective structure and calculation. In the objective function we have a new infeasibility component u, where u denotes the number of unsatisfied booked maintenance slots. We define another pressure parameter for u component, μ, and initialize μ_{max} similar to λ_{max} by using u instead of h. Rolling horizon based capacity maintenance assignment model (RHCMA) can be written with the addition of following constraints to CMA:

$$\sum_{\substack{k,l:\\ m_{kl}\in\overline{M}_p}} m^p_{kl} \geq 1 \qquad p \in \mathscr{P} \tag{6.11}$$

$$\sum_{p\in E(k,l)} m^p_{kl} = m_{kl} \qquad k \in C'', l = 1,...,|M_k| \tag{6.12}$$

$$m^p_{kl} \in \{0,1\} \qquad k \in C'', l = 1,...,|M_k|, p \in E(k,l) \tag{6.13}$$

Similar to $\overline{R}_p$, $\overline{M}_p$ is a subset of M_k and it contains the maintenance opportunities which can take place in slot p. Similar to y^p_{ijk} variables, m^p_{kl} variables are defined to

represent the maintenance opportunities in set $\overline{M}_p$. m_{kl}^p is a binary variable that takes value 1 if l^{th} maintenance opportunity of aircraft k is used during slot p, and 0 otherwise. Similar to $A(i,j)$, $E(k,l)$ is an index set of p indices of $\overline{M}_p$ sets that contain m_{kl} as an element.

Similar to capacitated case, during CA iterations of rolling horizon approach, RHCMA is not solved for every route assignment we generate. We first assign the best maintenances to aircrafts independently and calculate objective components as we do in uncapacitated case and check whether the assignments we find are capacity and slot feasible. RHCMA is only solved if the maintenance assignments we find are capacity or slot infeasible.

6.2.1. Maintenance Booking Preservation Example

An example of maintenance booking preservation with different aircrafts is presented in Figure 6.3, where two different route assignment segments regarding the third day of a route assignment is shown. The length of intervals between two consecutive time lines is 15 minutes and notice that night time is squeezed since there are no flights during that interval (between 22:00-03:00). Gray rectangles represent assigned flights of corresponding aircrafts. Black and checkered rectangles represent the intervals in airports a_1 and a_2 during which the corresponding aircrafts must undergo maintenance for eight hours. For instance, in the first route assignment aircraft k_1 arrives to a_1 at 18:05 and we know that it must undergo maintenance until its departure at 05:00.

In the example the first route assignment (upper one) is done at the beginning of the planning horizon and by looking at the results we decide to book two slots for aircraft k_1 and k_2 in airports a_1 and a_2 respectively. The first booked slot, p_1, is for aircraft k_1 in airport a_1 between 20:00 and 04:00 and the second one, p_2 is for aircraft k_2 in airport a_2 between 21:00 and 05:00. At the end of the first day, due to delays caused by some stochastic events, the remaining time of k_1 turns out to be less than our expectation, causing its existing route assignment to become infeasible. Therefore, at

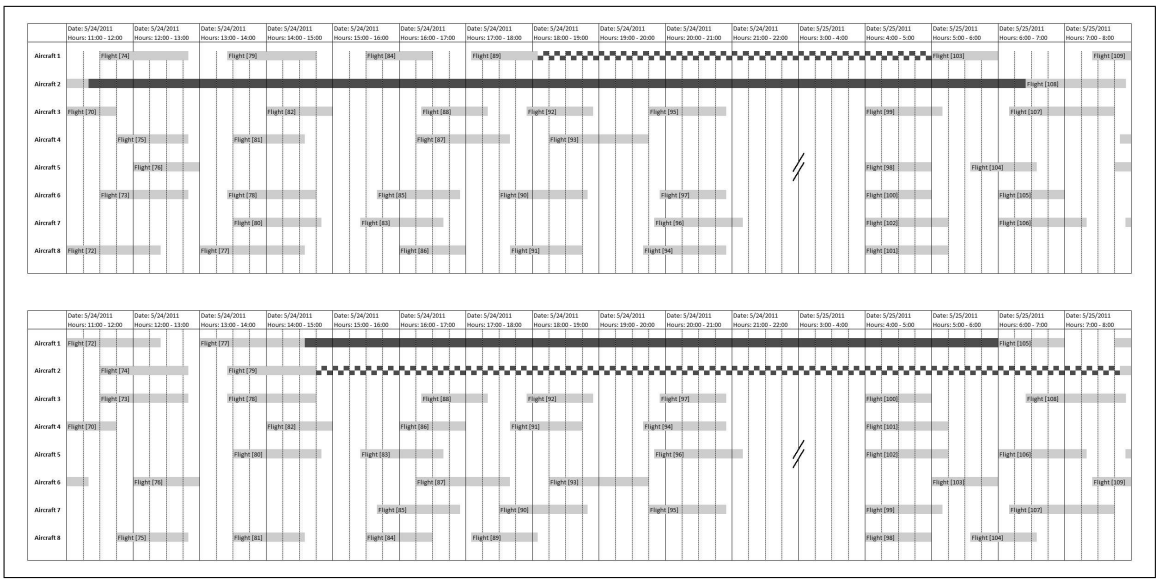

Figure 6.3. Rolling Horizon Example.

the end of the first day, the horizon is shifted one day and new weekly route assignments (below one) are determined with respect to current states of the aircrafts. As seen in the second assignment the routes are changed and the maintenance slots p_1 and p_2 are utilized by different aircrafts. At the beginning of the horizon, p_1 was booked for aircraft k_1 and p_2 was booked for aircraft k_2; however, at the end of the first day, k_1 becomes the one which utilizes p_2 and k_2 becomes the one which utilizes p_1 assuming that no changes occur until maintenance operations.

7. CONCLUSION

In the scope of this study, we proposed a new ILP formulation for OAMRP with time consideration by modifying connection network and described exact and heuristic methods to solve it. From operational perspective, we took airport capacity restrictions and already booked maintenance slots into consideration. Our rolling horizon approach is unique in the sense that it does not enforce fixed routes or dedicate certain arrivals for preserving already booked maintenance operations.

On exact side of solution methodologies, we used B&B with different prioritization schemes on variable sets and compared them with the default branching scheme of CPLEX. Some of the prioritization schemes seem to perform better than the default one in terms of the number of optimal solution found and the required CPU time; however, they were not as robust as the default scheme under different test cases. As a heuristic method, we utilized CA, which is an extension of SA method where infeasibilities are penalized with a variable penalty multiplier. CA is a relatively new heuristic and never applied aircraft maintenance routing problems before. In medium scaled cases, CA was very effective in finding high quality solutions in significantly shorter amount of times compared to all exact methods tested. Additionally, in large scaled cases where exact methods fail to find even feasible solutions, CA yields a feasible solution within the first few minutes and decrease the underutilization of remaining flying time under 1% at the end of the first two hours.

In addition to solution quality, CA provides the responsiveness required by airline environment. In case of emergent state changes, CA based solution technique is able to provide new feasible solutions on the order of minutes. This is an important property of CA because such emergent state changes are very likely to occur in airline industry.

Contrary to most of the studies which focus OAMRP on daily basis, we try to solve the problem for one-week planning horizon because planning entire week on a

daily basis may lead to low quality results due to its myopic perspective. We also proposed a rolling horizon based method that can be utilized to determine daily routes by taking longer horizons into account.

From modeling perspective, integrating OAMRP with crew scheduling in an efficient way can be a challenging future research subject. In terms of solution methodology, more sophisticated branching schemes may be designed to increase the efficiency of B&B for large scaled cases. Additionally, finding high quality lower bounds for OAMRP may improve the performance of exact methods significantly because lower bounds either improve very slowly or stuck at very poor values during optimization. On the other hand, more intelligent neighbor generation rules may be looked for to further improve the solution quality of CA.

REFERENCES

Afsar, H. M., M. -L. Espinouse, and B. Penz, 2006, "A two-step heuristic to build flight and maintenance planning in a rolling horizon", In: Proceedings of Service Systems and Service Management, *2006 IEEE International Conference on Service Systems and Service Management*, Troyes, France, 25-27.10.2006, IEEE.

Afsar, H. M., M. -L. Espinouse, and B. Penz, 2009, "Building flight planning for an airline company under maintenance constraints", *Journal of Quality in Maintenance Engineering*, Vol. 15(4), pp. 430-443.

Argüello, M. F., and J. F. Bard, 1997, "A GRASP for aircraft routing in response to groundings and delays", *Journal of Combinatorial Optimization*, Vol. 5, pp. 211-228.

Barnhart, C., N. L. Boland, L. W. Clarke, E. L. Johnson, G. L. Nemhauser, and R. G. Shenoi, 1998, "Flight string models for aircraft fleeting and routing", *Transportation Science*, Vol. 32(3), pp. 208-220.

Cerny, V., 1985, "Thermodynamical Approach to the Traveling Salesman Problem: An efficient simulation algorithm", *Journal of Optimization Theory and Applications*, Vol. 45, pp. 41-51.

Clarke, L. W., C. A. Hane, E. L. Johnson, and G. L. Nemhauser, 1996, "Maintenance and crew considerations in fleet assignment", *Transportation Science*, Vol. 30(3), pp. 249-260.

Clarke, L. W., E. L. Johnson, G. L. Nemhauser, and Z. Zhu, 1997. "The aircraft rotation problem", *Annals of Operations Research*, Vol. 69, pp. 33-46.

Cohn, A. M., and C. Barnhart, 2003, "Improving crew scheduling by incorporating key maintenance routing decisions", *Operations Research*, Vol. 51(3), pp. 387-396.

Cordeau, J. F., G. Stojković, F. Soumis, and J. Desrosiers, 2001, "Benders decomposition for simultaneous aircraft routing and crew scheduling", *Transportation Science*, Vol. 35(4), pp. 375-388.

Daskin, M. S., and N. D. Panayotopoulos, 1989, "Assigning aircraft to routes in hub and spoke networks.", *Transportation Science*, Vol. 23(2), pp. 91-99.

Desaulniers, G., J. Desrosiers, Y. Dumas, M. M. Solomon, and F. Soumis, 1997, "Daily aircraft routing and scheduling", *Management Science*, Vol. 43, pp. 841-855.

Dowsland, A. K., 1993, "Simulated annealing", In: C. R. Reeves (Ed.), *Modern Heuristic Techniques for Combinatorial Problems*, Oxford, Blackwell.

Etschmaier, M. M., and D. F. X. Mathaisel, 1985, "Airline scheduling: An overview", *Transportation Science*, Vol. 19(2), pp. 127-138.

Feo, T. A., and J. F. Bard, 1989, "Flight scheduling and maintenance base planning", *Management Science*, Vol. 35, pp. 1415-1432.

Gabteni, S., and M. Grönkvist, 2008, "Combining column generation and constraint programming to solve the tail assignment problem", *Annals of Operations Research*, Vol. 171, pp. 61-76.

Gopalakrishnan, B., and E. L. Johnson, 2005, "Airline crew scheduling: State-of-the-art", *Annals of Operations Research*, Vol. 140, pp. 305-337.

Gopalan, R., and K. T. Talluri, 1998, "The aircraft maintenance routing problem", *Operations Research*, Vol. 46(2), pp. 260-271.

Grönkvist, M., 2005, *The tail assignment problem*, doctoral dissertation, Chalmers University of Technology and Göteborg University.

Hane, C. A., C. Barnhart, E. L. Johnson, R. E. Marsten, G. L. Nemhauser, and G. Sigismondi, 1995, "The fleet assignment problem: Solving a large-scale integer program", *Mathematical Programming*, Vol. 70, pp. 211-232.

Haouari, M., N. Aissaoui, and F. Z. Mansour, 2009, "Network flow-based approaches for integrated aircraft fleeting and routing", *European Journal Of Operational Research*, Vol. 193, pp. 591-599.

IBM ILOG CPLEX Optimizer, 2012, *IBM - Mathematical Programming: Linear Programming, Mixed-Integer Programming and Quadratic Programming - IBM*

ILOG CPLEX Optimizer - Software, http://www-01.ibm.com/software/integration/optimization/cplex-optimizer/, accessed 08.05.2012.

IBM ILOG CPLEX v12.1 - User's Manual for CPLEX, 2009, *IBM ILOG CPLEX v12.1 - User's Manual for CPLEX*, ftp://ftp.software.ibm.com/software/websphere/ilog/docs/optimization/cplex/ps_usrmancplex.pdf, accessed 08.05.2012.

Kabbani, N. M., and B. W. Patty, 1992, "Aircraft routing at American airlines", In: Proceedings of the 32nd Annual Symposium of AGIFORS, *32nd Annual Symposium of AGIFORS*, Budapest, Hungary, 1992.

Kirkpatrick, S., C. D. Gelatt, and M. P. Vecchi, 1983, "Optimization by simulated annealing", *Science*, Vol. 220, pp. 671-680.

Klabjan, D., E. L. Johnson, G. L. Nemhauser, E. Gelman, and S. Ramaswamy, 2002, "Airline crew scheduling with time windows and plane-count constraints", *Transportation Science*, Vol. 36(3), pp. 337-348.

Liang, Z., and W. A. Chaovalitwongse, 2009, "The aircraft maintenance routing problem", In: W. A. Chaovalitwongse, K. C. Furman, & P. M. Pardalo (Eds.), *Optimization and Logistics Challenges in the Enterprise*, pp. 327-348, Springer, New York.

Liang, Z., W. A. Chaovalitwongse, H. C. Huang, and E. L. Johnson, 2011, "On a new rotation tour network model for aircraft maintenance routing problem", *Transportation Science*, Vol. 45(1), pp. 109-120.

Mak, V., and N. Boland, 2000, "Heuristic approaches to the asymmetric travelling salesman problem with replenishment arcs", *International Transactions in Operational Research*, Vol. 7, pp. 431-447.

Mercier, A., J. F. Cordeau, and F. Soumis, 2005, "A computational study of Benders decomposition for the integrated aircraft routing and crew scheduling problem", *Computers & Operations Research*, Vol. 32, pp. 1451-1476.

Mercier, A., and F. Soumis, 2007, "An integrated aircraft routing, crew scheduling and flight retiming model", *Computers & Operations Research*, Vol. 34, pp. 2251–2265.

Morse, C., 1997, *Stochastic Equipment Replacement with Budget Constraints*, doctoral dissertation, University of Michigan.

Ohlmann, J. W., J. C. Bean, and S. G. Henderson, 2004, "Convergence in Probability of Compressed Annealing", *Mathematics of Operations Research*, Vol. 29, pp. 837-860.

Ohlmann, J. W., and B. W. Thomas, 2007, "A compressed-annealing heuristic for the traveling salesman problem with time windows", *INFORMS Journal on Computing*, Vol. 19, pp. 80-90.

Orhan, I., M. Kapanoglu, and T. H. Karakoc, 2011, "Concurrent aircraft routing and maintenance scheduling", *Journal of Aeronautics and Space Technologies*, Vol. 5(1), pp. 73-79.

Sarac, A., R. Batta, and C. M. Rump, 2006, "A branch-and-price approach for operational aircraft maintenance routing", *European Journal of Operational Research*, Vol. 175, pp. 1850-1869.

Sherali, H. D., E. K. Bish, and X. Zhu, 2006, "Airline fleet assignment concepts, models, and algorithms", *European Journal of Operational Research*, Vol. 172, pp. 1-30.

Sosnowska, D., and J. Rolim, 2001, "Fleet Scheduling Optimization: A Simulated Annealing Approach", In: E. K. Burke, & W. Erben (Eds.), *Practice and Theory of Automated Timetabling III*, pp. 227-241, Springer, Berlin.

Sriram, C., and A. Haghani, 2003, "An optimization model for aircraft maintenance scheduling and re-assignment", *Transportation Research Part A: Policy and Practice*, Vol. 37(1), pp. 29-48.

Talluri, K. T., 1998, "The four-day aircraft maintenance routing problem", *Transportation Science*, Vol. 32(1), pp. 43-53.

Theodoracatos, V. E., and J. L. Grimsley, 1995, "The optimal packing of arbitrarily-shaped polygons using simulated annealing and polynomial-time cooling schedules", *Computer Methods in Applied Mechanics and Engineering*, Vol. 125, pp. 53-70.

Weide, O., D. Ryan, and M. Ehrgott, 2010, "An iterative approach to robust and integrated aircraft routing and crew scheduling", *Computers & Operations Research*, Vol. 37, pp. 833-844.

APPENDIX A: BASIC FLIGHT NETWORK ELEMENTS

Flight Element: Flight element keeps track of arrival and departure information of a flight leg along with lists of ordinary & replenishment arcs that either enter into or emanate from that flight. We use *index* field to label flight legs with distinct integer numbers. The object diagram of flight element can be seen in Figure A.1.

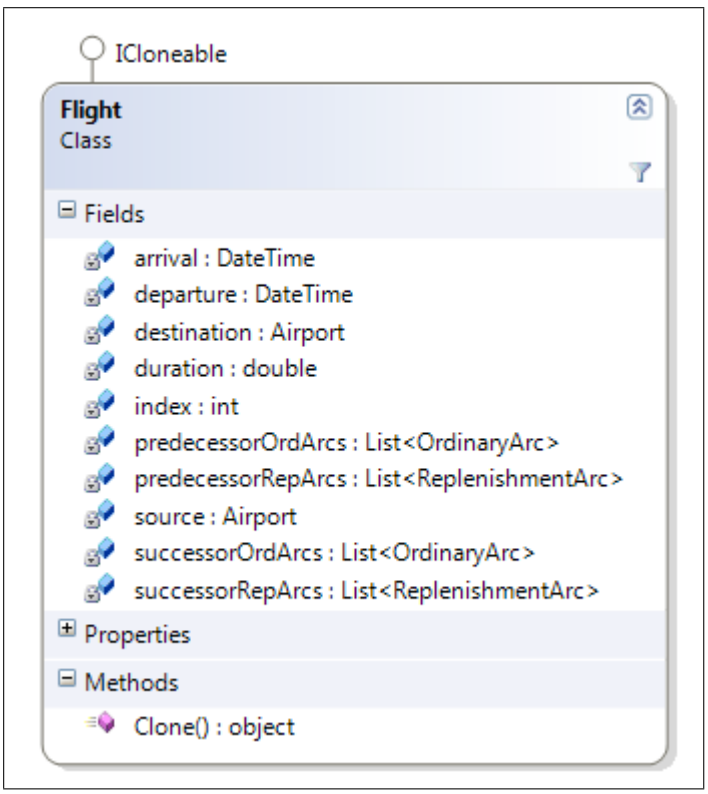

Figure A.1. Class Diagram of Flight Class.

Ordinary Arc Element: Ordinary arc element represents the ordinary connection (connection without a maintenance operation) between flight legs i and j. INumVar arrays x and z have lengths of $|K|$ and they represent x_{ijk} and z_{ijk} variables respectively. Each ordinary arc keeps the indices (as headIndex and tailIndex) of flight legs that it connects. In addition, *groundtime* is the amount of time spent in the airport between connected flight legs and *throughvalue* is the benefit obtained by utilizing the arc. In the scope of our study, through values of all arcs are assumed to be equal; hence, this field is never used. The object diagram of ordinary arc element can be seen in Figure

A.2.

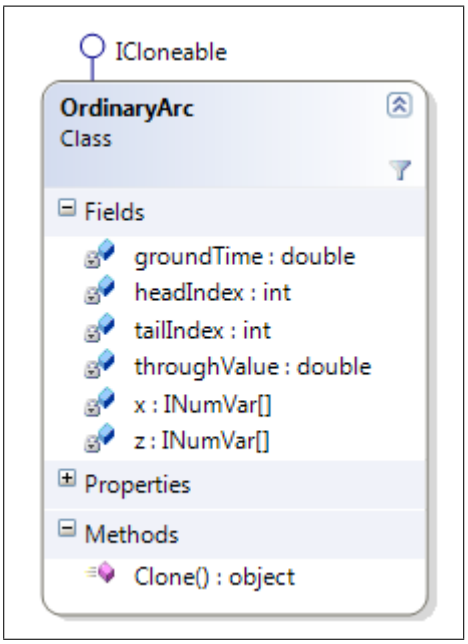

Figure A.2. Class Diagram of Ordinary Arc Class.

Replenishment Arc Element: The structure of replenishment arc element is the same as that of ordinary arc. The only difference between them stems from INumVar arrays. INumVar array y has a length of $|C|$ and it represents y_{ijk} variables. The object diagram of replenishment arc element can be seen in Figure A.3.

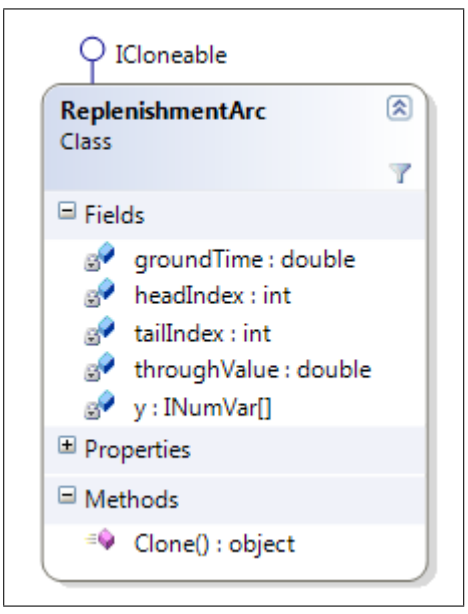

Figure A.3. Class Diagram of Replenishment Arc Class.

Airport Element: As seen in Figure A.4, airport element consists of two fields representing its name and maintenance capacity. An airport is a maintenance airport when *replenishmentCapacity* field is greater than 0.

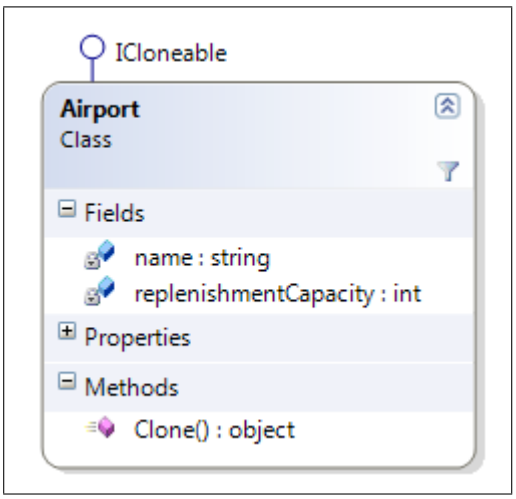

Figure A.4. Class Diagram of Airport Class.

Flight Network Element: Flight Network element is the constructor class of the flight network. The inputs are the lists of flight elements and airports. In addition, we provide the minimum required (35 minutes for ordinary arc & 480 minutes for replenishment arc) and maximum allowed ground time (1435 minutes for ordinary arc & 1440 minutes for replenishment arc) durations to feasibly connect two flight legs either with ordinary arcs or with replenishment arcs. The maximum allowed ground time parameter is used not to generate unnecessarily long arcs in flight network.

At the beginning, individual flight elements have empty arc lists. Within the constructor of flight network element, *ConstructFeasibleOrdSuccessorsAndPredecessors()* and *ConstructFeasibleRepSuccessorsAndPredecessors()* functions are called in order to construct the flight network by filling the arc lists of flight elements. *ordArcElements* and *repArcElements* are the lists which consist of all ordinary and all replenishment arc elements of flight network respectively. The object diagram of flight network element can be seen in Figure A.5.

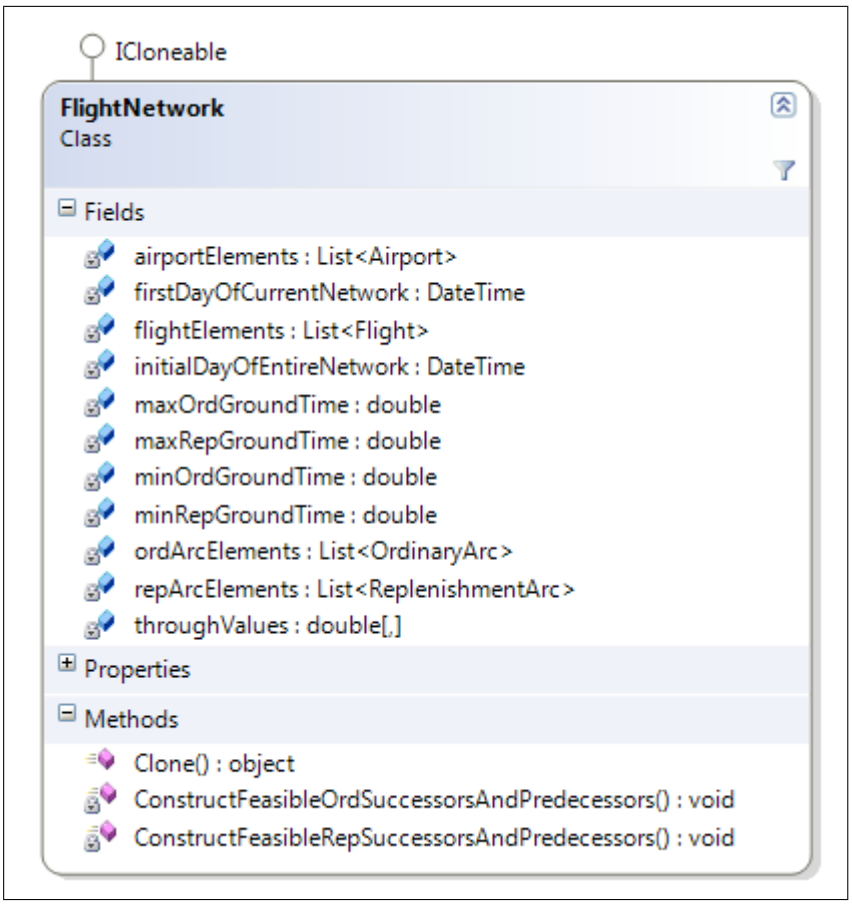

Figure A.5. Class Diagram of Flight Network Class.

APPENDIX B: FLIGHT DATA & TEST CASES

The flight data and test cases used for 8-aircraft and 20-aircraft fleet experiments can be found at http://www.bufaim.boun.edu.tr/OAMRP-Data.zip. The zipped file, *OAMRP-Data.zip*, contains two separate folders for 8-aircraft and 20-aircraft fleet experiments. Under each folder, corresponding flight data and airports are provided in TXT format named as *Airports.txt* and *Flights.txt* respectively. Test cases can be found under the same folders in XLSX format named as *Test_Cases.xlsx*.