3-10-2010

Optimizing an F-16 Squadron Weekly Pilot Schedule for the Turkish Air Force

Murat Yavuz

OPTIMIZING AN F-16 SQUADRON WEEKLY PILOT SCHEDULE FOR THE TURKISH AIR FORCE

THESIS

Murat Yavuz, First Lieutenant, TUAF

AFIT-OR-MS-ENS-10-11

DEPARTMENT OF THE AIR FORCE
AIR UNIVERSITY

AIR FORCE INSTITUTE OF TECHNOLOGY

Wright-Patterson Air Force Base, Ohio

The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Turkish Air Force, Department of Defense, or the Turkish Government.

AFIT-OR-MS-ENS-10-11

OPTIMIZING AN F-16 SQUADRON WEEKLY PILOT SCHEDULE FOR THE TURKISH AIR FORCE

THESIS

Presented to the Faculty

Department of Operational Sciences

Graduate School of Engineering and Management

Air Force Institute of Technology

Air University

Air Education and Training Command

In Partial Fulfillment of the Requirements for the

Degree of Master of Science in Logistics Management

Murat Yavuz, B.S.E.E

First Lieutenant, TUAF

March 2010

AFIT-OR-MS-ENS-10-11

OPTIMIZING AN F-16 SQUADRON WEEKLY PILOT SCHEDULE FOR THE TURKISH AIR FORCE

Murat Yavuz, B.S.E.E
First Lieutenant, TUAF

Approved:

//SIGNED//	19 March 2010
Dr. James T. Moore (Chairman)	date
//SIGNED//	19 Match 2010
Dr. Jeffery D. Weir (Member)	date

Abstract

Fighter squadrons in the Turkish Air Force build flight schedules for weekly periods. This process requires a great deal of time and does not seek optimality. Schedules are built with feasibility concerns. The Turkish Air Force doesn't have an automated tool for flight scheduling. Many constraints including crew rest, number of sorties flown in a month, and duty currencies affect the schedule. Providing an automated scheduling tool may help schedulers save time for other squadron tasks including mission preparation, briefing, and debriefing. In this research, a heuristic approach to the problem is developed. Greedy Randomized Adaptive Search Procedures (GRASP) is applied to the weekly pilot scheduling problem. Manual scheduler inputs are allowed. A code for GRASP implementation is written in MATLAB.

Two different approaches are used in the analysis. First, the code is run for four weekly schedules taken from an F-16 squadron of the Turkish Air Force and second, a weekly flight schedule is created randomly. In the second approach, the created flight schedule is used for three different scenarios which represent possible real life situations.

For all scenarios and real schedules, GRASP performed well and smaller standard deviations in sortie numbers are obtained while keeping all pilots within the currency limit of each mission.

I would like to dedicate this thesis to my fellow friend Fatih Korçam who died in an F-16 crash on April 14, 2009 while I was at AFIT for my master degree program. His memories will live with me forever.

Acknowledgments

First and foremost, I would like to thank to the Republic of Turkey and the great Turkish Nation for providing me the opportunity to attend this program. I hope I can pay them back everything they provided me by serving during rest of my life.

I would like to express my sincere appreciation to my faculty advisor, Dr. James T. Moore, for his guidance and support throughout the course of this thesis effort. I certainly appreciate his insight and his experience. I would, also, like to thank my reader Dr. Jeffery D. Weir for his time and his inputs during the research.

I would, also, like to express my gratitude to the faculty members for teaching me all the precious classes.

Finally, I would like to thank my family for their patience and their support. Their existence gave me strength for overcoming the difficulties I have faced during this research.

Murat Yavuz

TABLE OF CONTENTS

Page

LIST OF FIGURES

LIST OF TABLES

OPTIMIZING AN F-16 SQUADRON WEEKLY PILOT SCHEDULE FOR THE TURKISH AIR FORCE

1. INTRODUCTION

1.1 Problem Statement

The fighter squadrons of the Turkish Air Force prepare flight schedules for weekly periods. Missions can be executed by 4 ships, by 2 ships or sometimes by a single ship. Each pilot aircraft matching is called a pilot-sortie. The total number of pilot-sorties for any given day generally varies between 0 and 30, depending on the operational sortie requirements, the number of available aircraft, the number of available pilots, weather conditions, etc. This yields up to 150 pilot-sorties for a week. Although 150 is not a large number for regular assignment problems, it requires more effort to solve the flight scheduling problem because of its non-linearity and dynamic environment. In fighter squadrons, schedulers are generally pilots, and they prepare the schedule in addition to their other squadron duties such as flight missions, Quick Reaction Alert (QRA) duties, Supervisor of Flight (SOF) duties etc. O.B.Gokcen says that it takes about 10 hours for a flight scheduler to prepare a weekly flight schedule (Gokcen, 2008). This means the scheduler is not able to perform other duties while preparing the schedule.

Currently, in almost all the Turkish Air Force fighter squadrons, on the last day of the flight week, the scheduler starts with a large paper or a spreadsheet designed for the weekly schedule. First, the scheduler coordinates with maintenance for the next week's available aircraft status. The available number of aircraft can change from day to day. For every period of the flight day, at least one aircraft tends to be reserved for use in case of ground aborts which occur when a pilot experiences a problem before the takeoff phase of a mission and must not continue the mission with that problem.

Next, coordination must be made with other squadron(s) on the same base for the use of the runway, because all takeoffs and landings must have at least 5 minute intervals between them. Also, training areas must be coordinated between squadrons.

Before scheduling training missions, operational missions coming from headquarters are directly mounted on the schedule. The operational missions have their own predefined take off time, landing time, number of aircraft and operation area.

After planning all flights, the scheduler assigns pilots to the flights. During this assignment phase, crew rest time, briefing and debriefing time and other tasks like Quick Reaction Alert (QRA) duties are considered. In addition to these constraints, pilot availabilities and qualifications, monthly flight hours and remaining available flight days in the associated month for each pilot must be taken into consideration. Since there are so many factors affecting or restricting flight scheduling, the problem becomes complex. This process requires a great deal of time and does not look for optimal solutions. Schedules are built with feasibility as the main objective.

The Turkish Air Force uses a national network and database system called HVBS (Hava Kuvvetleri Bilgi Sistemi (Air Force Database Network)) and a set of planning

modules called MY (Muharebe Yonetimi (Combat Management)) which interact with each other. But these two tools do not provide daily or weekly flight schedules. They can only be used to check the feasibility of the manually built flight schedules. Providing a scheduling tool may help a scheduler save time for other jobs in the squadron such as mission preparation, briefing, and debriefing. Also, the goals of maintaining pilots' currencies, getting rid of the unintentionally biased flight schedules, and saving scheduler's time can be fulfilled by optimizing the effectiveness of the weekly flight schedule.

1.2 Specifications of a Flight Schedule

After graduation from Combat Readiness Training, all pilots receive assignments to their first operational squadrons. In these squadrons, new pilots have to fly a certain number of sorties in the F-16D two-seat aircraft with an instructor pilot (IP) before their first solo flight with the F-16C single-seat aircraft. Then they achieve Basic Mission Capable (BMC) status. After achieving BMC status, the pilots begin to fly training flights for different missions. But, before flying a solo flight for any given mission, the mission must be performed in the F-16D with an IP a required number of times. While assigning the pilots to the aircrafts, the schedulers must check if the pilot is allowed to fly that sortie or not.

In order to gain Combat Mission Ready (CMR) status, the pilot must have a specified number of IFR (Instrument Flight Rules, the flight conditions in which the horizon cannot be defined clearly) flight hours and succeed in the Instrument Qualification (IQ) and the Tactical Qualification (TQ) check flights. A CMR pilot is allowed to perform solo flights for all primary missions of the squadron.

There are different pilot classifications. First, there is an IFR category classification that is represented by CAT1 (Category 1), CAT2 (Category 2) or CAT3 (Category 3). Every category has its own ceiling and visibility limit and determines whether a pilot can fly in a given weather condition or not. Every pilot begins without a category certification and goes through CAT3, CAT2 and eventually CAT1. In order to advance to a higher category, the pilot must have a specific number of IFR flight hours and succeed in an Instrument Category Qualification (ICQ) check flight. In the pilot assignment process, the scheduler must take weather forecasts into consideration and assign the pilots that are qualified to fly under expected weather conditions to the following week's missions.

The second classification is the flight status classification. Each pilot has a flight status such as Wingman (W), 2-Ship Flight Lead (2FL), 4-Ship Flight Lead (4FL) and Instructor Pilot (IP). Some of the IPs can be designated as Check Pilots (CP). Every pilot begins his/her flight career as a wingman and goes through 2FL, 4FL and finally IP. In order to advance to a higher flight status, the pilot must have a certain number of flight hours and succeed in the specific check flights. The scheduler must take the flight status of the pilots into consideration when assigning the pilots to missions. A pilot can fly the missions which require a flight status no higher than his/her flight status. But, generally, it is not preferred to assign a pilot to a spot with lower status.

Beside the classifications, pilots need to satisfy some requirements. The first of these requirements is mission currency. Each mission has its own currency limit in terms of the number of days. The currency limit implies that the pilot must perform a mission within a number of days since the date he/she flew the same mission, in order to be able

to fly solo. Otherwise, the pilot must perform the mission in an F-16D with an IP. After performing the mission, the pilot may fly solo before the end of the currency limit unless the mission is a night flight. For a night flight, if a pilot does not perform at least one night flight every 45 days, a night flight in an F-16D with an IP, followed by a night solo flight within 7 days, must be performed. Otherwise, the pilot exceeds the currency limit. The scheduler must take the pilot's currencies into consideration in order to prevent assigning a pilot to a mission improperly.

The second requirement is the number of sorties flown in the associated month. Each pilot is supposed to fly at least a specific number of sorties in a month. The extra number of sorties cannot be transferred to the next month. So, at the beginning of each month, pilots begin with a zero sortie count. The scheduler must take the number of sorties flown by each pilot in the associated month into consideration in order to prevent unfairly different flight hours for each pilot.

The third requirement is check flights. Each pilot has to fly an IQ mission within the 30 days before his/her date of birth, and a TQ mission not earlier than 5 months and no later than 6 months after his/her date of birth. Otherwise, the pilot cannot perform any solo flights. So, the scheduler must take these check flights into consideration in order to maintain the pilot's availability at all times.

Other than these requirements and classifications, the Squadron Commander (SC) and the Director of Operations (DO) may want to make inputs to the schedule. These inputs have to be evaluated and integrated in the schedule by the scheduler.

1.3 Research Question

The research question is:

How can a feasible and more effective weekly pilot schedule be built in a short amount of time?

The main objective of this research is to provide the schedulers with an automated tool that saves them a considerable amount of time in the flight scheduling process while creating feasible pilot schedules and increasing the effectiveness of the schedule in terms of pilots' currencies and monthly flight hour requirements.

1.4 Scope and Limitations

This research is interested in the pilot assignment portion of squadron level flight scheduling. This assignment portion consists of not only the pilot assignment to an aircraft but also pilot assignment to the simulator, the Runway Supervisor Unit (RSU) and Supervisor of Flight (SOF) duties.

Many different inputs such as refresher flights for the pilots who did not fly more than 20 hours during the last year, extra sorties for the pilots who have not been in the squadron for two weeks or upgrade sorties for specific missions can affect the schedule. In this research, all of these rare events are considered manual input for the scheduler. Before letting the tool operate and assign the pilots to aircraft, the scheduler accomplishes the manual pilot assignments.

IQ and TQ sorties, F-16D flights for the pilots out of their currency limit, SC or DO inputs, new pilot's basic training and the solo night flights which must be flown within 7 days after the F-16D night flights are left to the scheduler as manual inputs.

These events occur rarely. The main objective of the research is to build a schedule that prevents pilots from losing their currencies.

1.5 Assumptions

Several assumptions are made:

- At the beginning of the flight year, all pilots satisfy their currency requirements for every mission.
- A minimum of two hours before takeoff time and a minimum of one hour after landing time are required for briefing and debriefing, correspondingly.
- A pilot can fly at most two sorties a day and must have 12 hours of crew rest before the next day's flight briefing.
- Night flights, Instrument Qualification (IQ) and Tactical Qualification (TQ) missions are assumed to be critical missions since losing currency on these flights affects other missions' currencies.
- A pilot can fly only one night sortie a day.
- A pilot cannot be SOF an entire day.
- A maximum of 12 flight wings can be scheduled in a given day.

1.6 Summary

In this chapter, the research question is stated and the scope of the research is defined. Also, the assumptions of the research are revealed. In Chapter 2, the background of the flight scheduling problem, previous research in scheduling and heuristic implementations are discussed. The methodology of the research is presented in Chapter 3 and this methodology is analyzed based on real life flight schedules and a generated

flight schedule in Chapter 4. In Chapter 5, conclusions and future recommendations are presented.

2. BACKGROUND:

2.1 General

In this chapter, previous research about flight scheduling and about heuristic implementations on scheduling problems is presented in order to provide background for the weekly pilot scheduling problem.

2.2 Scheduling

Scheduling is a decision-making process that can be encountered often in our daily lives. Basically, deciding to get your hair cut before or after the weekly shopping or deciding when to wash your car are examples of scheduling. More generally, “Scheduling is a decision-making process that deals with the allocation of resources to tasks over given time periods and that has the goal of optimizing one or more objectives” (Pinedo, 2008).

Although the scheduling problem is a common problem in human life and many studies have been done, the theoretical models usually do not completely fit real life situations. There have been many papers about the differences between theoretical scheduling models and real world problems (Pinedo, 2008). Every real world problem has its own characteristics and properties. Pinedo explains some of the common differences very clearly. The ones that are related to this research can be summarized as:

i. In theory, it is assumed *n* jobs have to be scheduled. The problem is assumed to be solved after scheduling these *n* jobs. But, in the real world new jobs are continuously added. The current *n* jobs have to be scheduled without perfect information about the future.

ii. In the real world, a schedule is built according to certain assumptions. Then some changes in the schedule are required based on future random events which may be encountered. Theoretical models do not take these random events into consideration and do not focus on the re-sequencing problem.

iii. Machine environments in the theoretical models are often less complicated than they are in the real world. Real world situations have more restrictions on processing and more constraints as well.

iv. In real world problems, the priorities of the jobs may vary from time to time. However, in mathematical models they are assumed to be fixed.

v. Preferences are not often taken into account in mathematical models. In a mathematical model, there are only two options for a job. It either can or cannot be processed on a given machine. In the real world, although a job can be processed on a given machine, the scheduler may prefer to schedule the job on a different machine.

vi. Despite the fact that in the real world, machines are not continuously available, most theoretical models assume that there is no restriction on machine availability. Hence, the models do not have machine availability constraints.

vii. Most theoretical research has focused on models with a single objective. Real world problems usually have a number of objectives and the importance of the objectives may change over time. Most theoretical models focus on a single objective. (Pinedo 2008)

In spite of these differences between scheduling theory and real life, the theoretical models provide valuable insight into real life scheduling problems (Pinedo 2008). Real life problems can be solved using proper scheduling approaches and their theoretical models.

2.2.1 Scheduling in Aviation

In the flight scheduling area, there has been research on both military and commercial flight scheduling problems. It is easy to find different approaches to these problems.

Aslan's research focused on the scheduling of training events in the Turkish Air Force Oncel F-16 training squadron. His research provided a tool for the development of a Daily Flight Schedule (DFS) (Aslan 2003). It uses decision rules to create an initial schedule and a bottleneck heuristic to create a suggested DFS. It provides a user interface that allows the scheduler to interact with the decision support tool. The author claims this tool provides the capability to produce both weekly schedules and the entire flight training program (Aslan 2003). In Aslan's research, all flights are predefined in a pilot training program. So, the mission sequence that must be followed by a pilot in training is

known. In Aslan's research, according to the predefined rules the missions are ordered and the suggested DFS is created by following the steps in Figure1.

1. Persons are assigned to duties
2. Scheduler selects the MOL
3. The missions are ordered according to the selected priority rule
4. If needed, "must-fly" student missions are prioritized in the MOL
5. Ties are broken according to the LFJ-LFM first rule
6. DFS is generated and if needed missions can be altered or a new mission can be inserted
7. Other resources for the mission are recorded on the DFS
8. Simulator schedule is produced (if required)
9. Course work is scheduled.

Figure 1. Feasible Initial Solution Construction Heuristic (Aslan, 2003)

In his research, Aslan provides a flow chart that shows how the software goes through these steps in his heuristic. Figure 2 shows the flow chart provided by Aslan. In this chart, MOL stands for the Mission Order List and LFJ-LFM stands for Least Flexible Job-Least Flexible Machine. Even though the heuristic is created for daily flight scheduling, the author claims that a weekly schedule can be created using the heuristic (Aslan,2003).

.

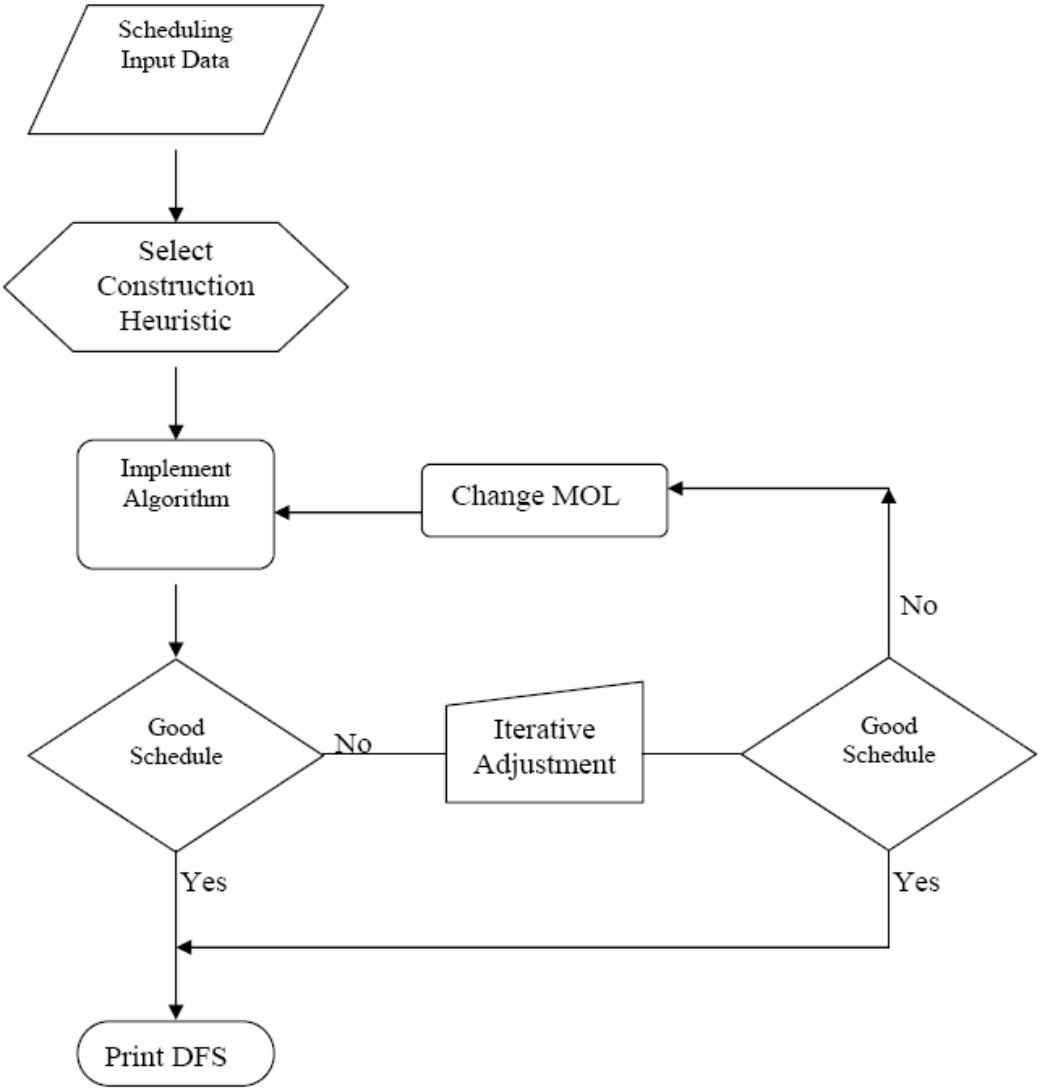


Figure 2. Software implementation of Construction Heuristic (Aslan 2003)

Similar to Aslan's research, Nguyen's research focused on Undergraduate Pilot Training in the 87th Flying Training Squadron. With Visual Basic software, the research provides a Graphical User Interface and develops scheduling rules in order to find an initial feasible solution. An iterative process which can be performed step by step by the

scheduler is provided for the purpose of developing a final schedule. To accommodate changes to the constructed schedule that occur daily, an attrition environment was created (Nguyen, 2002). Using Visual Interactive Modeling (VIM), the scheduler can adjust or iterate the daily schedule by changing that day's specific requirements. The process is repeated until the schedules for five days are completed (Nguyen, 2002). Parts of the applications in Nguyen's research can be very useful in building fighter squadron pilot schedules. Since the research focused on undergraduate pilot training, as in Aslan's research, the training missions and their sequence are predefined. This makes the problem less complex than the scheduling problem of an operational squadron.

Recent research on flight scheduling has been accomplished by Boyd, Cunningham, Gray and Parker (2006). They use a Premium Solver Platform and the relatively new approach of using a minimum cost network flow model to build weekly flight schedules. The authors stated the model needed further improvements in order to be applied to real life fighter squadron scheduling problems.

In their research, a flight day is divided into two periods named AM GO and PM GO. A total of 1990 variables, 260 integer variables and 3463 constraints are needed based on these two periods. The authors recommend constructing a total of 16 one-hour periods (Boyd et al., 2006). Use of two periods in a flight day may cause the omission of a pilot during the whole day in the model although the pilot is not available for only 2 hours; the last hour of the AM GO period and the first hour of the PM GO.

Another limitation of the research is a "feast or famine" cycle during the scheduling process. The program takes the flight hours of the previous week as an input

but the model does not update the individual flight hours with the corresponding week's flights when scheduling. So, if a pilot has less flight hours than the others in the previous week, the model may assign this pilot to more flights than are required which sometimes may cause a "feast or famine" cycle for individual pilots (Boyd et al., 2006).

An important improvement concerning "hard schedule" entries is recommended by the authors. The Director of Operations (DO) may have a direct input to the schedule such as a specific pilot-mission matching or specific IP for an inexperienced pilot. So, the authors recommend including the hard entries of the DO in the model (Boyd et al., 2006)

The authors construct the model with 5 stages of nodes. The first stage has the supply nodes, the second stage has the airspace nodes, the third, fourth and fifth stages have scheduling priorities, pilot qualifications, and pilot nodes, respectively. Each duty requiring a pilot is represented by a supply node. A path drawn between two nodes represents a valid path in the flow (Boyd et al., 2006). Figure 3 shows their flow model.

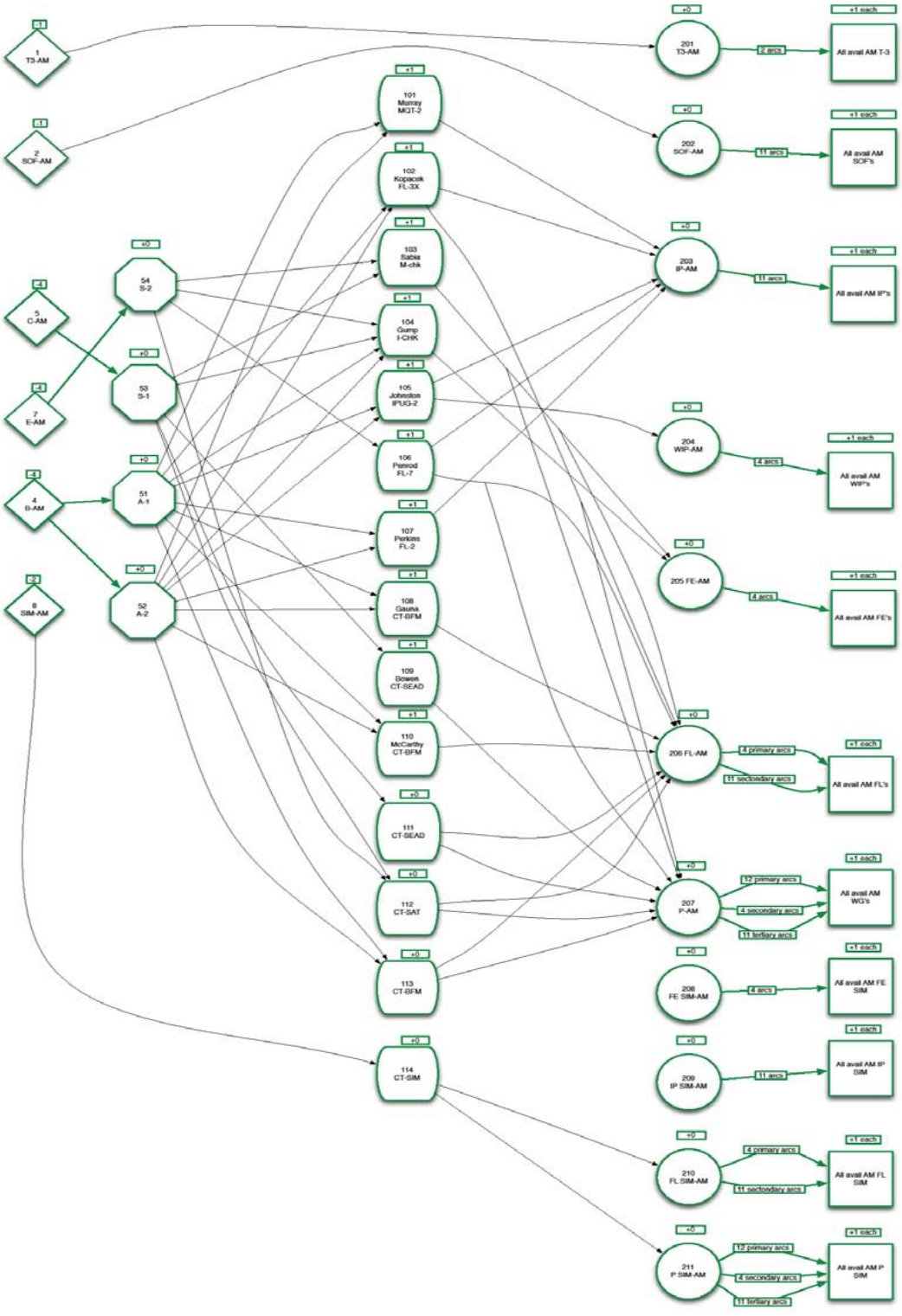


Figure 3. Monday AM Network Flow Model (Boyd et al., 2006)

Newlon's research uses a mathematical programming approach to the pilot scheduling problem. It uses binary variables for pilot assignments to the missions. Newlon uses a heuristic to assign pilots to the simulators based on the feasible solutions obtained from two different formulations and their solution approaches (Newlon, 2007).

Two different approaches are presented by Newlon. One of the two approaches is to build a pilot schedule for the entire week. The second approach is to divide the weekly scheduling problem into sub-problems and to solve each sub-problem. The sub-problem solutions are combined to form a good solution to the weekly scheduling problem. With this approach, Newlon divides the weekly flight schedule into ten periods comprised of AM GO and PM GO periods for Monday through Friday. Newlon's research has some important improvements over the research of Boyd et al. (2006). Newlon's approach updates the flight hours in each sub-problem by using the results from the previous sub-problem as an input for the following sub-problem (Newlon, 2007)

The research assumes that other pilots are always available for the seats that the squadron cannot fill. The objectives of Newlon's research are "to minimize the cost of utilizing non-squadron pilots and to minimize the penalties associated with non-compliance of sortie requirements" (Newlon, 2007).

The sub-problem methodology is recommended for use with future research (Newlon 2007). Decreasing the size of the problem by dividing it into sub-problems means Premium solver can be used more efficiently in Newlon's research. Also, this creates the opportunity to increase the number of variables and constraints for each sub-

problem which could cause a violation of the limits of the Solver if the increase were done in the main problem.

In Newlon's research, a network representation of the scheduling model is presented. Newlon's research provides many applications which could be a resource for the fighter squadron pilot scheduling problems. The key item, which is presented by Combs and Moore (2004), is to build a solution based on partial solutions (Newlon, 2007).

Gokcen's research focuses on building robust schedules for fighter squadrons. First, a basic scheduling model is created. Schedules created by the basic scheduling model are stressed by 10 different disruption types. Then the disrupted schedules are rescheduled, minimizing the total number of changes with respect to the previous schedule's objective function. Output schedules are ordered from the minimum to the maximum mean value of the total number of changes. The schedules which have the least mean value for the total number of changes are the most robust schedules (Gokcen, 2008). In Gokcen's research, there are some assumptions such as no two-seat aircraft which means no IP, that drive the research, but these assumptions cannot be made in modeling the general fighter squadron pilot scheduling problem.

Stojkovic and Soumis present "An Optimization Model for the Simultaneous Operational Flight and Pilot Scheduling Problem". Disruptions influence the planned flight schedule. Quick modifications to flight and crew scheduling and aircraft assignments are usually required. They propose an original approach that can solve crew and flight scheduling problems simultaneously (Stojkovic and Soumis, 2001). This approach divides the overall problem into sub-problems each consisting in scheduling

one pilot with the required qualification for each flight (Stojkovic and Soumis, 2001). As opposed to the traditional approach which first readjusts the flight schedule, then revises aircraft assignment and finally reconstructs the crew schedule, this approach optimizes all three objectives simultaneously. They use an integer nonlinear multicommodity minimum cost network flow model with time windows to formulate the problem. During the optimization process, a Dantzig-Wolfe decomposition and a Branch and Bound technique combination is used. In shorter time periods, better solutions than the traditional approach are obtained (Stojkovic and Soumis, 2001).

2.2.2 Heuristic in Scheduling

Even with an improved theoretical model of scheduling, modeling real life situations are extremely difficult due to their complexity and the probability of unexpected incidents. Pinedo writes;

> Almost all scheduling problems in the real world are strongly NP-hard; it would take a very long time to find an optimal solution on a PC, or even on a workstation.
>
> In practice, the scheduling environment is usually subject to a significant amount of randomness; it does not pay therefore to spend an enormous amount of computation time to find a supposedly optimal solution when within a couple of hours, because of some random event, either the machine environment or the job set changes. (Pinedo, 2008)

Because of this randomness and the strongly NP-hard nature of scheduling problems, schedulers do not always look for the optimal solution. In order to solve these hard problems, a heuristic is often used.

A heuristic can be defined as any method that looks for a “good” solution but not necessarily the best solution in a reasonable amount of computational time. Solution

quality, computational efficiency, robustness, consistency, ease of implementation and flexibility define the quality of the heuristic. Many different heuristics have been used for scheduling problems. Genetic Algorithms, Tabu Search, Simulated Annealing and Greedy Randomized Adaptive Search Procedure (GRASP) are heuristics.

As an example, Dowsland uses tabu search to solve the problem of producing rosters for a nursing staff in a large general hospital (Dowsland, 1998) and afterwards Aickelin and Dowsland use an Indirect Genetic algorithm to improve the solutions of the nurse scheduling problem which is similar to the fighter squadron pilot scheduling problem (Aickelin and Dowsland, 2004).

2.3 Greedy Adaptive Randomized Search Procedure (GRASP)

The GRASP (Greedy Randomized Adaptive Search Procedure) meta-heuristic is an iterative process. Each iteration consists of two phases: construction and local search. A feasible solution is built in the construction phase, and then in the local search phase, its neighborhood is investigated until a local minimum is found. The best overall solution is kept as the result (Resende and Ribeiro, 2003).

In the first phase, feasible solutions are constructed sequentially with one element in each iteration. Within each iteration, all feasible elements are ranked according to an adaptive greedy function based on the current state, and one is randomly selected from a restricted candidate list (RCL). The entire procedure is repeated many times. So, building an efficient construction phase is imperative (Rojanasoonthon and Bard, 2005).

One of the key points is to define how to construct the RCL. A greedy parameter "α " reflects the tradeoff between randomness and pure greedy function. This parameter

has a value between zero and one. For instance, in a maximization problem according to Equation (2.1), in the RCL, there is only the best element if "α" is chosen to be one or the RCL includes all feasible elements if "α" is chosen to be zero. Equation (2.1) defines the minimum required value in order for an element to be in the RCL. In Equation (2.1), Minimum refers to the minimum valued feasible element and Maximum refers to the maximum valued feasible element in the corresponding iteration. The number and the quality of the elements in the RCL vary according to the "α" value.

$$\text{Minimum Required Value} = \text{Minimum} + \alpha * (\text{Maximum} - \text{Minimum}) \qquad (2.1)$$

The heuristic is adaptive because the benefits associated with every element are updated at each iteration to reflect the changes caused by the selection of the previous element (Prabhaharan et al., 2006). The element to be used in the partial solution is randomly selected from the RCL. This is where the probabilistic aspect of the heuristic occurs. After the selected element is applied to the partial solution, the candidate list is updated and the values or costs of the elements are reevaluated; this represents the adaptive aspect of the heuristic (Resende and Ribeiro, 2003). An example pseudo code of the GRASP construction phase can be seen in Figure 4.

```
procedure Greedy_Randomized_Construction(Seed)
1    Solution ← ∅;
2    Evaluate the incremental costs of the candidate elements;
3    while Solution is not a complete solution do
4         Build the restricted candidate list (RCL);
5         Select an element s from the RCL at random;
6         Solution ← Solution ∪ {s};
7         Reevaluate the incremental costs;
8    end;
9    return Solution;
end Greedy_Randomized_Construction.
```

Figure 4. Pseudo-code of the construction phase (Resende and Ribeiro, 2003)

The solutions generated by a greedy randomized construction are not necessarily optimal. The local search phase usually improves the constructed solution in an iterative way that includes a search of a neighborhood for a better solution and replacing the current solution if a better solution is found (Resende and Ribeiro, 2003).

```
procedure Local_Search(Solution)
1    while Solution is not locally optimal do
2         Find s′ ∈ N(Solution) with f(s′) < f(Solution);
3         Solution ← s′;
4    end;
5    return Solution;
end Local_Search.
```

Figure 5. Pseudo-code of the local search phase (Resende and Ribeiro, 2003)

In GRASP, each iteration produces a sample solution from an unknown distribution and the nature of the RCL defines this sample solution's mean and variance. For example, if the RCL is restricted to a single element (only the best value is accepted to the RCL, $\alpha=1$), then the same solution will be produced at all iterations. The variance of the distribution will be zero and the mean will be equal to the value of the greedy solution. If the RCL is allowed to have more elements ($\alpha<1$), then the solutions may

differ in each iteration which causes a larger variance. Because of the randomness in this case, the mean solution value should be worse. However, the value of the best solution found outperforms the mean value and very often is optimal (Resende and Ribeiro, 2003)

An especially appealing characteristic of GRASP is the ease of its implementation. A parameter related to the stopping criterion and another one related to the quality of the elements in the RCL are considered to be the main parameters. After setting these parameters, the development can focus on implementing efficient data structures to assure quick iterations (Resende and Ribeiro, 2003).

2.3.1 GRASP in Scheduling

The Greedy Randomized Adaptive Search Procedure (GRASP) has been used in many types of scheduling problems. Prabhaharan, Kahn and Rakesh use GRASP to accomplish flow shop scheduling. Although many different heuristics, such as Genetic Algorithms and Simulated Annealing, have been used to solve flow shop scheduling problems with the objective of minimizing the makespan, the authors use only the construction phase of GRASP to solve this problem. Their computational experiments indicate that the GRASP Algorithm outperforms the traditional NEH (Nawaz, Enscore and Ham, 1983) algorithm which is a popular heuristic for permutation flow shop problems. The authors claim that GRASP may give an optimal solution in the construction phase itself, but sometimes a local search may be needed to find improved solutions (Prabhaharan et al., 2005).

Another implementation of GRASP in scheduling is performed by Rojanasoonthon and Bard. They focus on the problem of parallel machine scheduling with time window constraints. A further complication of the problem is the unique

priority classes of the jobs. A higher priority job is valued infinitely more than a lower priority job. So, a higher priority job is preferred to any number of low priority jobs. The problem is driven by NASA to improve the utilization of antennas on its tracking and data relay satellite system (Rojanasoonthon and Bard, 2005).

The authors divide the process into two phases. In the first phase, they start with an empty schedule and they choose a job from the RCL which has a predetermined size. By limiting the size of the RCL, the authors can choose a job from a predetermined number of best jobs. After choosing the job to be inserted, the feasibility is checked, and if it is feasible, the job is inserted in the schedule and all parameters are updated. This process is repeated until no more jobs can be inserted (Rojanasoonthon and Bard, 2005).

In the second phase, the authors claim that improvement can be accomplished either by relocating a subset of current jobs and performing additional insertions to this empty spot or by removing a job from a machine and inserting two or more jobs in its place (Rojanasoonthon and Bard, 2005).

The authors claim that GRASP can find good feasible solutions to practical sized problems in a reasonable amount of time. They compare the results of GRASP to the optimal value and to the Reddy-Brown heuristic which was the only other methodology for the Parallel Machine Scheduling with Time Windows problem (Rojanasoonthon and Bard, 2005). The results can be seen on Table 1.

Table 1.Comparison of GRASP and R-B Solutions with Optimal Solutions (Rajansoonthon and Bard, 2005)

Problem	Optimal value	GRASP solution	R-B solution	GRASP % gap	R-B % gap	CPU times (sec)		
						CPLEX	GRASP	R-B
lpltw1-20	16	16	16	0	0	8,993	0.34	0.01
lpltw2-20	17	17	13	0	23.53	2,522	0.33	0.01
lpltw3-20	17	17	13	0	23.53	2,520	0.34	0.01
lpltw4-20	17	17	13	0	23.53	2,525	0.33	0
lpltw5-20	16	15	13	6.25	18.75	478	0.33	0
lpttw1-20	12	11	12	8.33	0	7	0.22	0
lpttw2-20	13	13	13	0	0	2	0.33	0.01
lpttw3-20	13	13	13	0	0	1	0.35	0
lpttw4-20	13	13	13	0	0	1	0.35	0
lpttw5-20	13	13	11	0	15.38	1	0.3	0
rand1-20	14	14	13	0	7.14	12	0.25	0
rand2-20	14	14	13	0	7.14	469	0.4	0
rand3-20	14	14	13	0	7.14	468	0.4	0
rand4-20	16	16	15	0	6.25	5	0.28	0
rand5-20	16	16	15	0	6.25	5	0.29	0
spltw1-20	19	19	15	0	21.05	146	0.35	0
spltw2-20	19	19	15	0	21.05	149	0.35	0.01
spltw3-20	19	19	15	0	21.05	146	0.35	0.01
spltw4-20	17	17	15	0	11.76	24	0.33	0.01
spltw5-20	17	17	15	0	11.76	25	0.33	0
spttw1-20	14	14	13	0	7.14	2	0.34	0
spttw2-20	14	14	13	0	7.14	1	0.35	0
spttw3-20	14	14	13	0	7.14	1	0.35	0
spttw4-20	13	13	11	0	15.38	1	0.31	0
spttw5-20	13	13	11	0	15.38	1	0.23	0.01

This chapter focused on previous research on flight scheduling problem and heuristic solution of scheduling problems is represented. Chapter 3, a relatively new approach, implementation of GRASP to the Turkish Air Force fighter squadrons' pilot scheduling problem is presented.

3. METHODOLOGY

In this research, a relatively new approach is used for the weekly Turkish Air Force Fighter Squadrons' pilot scheduling problem. Only the construction phase of GRASP is applied to the problem. To apply GRASP to the pilot scheduling problem, each pilot-mission matching is graded according to the scheduler's preferences, mission requirements and pilot qualifications. At the end of this grading process, GRASP is applied. Due to the complexity, nonlinearity and the dynamic environment of the weekly schedule, the local search phase of the GRASP becomes nearly impossible to apply. So, the construction phase of GRASP is applied with different "α" values and the process is repeated many times in order to find a good solution. One handicap of the problem is not having an explicit optimum solution. Since the preferences of the scheduler can vary over time or from person to person, it strongly affects the grading process and the solution correspondingly. So, the objective of the research is defined as building a weekly pilot schedule that helps the scheduler prevent pilots from losing their currency for any mission and provides pilots an equivalent number of sorties in a month.

3.1 Inputs for Grading

In Chapter 1, phases of building a weekly schedule process are represented. Since this research focuses on the pilot assignment phase of scheduling, a *flight schedule* consisting of pre-scheduled missions with takeoff times, landing times and category requirements is needed. This flight schedule is built based on both the scheduler's and the headquarters' (HQ) inputs (HQ dictates the operational flights). Pilot assignments and thus the weekly pilot schedule are done with respect to this flight schedule.

Before starting to construct a weekly pilot schedule, the question of “Which pilot-mission matches are more important for the scheduler?” or “What are the scheduler’s preferences?” has to be answered. The answer to these questions may be a match of a pilot and a mission for which the pilot is at the currency limit or the answer may be a match of a pilot who has flown fewer sorties than other pilots and a mission. These are the two major concerns of the schedulers. So, each pilot’s number of sorties and remaining time to the currency limit for the associated mission must be taken into account for each pilot-mission match when grading.

The grading process is comprehensive. In order to grade a pilot-mission match, remaining time to the currency limit for the pilot, the pilot’s number of sorties in the associated month, the pilot’s status, the pilot’s category, and the availability of the pilot must be considered. During the grading process, all these factors represent an input.

3.1.1 Remaining Time to the Currency Limit

Each mission has its own currency limit in terms of days. The currency limit may differ depending on the mission. This currency limit represents the maximum number of days that a pilot can spend between two consecutive flights of the same mission in order to be able to fly the mission without an IP’s supervision. If a pilot does not fly a mission before exceeding the mission’s currency limit, then the pilot must fly the next sortie of the mission under the supervision of an IP. This may create a problem for the scheduler or may cause the overloading of an IP. So, to maintain each pilot within the currency limit of each mission, a proper grading must be accomplished for each pilot-mission match according to the time remaining to the currency limit. This grading must force the algorithm to assign the pilots who are closer to the currency limit of the corresponding

missions. So, as a pilot approaches the currency limit of a mission, the grade for that pilot-mission match has to be increased.

3.1.2 Number of Sorties Flown In the Same Month

In the Turkish Air Force, all the F-16 pilots who are active flyers (pilots in the squadrons) are required to fly at least 10 sorties a month. Because of this requirement, the schedulers tend to assign the pilots with a fewer number of sorties to the appropriate missions instead of the pilots with a greater number of sorties. This tendency has to be taken into account when grading a pilot-mission match.

The number of sorties a pilot needs to reach 10 sorties in a month and the number of available flight days until the end of the month play a key role when grading a pilot-mission match. Because of TDYs or other duties outside the squadron, the number of available flight days may vary for each pilot. So, the ratio of the number of remaining sorties to the available flight days influences the grade of the pilot-mission match with respect to the10 sortie requirement.

3.1.3 Pilot's Status

For the Turkish Air Force F-16 squadrons, 4 different pilot statuses exist; Instructor Pilot (IP), 4 Ship Wing Leader (4LD), 2 Ship Wing Leader (2LD), and Wingman (W). Although most of the squadrons have Check Pilots (CP), since they are all IPs and since the check flight is a rare event, check flights and CPs are not covered in this research. They require manual input by the scheduler.

A pilot cannot be assigned to a mission that requires a higher status. However, a pilot can fly a mission with the same or lower status. Although the pilot is allowed to fly

a mission in a lower status, it is not preferred. IPs are exceptional in this case because all IPs may serve as 4LD and this is not viewed as a drawback.

To balance the grade of the pilot-mission matches, the numbers 0.8 and 0.5 are defined as multipliers for the pilot-mission match grade of the pilots who fly a mission with one degree lower status and two degrees lower status, respectively. For example, if a 4LD is assigned to a mission as a wingman, then during the grading process the multiplier of 0.5 is used as an input. Table 2 shows the multipliers as a function of pilot status and the required status for the mission.

Table 2. Status Multiplier Matrix

REQUIRED STATUS	*PILOT STATUS* **IP**	**4LD**	**2LD**	**W**
IP	1	0	0	0
4LD	1	1	0	0
2LD	0.8	0.8	1	0
W	0.5	0.5	0.8	1

3.1.4 Pilot's Category

The limitations concerning pilot categories are explained in Chapter 1. A pilot cannot fly a mission that requires a higher category than the pilot's category. As the category number decreases, the permissible visibility and the ceiling limits decrease. A category 1 (CAT1) pilot can perform take off and landing under some weather conditions that other category pilots cannot. Categories can be listed from the highest to the lowest

category as CAT1, CAT2, and CAT3. So, a multiplier of zero or one is used when grading. For each pilot-mission match, the total grade is multiplied by zero if the mission requires a higher category or it is multiplied by one if the required category is lower than or equal to the pilot's category. The matrix in Table 3 shows the matrix for the category multiplier.

Table 3. Multipliers according the Categories

MISSION CATEGORY		*PILOT CATEGORY* ***CAT1***	***CAT2***	***CAT3***
	CAT1	1	0	0
	CAT2	1	1	0
	CAT3	1	1	1

3.1.5 Pilot Availability

It is nearly impossible to have all the pilots in the squadron available all the time. Some pilots may have other duties such as Quick Reaction Alert (QRA) duty, SFO duty or sometimes may be on leave. Other than these instances, a flight takes 3 additional hours of pilot time due to the briefing and debriefing of the flight. The pilot is not available during this time period. Hence, the pilot should not be assigned to any mission that occupies any time interval in this time period. In order to prevent the program from assigning a pilot who is not available at that moment to a mission, the grades of the pilot-mission matches must be updated according to the availability of the pilots. To

accomplish the proper update, the total grade of a pilot-mission match is multiplied by one if the pilot is available for the mission or it is multiplied by zero if the pilot is not available.

3.1.6 Calculating the Grades

As the first input, last flight days are used to initiate the calculation. The following formula is used for calculating the first input of the grades.

$$Grade1_{i,j} = 180+[\text{Flight Date}_{i}-\text{Last Flight Date}_{mission(i),j}]-\text{Currency Limit}_{mission(i)} \quad (3.1)$$

Where;

i: Number of the pilot-mission match on the schedule, i=1,2,…,285

j: Number of pilot , j=1,2,…, number of pilots

mission(i): Number of the mission type (including other duties) of i^{th} match, mission type=1,2,….,24

In the Equation (3.1), 180 is set as a constant because the largest currency limit is 180 days (for Air to Air Refueling (AAR)). Flight Date refers to the date of the mission of a pilot-mission match in the weekly schedule while Last Flight Date $_{mission\ type,\ j}$ represents the date of the j^{th} pilot's latest flight in the corresponding mission type. According to Equation (3.1), the grade of a pilot-mission match increases as the pilot approaches the currency limit of the mission. For example, for a pilot who flew the previous BFM mission 35 days before the date of a Basic Fighter Maneuvers (BFM) mission in the weekly flight schedule, the term [Flight Date$_i$ - Last Flight Date $_{mission\ type,j}$] is equal to 35. Since the currency limit of BFM missions is 60 days, the first part of the

grade is calculated as 180 + 35 - 60 = 155. The maximum number of 180 is obtained in the last day of the currency limit. The grade is increased by 50, 100 or 250 automatically for the pilots who are within 15, 10 or 5 days of the currency limit, respectively.

Since night flights are assumed to be critical flights, their grades are multiplied by 3 at the end the first part of the grading. Also, for the flights including an AAR event, AAR is assumed to be a different mission. Hence, a grade for AAR is calculated based on Equation (3.1) and added to the mission's grade.

The second input for the grades is made according to the pilot's number of sorties in the month and the remaining available flight days until the end of the month. Equation (3.2) is used to calculate the second input for the grade.

$$Grade2_{i,j} = 500*(20-nos_j) \,/\, \text{remaining available days in the month} \quad (3.2)$$

Where;

i: Number of the pilot-mission match on the schedule, i=1,2,…,285

j: Number of pilot , j=1,2,…, number of pilots

nos(j): Number of sorties flown by the j th pilot during this month

In Equation (3.2), a multiplier of 500 is used in order to assure assigning the pilots with a smaller number of sorties to a mission. Following this step, the grades calculated in Equations (3.1) and (3.2) are summed.

$$Grade_{i,j} = Grade1_{i,j} + Grade2_{i,j} \quad (3.3)$$

At the end of the grading process, the grades are updated according to the status, the category, and the availability of the pilots as explained in the next sections.

3.2 Updating Inputs in Each Iteration

Updating the inputs is a requirement. Otherwise, as in the research of Boyd, Cunningham, Gray and Parker, lack of timely update may cause a feast or famine cycle (Boyd et al., 2006). Newlon improved the scheduling process by using the results of a sub-problem as an input for the following sub-problem (Newlon, 2007). In Newlon's research, a weekly schedule is built beginning with the Monday AM period and ending with the Friday PM period. Even though it represents an improvement in scheduling, in real world situations it may not be adequate. For example, if a flight on Wednesday PM period is more important for a specific pilot and if the pilot is assigned to a flight during the Wednesday AM period without looking at the Wednesday PM period flights, due to the takeoff and landing time or briefing and debriefing time of the flight during the Wednesday AM period, assigning the pilot to the flight during the Wednesday PM period may become impossible. This yields an extremely dynamic environment in which each flight may affect the importance of every other flight. Because of the remaining days to the currency limit, the number of sorties and pilot availability inputs change after each assignment. So, in this case, the weekly scheduling problem cannot be divided into sub-problems, and it is important to look at the entire weekly schedule at once.

After assigning a pilot to a mission, the number of sorties of the pilot has to be increased by one, the remaining time to the currency limit must be updated according to the date of the mission and the pilot must be shown as unavailable during the briefing time, flight time and debriefing time. An exception takes place for night flights. Due to

crew rest requirements, the pilot who is assigned to a night flight must be made unavailable between the beginning of the day and 12 hours before the night flight's landing time and the 12 hours after the night flight's landing time. One more update has to be done in order to prevent a double assignment for the same mission. So, after assigning a pilot to a mission, all grades related to that mission must be set to zero.

3.3 Manual Matches

Other than the scheduler's concerns such as number of sorties and currencies, the squadron commander, director of flight or even the scheduler may prefer a pilot-mission match no matter what its grade is. There may be different reasons for this. A more experienced IP may be preferred to fly with a new pilot in the squadron or it may be preferred to assign a pilot to a mission if it is needed for the improvement of the pilot's skills on that mission.

Given a strict input, the scheduler makes it manually. At the beginning of the scheduling process required updates must be made for the assignments that are made manually by the scheduler. These required updates are the same as the updates discussed in Section 3.2.

3.4 Input Matrices

Four different matrices are constructed to supply the inputs such as number of sorties, last flight day of the mission, pilot status, pilot category, available days of the pilots in the month, and available times of the pilots during the day. For each pilot-mission match, the information in these matrices are used with the scheduler inputs consisting of mission type, takeoff time, landing time, category requirement and manual

matches, if they exist, to build a matrix of grades whose columns represent the pilots and whose rows represent each seat that must have a pilot. This matrix of grades is used directly in the GRASP implementation. In the matrix of grades, each value represents the benefit of an associated pilot-mission match. If a value is zero, then assigning the pilot to that mission is infeasible.

3.5 GRASP Implementation

Since, in general, the greater the grade the more the match is preferred, the scheduling algorithm chooses the higher graded pilot-mission matches. However, before implementing the algorithm, one point has to be clarified. Even though a pilot in a higher status can fly a mission that requires a lower status, for a flight requiring an IP, only an IP can perform the flight. So, IPs are considered as critical elements in the weekly schedule. Because of that, assignments of IPs and first pilots (FP) are made separately. This separation means the GRASP implementation has 4 phases.

3.5.1 Phase 1: Updates for Manual Matches

As mentioned in Section 3.3, at the beginning of the GRASP procedure, the matrix of grades is updated according to “strict inputs”. The updating process is the same as it is described in Section 3.2. The GRASP algorithm should not be run prior to these updates. Otherwise, assignments which violate the strict inputs may occur.

3.5.2 Phase 2: IP Assignments

A matrix of grades consisting of only the columns related to IPs and only the rows related to IP required flights is used in this phase. For this matrix, category number and

the status of the pilots are ignored since all IPs have the highest category number and the highest status. For an IP, there is no restriction in terms of status or category.

$$\text{Criteria} = \text{Minimum} + \alpha * (\text{Maximum} - \text{Minimum}) \quad (3.4)$$

When the algorithm is initiated, the maximum and the minimum entries in the matrix of grades are found and used to determine the criteria value which is shown in Equation (3.4). The criteria value defines the minimum grade that a pilot-mission match has to exceed for being included in the RCL. After determining the criteria, all entries in the matrix of grades which means all IP-mission matches are checked and the ones with a grade greater than or equal to the criteria value are added to the RCL. At the end of this process, one of the matches is randomly selected from those on the RCL. In the equation, α is the greedy variable and ranges from 0 to 1. If α is set to 0, then the criteria is equal to the minimum grade in the matrix of grades which implies that all assignments are made randomly. On the other side, if α is set to 1, then the criteria is equal to the maximum grade in the matrix of grades which implies that only the best graded pilot-mission match is included in the RCL.

After assigning the pilot to the mission, all inputs are updated. The matrix of grades is rebuilt according to the updated inputs and the algorithm continues in the same fashion until all missions are matched with a pilot or until no available pilot is left, whichever occurs first.

3.5.3 Phase 3: First Pilot Assignments

All pilots flying in the front seat of an F-16 D aircraft or all pilots flying an F-16 C aircraft are called “First Pilot” (FP) regardless the pilot status. FP assignments are made almost the same as the IP assignments. The difference is in the matrix of grades. When building the matrix of grades, all IPs are considered as 4LD. The process continues until all missions are matched with a pilot or until no available pilot-mission match is left (which occurs when the criteria is equal to zero). If there is lack of available pilots for a mission, then the algorithm assigns a generic pilot to the mission with the name of “SPARE”. In this case, after the schedule is done, the scheduler must fix this assignment by either making changes to the created schedule or importing a pilot from another division such as headquarters.

3.5.4 Phase 4: Evaluation

At the end of the assignment process, the created schedule is evaluated. This evaluation is performed by computing the benefits of each flight. Benefits are calculated using the same process in the calculation of the grades. During the construction phase of the schedule, the grades are calculated based on current conditions. These grades cannot be used during the evaluation process. To evaluate the schedule, every pilot-mission match is evaluated starting from the first flight on Monday morning and going through the last flight on Friday. For the first flight on Monday, all the inputs are reset to the previous week’s values. After evaluating each flight, the inputs are updated based on the flight’s characteristics such as the date of the flight, takeoff and landing times, etc.

The whole process beginning with Phase 1 is repeated a number of times which is defined by the scheduler. At the end of each iteration, number of missing pilot-mission

matches and total benefit of the schedule are calculated. The created weekly pilot schedule is compared to the weekly pilot schedules found in other iterations in terms of the number of missing pilot-mission matches for flight duties, for other duties and the total benefit of the schedule. First, a schedule with a smaller number of missing pilot-mission match for flight duties is preferred. If there is a tie, the numbers of missing pilot-mission matches for other duties are compared and the schedule with the smaller number is preferred. If two schedules have the same number of missing pilot-mission matches for flight duties and for other duties, then the total benefits are compared and the pilot schedule with greater total benefit is preferred. The algorithm creates the best schedule based on the scheduler's preferences and the grading properties. A pseudo code of the process is presented in Figure 6.

```
Begin
Import the inputs
For i=1 to number_of_iterations
        Update the inputs according to the manual assignments
        While there exist a possible pilot-mission match
                While criteria ~= 0
                        Calculate the grades for each IP-mission match
                        Calculate the criteria using Eq. (3.4)
                        Find the matches whose grade >= criteria and import them to the RCL
                        Choose randomly one match from the RCL
                        Assign the pilot to mission
                        Update the inputs
                End
                While criteria ~= 0
                        Calculate the grades for each pilot-mission match
                        Calculate the criteria using Eq. (3.4)
                        Find the matches whose grade >= criteria and import them to the RCL
                        Choose randomly one match from the RCL
                        Assign the pilot to mission
                        Update the inputs
                End
                Reset all inputs to the previous week's values
                For j=1 to number of pilot-sorties in the schedule
                        Calculate the benefit of the jth flight on the schedule
                        Total Benefit = Total Benefit + the benefit of the jth flight
                        Update the input based on first j flights
                End
        End
End
```

Figure 6. Pseudo Code for Implementation of GRASP

3.6 Simulator, SOF and RSU Duties

Simulator flights, SOF duties and RSU duties are treated as a flight mission, since all of these duties have their own beginning and ending times similar to a flight's takeoff and landing times. These duties are included in the matrix of grades as they are a flight mission. However, the grading process is different for these duties. SOF duties can be performed by IPs and 4LDs while RSU duties are performed by Ws and 2LDs. Simulator flights can be performed by all pilots. The corresponding pilots are considered as

candidates for these duties. Then the matches for these candidates are graded according to a comparison of their past performance for the corresponding duty to the average performance of the other pilots. For example, if a pilot has performed only 2 RSU duties during this month and if the average is 4, then the pilot-RSU duty match will receive a higher grade.

A different application takes place for the simulator sortie. All of the Turkish Air Force F-16 Air Bases do not accommodate simulators. So, the pilots at a base without a simulator go to another base for simulator flights and when a pilot goes to another base for a simulator flight, the pilot becomes unavailable for any other event during the day.

During the analysis phase, Equation (3.5) is used to compute the grades of pilot-simulator matches. The goal of this equation is to have equivalent numbers of simulator sorties for each pilot.

$$Grade_{i,j} = 25+25*(\text{ Total sim })/\text{Number of pilots} - sim_j \quad (3.5)$$

Where;

j: Pilot number, j=1,2,…,number of pilots

sim_j: Number of simulator sorties flown by pilot j in that month.

sim: Simulator sorties

Equation (3.5) is used for RSU and SOF duties as well. Status and category updates are performed after the calculation.

Figure 7 represents a flow chart of the algorithm and gives a general idea about how the process is accomplised.

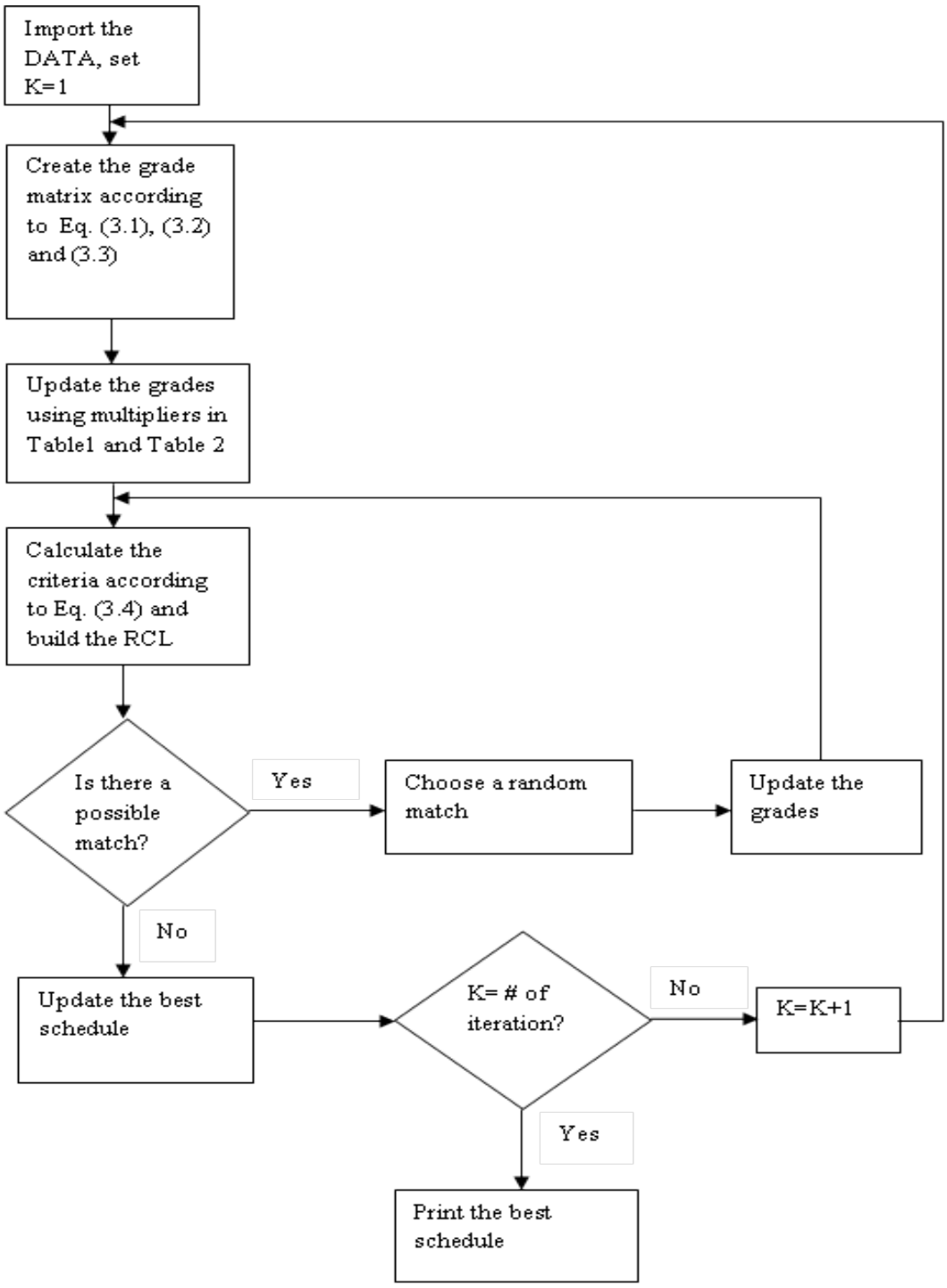


Figure 7. Flow Chart of the Overall Process

In this chapter, an implementation of GRASP for the weekly pilot scheduling problem is presented. The next chapter presents the analysis of the algorithm and the results of algorithm execution.

4. ANALYSIS

4.1 General

In this chapter, a MATLAB code using GRASP to accomplish fighter pilot scheduling is implemented. These implementations are accomplished under two main situations.

First, the code is used to create 4 consecutive weekly pilot schedules for the Turkish Air Force 192nd Fighter Squadron with respect to real weekly flight schedules provided by the same squadron. Some assumptions are made in this part in order to prevent overloading the code in terms of number of mission types and number of pilots.

Second, a randomly built weekly flight schedule is used by the algorithm to create weekly pilot schedules for four consecutive weeks and the reaction of the code is observed under three different scenarios. At the end of the each scenario, the number of flights, the number of SOF duties, the number of RSU duties and the number of simulator flights are calculated. The mean and the standard deviation of the distribution of flights, SOF duties, RSU duties, and simulator flights are also are checked.

4.2 The Real Life Weekly Schedule

For this part of the analysis, 4 weekly flight schedules for four consecutive weeks are imported from 192nd Fighter Squadron of the Turkish Air Force. Besides the most common missions, some rare missions are included in these schedules. Hence, all of these rare missions are named as “miscellaneous” flights (MISC) in order to ease the implementation. Also, the pilots in Mission Qualification Training (MQT) and the pilots from other divisions are inserted in the schedule as the scheduler’s manual inputs. All the

trainees are inserted in the schedule with their own names but for the pilots coming from other divisions the word “XXXXX” is used in the manual inputs. So, the data for the other divisions’ pilot are not tracked. Additional assumptions are made to decrease the workload of the code.

4.2.1 Assumptions

The following assumptions are made for this part:

- All night flights are performed with Night Vision Goggle (NVG).
- The Air to Air Refueling (AAR) mission and the Night Air to Air Refueling (NAAR) mission are considered to be the same.
- There are a total of 10 different mission types; Instrument Flight (IF), Intercept (INT), Basic Fighter Maneuvers (BFM), Air Combat Maneuvering (ACM), Air Combat Tactics (ACT), Air to Ground (AG), Operational Flights (OPER), Miscellaneous Flights (MISC), Night Vision Goggle (NVG), Air to Air Refueling (AAR).
- The currency limit is 60 days for all mission types except for NVG and AAR which have 45 and 180 day limits, respectively.
- The two hours briefing time can be decreased to 90 minutes but a penalty must be applied.
- The first goal of the scheduler is to provide pilot-mission matches. Providing pilots for other duties is the second goal.

- All flights in the weekly schedules are assumed to be flown by the assigned pilot and no aborts occur.

- All runs include 50 iterations and a variable α value which is set to 0.99 at the beginning of each run and which is decreased by 0.006 in each iteration.

- There are five IPs, nine 4LDs, three 2LDs and eleven Ws in the squadron.

4.2.2 Results of the Real Life Scenario

The scheduling algorithm is applied to four different flight schedules. At the end of each week, the number of sorties, the number of SOF duties, the number of RSU duties, the number of simulator flights and the last flight day information are calculated and are used as the inputs for the following week. Although the given schedules are real schedules, they do not include the final revisions made by the scheduler. This causes the schedule to have infeasible entries by the scheduler.

All four schedules have different numbers of infeasible entries. The infeasibilities are caused by the violation of briefing and debriefing time constraints. Sometimes due to the density of the flight missions, pilots can have a short briefing called “step briefing” or can make the briefing one day before the flight day. Also, debriefings can be left to the end of day. These instances are impossible to model in the code and only the scheduler can decide to have those kinds of instances after agreeing with the leaders of the corresponding flights.

Two different scenarios are applied. First, the code is run for each week, assuming all previous weeks were scheduled as it is in the real schedules. So, in this case, four different weekly schedules are created and created weekly schedules are independent.

Second, starting from the first week, all four weeks are scheduled and each created schedule's data are used as inputs for the following week.

4.2.2.1 Scheduling Each Week Separately

In this part, four independent weekly pilot schedules are created based on the data from the real squadron flight schedules. At the end of the scheduling process, the scheduled activities, such as number of sorties, last flight days, etc., are not carried to the following week. The inputs for the following week are based on the data of the corresponding real squadron flight schedule. In this part of the research, the creating capability of the scheduling algorithm is monitored and evaluated.

Table 4. Results For Scheduling Each Week Separately

Week	Total Pilot-Sortie[1]	Requested Matches[1]	Missing Matches[1]	Infeasible Matches[3]	Time (sec)	Identical Matches[1]	Ratio (%)[2]
1	43+5	23+3	1+0	2	4.45	7+0	26.92308
2	122+12	55+11	3+3	8	14.59	20+4	36.36364
3	103+12	52+9	7+3	10	11.59	14+3	27.86885
4	105+12	51+8	5+1	8	11.81	11+4	25.42373
						Average :	29.144825

[1] : Number of Flights + Number of Other Duties

[2] : Total Number Of Identical Matches / Total number of Requested Matches

[3] : Number of Infeasible Matches In the Real Schedule

As can be seen on Table 4, the code provides same pilot-mission matches as the real schedule for 29 percent of the scheduled sorties. This average could be larger but the infeasible entries in the real schedules and the large number of manual entries dramatically change the dynamic environment of the problem which causes the number of identical matches to be smaller. The scheduler's manual inputs are not included in the identical matches. Although the real schedules have a total of 28 infeasible entries, the computed schedules cannot provide a total of 23 pilot-mission matches due to the infeasibility concerns. So, the computed schedule outperforms the real schedule in terms

of both feasibility and the number of missing pilot-mission matches. Less than fifteen seconds are spent to create each schedule. The creation time varies according to the number of required pilot-mission matches, number of pilots and the number of manual inputs.

In both the computed schedule and the real schedule, there is no violation of currency limit for any pilot. In this case, since the inputs are reset to the real world situation values at the beginning of each week, keeping the pilots within their currency limit is provided by the scheduler of the squadron.

4.2.2.2 Scheduling Four Weeks Consecutively

Four weekly flight schedules from a real world squadron are run through the code consecutively. The inputs are updated according to the calculated schedule of the previous week. For the first week, the real life data is used as inputs. As in the previous section, no violation of the currency limit occurs but the distribution of the sorties is changed.

In the real life scenario, two of the pilots are not available during the month and five of the pilots are available less than 15 days during the month. In such cases, the squadron scheduler has the right to ignore the requirements of those pilots who are available less than 15 days during the month (TUAF HKY 164-1C). There is not an input for this in the GRASP algorithm. Hence, the code tries to satisfy all requirements for these pilots which would be beneficial for the scheduler and the pilots. The squadron has four pilots in MQT, so these pilots are left out of the input data and all flights for these pilots are manually arranged by the squadron scheduler.

Table 5 compares the real life schedules and the computed schedules based on the number of sorties, the number of SOF duties and the number of RSU duties at the end of the fourth week.

Table 5. Comparison of the Results for Four Consecutive Weekly Schedules

	Real Schedules			Computed Schedules			
Pilot	Sorties	RSU	SOF	Sorties	RSU	SOF	Days
1	23	0	1	17	0	2	17
2	10	0	0	10	0	2	6
3	10	0	1	13	0	2	6
4	17	0	3	16	0	1	14
5	14	0	2	14	0	3	14
6	22	0	0	12	0	4	17
7	8	0	1	10	0	1	9
8	2	0	1	3	0	0	3
9	9	0	0	12	0	0	6
10	8	0	0	9	0	1	8
11	10	0	1	10	0	0	8
12	4	0	1	4	0	0	3
13	11	0	1	10	0	0	7
14	18	0	2	13	0	2	12
15	14	0	0	10	0	0	9
16	14	2	0	13	3	0	11
17	13	1	0	14	1	0	13
18	0	0	0	0	0	0	0
19	0	0	0	0	0	0	0
20	14	3	0	12	2	0	15
21	11	1	0	9	2	0	11
22	7	1	0	9	1	0	6
23	8	1	0	6	0	0	6
24	11	3	0	12	2	0	12
AVRG :	12.6		AVRG :	11.55			
STDV :	4.523622		STDV :	2.625282			

According to the Table 5, Pilot 8 and Pilot 12 are available for only three days during the month. This limits their assignments by the GRASP algorithm. In addition to Pilot 8 and Pilot12, Pilot 18 and Pilot 19 are also not included in the mean and standard deviation calculations since they are unavailable during the month. The only place that the real schedule seems to be better than the computed one is the number of sorties for

Pilot 23. Since Pilot 23 is available for only 6 days during month, Pilot 23 represents a critical situation. Although the scheduler assigns 2 sorties more than the code does to this pilot, both of these sorties are infeasible because of the briefing and debriefing time constraints. Additionally, a total of 28 matches in the real schedules are actually infeasible for this reason.

The code decreases the standard deviation of the number of sorties per pilot which means all pilots tend to fly nearly equal numbers of sorties during month and this satisfies one of the research's objectives. The standard deviation would decrease if the number of manual inputs decreases. After the fourth week, 2 flight days remain until the end of the month. So both the code and the scheduler can assign Pilot10 and Pilot19 more sorties in order to satisfy the 10 sorties per month requirement.

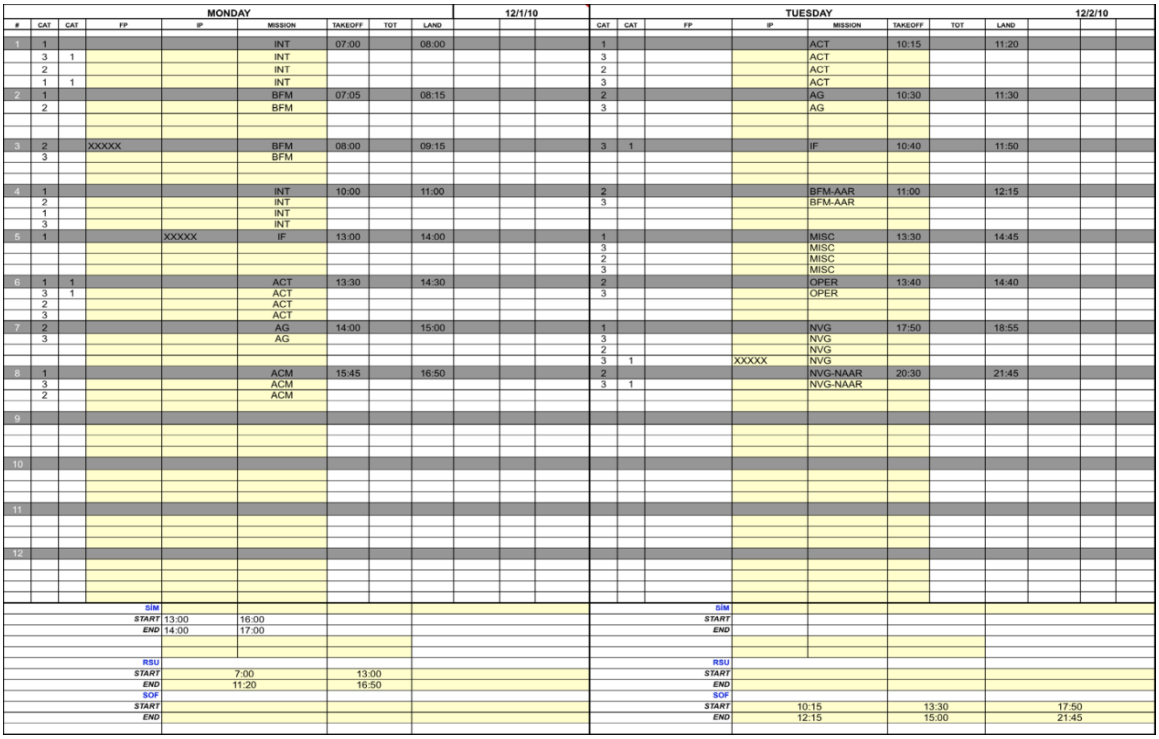
4.3 Manually Generated Schedule

For this part of the research, a manual *weekly flight schedule* which includes all 10 mission types is created. The code is run based on the same weekly flight schedule for four consecutive weeks and at the end of each week, the data is transferred to the following week in order to be used as inputs. Three different scenarios are tested. In the first scenario, the last flight dates of all missions are set to 30 days before the first day of the first week for all pilots. All pilots are assumed to be available during whole month. In the second scenario, some pilots are assumed to be available less than the other pilots during the month. In the last scenario, almost all last flight days are set randomly in a fashion that provides more than 40 days to the currency limit of the missions but some last flight days for some pilots are set to random dates that provides less than 20 days to

the currency limit of some missions. For all three scenarios, the code is run for four consecutive weeks and the results are collected.

4.3.1 The Generated Schedule

A weekly flight schedule that includes at least one flight of every mission type is manually created. None of the mission types, category requirements and the takeoff-landing times are defined intentionally. This schedule is used to test the code for the three scenarios.



MONDAY									12/1/10			TUESDAY									12/2/10		
#	CAT	CAT	FP	IP	MISSION	TAKEOFF	TOT	LAND				CAT	CAT	FP	IP	MISSION	TAKEOFF	TOT	LAND				
1	1				INT	07:00		08:00				1				ACT	10:15		11:20				
	3	1			INT							3				ACT							
	2				INT							2				ACT							
	1	1			INT							3				ACT							
2	1				BFM	07:05		08:15				2				AG	10:30		11:30				
	2				BFM							3				AG							
3	2		XXXXX		BFM	08:00		09:15				3	1			IF	10:40		11:50				
	3				BFM																		
4	1				INT	10:00		11:00				2				BFM-AAR	11:00		12:15				
	2				INT							3				BFM-AAR							
	1				INT																		
	3				INT																		
5	1			XXXXX	IF	13:00		14:00				1				MISC	13:30		14:45				
												3				MISC							
												2				MISC							
												3				MISC							
6	1	1			ACT	13:30		14:30				2				OPER	13:40		14:40				
	3	1			ACT							3				OPER							
	2				ACT																		
	3				ACT																		
7	2				AG	14:00		15:00				1				NVG	17:50		18:55				
	3				AG							3				NVG							
												2				NVG							
												3	1		XXXXX	NVG							
8	1				ACM	15:45		16:50				2				NVG-NAAR	20:30		21:45				
	3				ACM							3	1			NVG-NAAR							
	2				ACM																		
9																							
10																							
11																							
12																							
SİM												SİM											
START				13:00	16:00							START											
END				14:00	17:00							END											
RSU												RSU											
START				7:00		13:00						START											
END				11:20		16:50						END											
SOF												SOF											
START												START			10:15		13:30		17:50				
END												END			12:15		15:00		21:45				

Figure 8. Monday and Tuesday Flights In the Generated Schedule

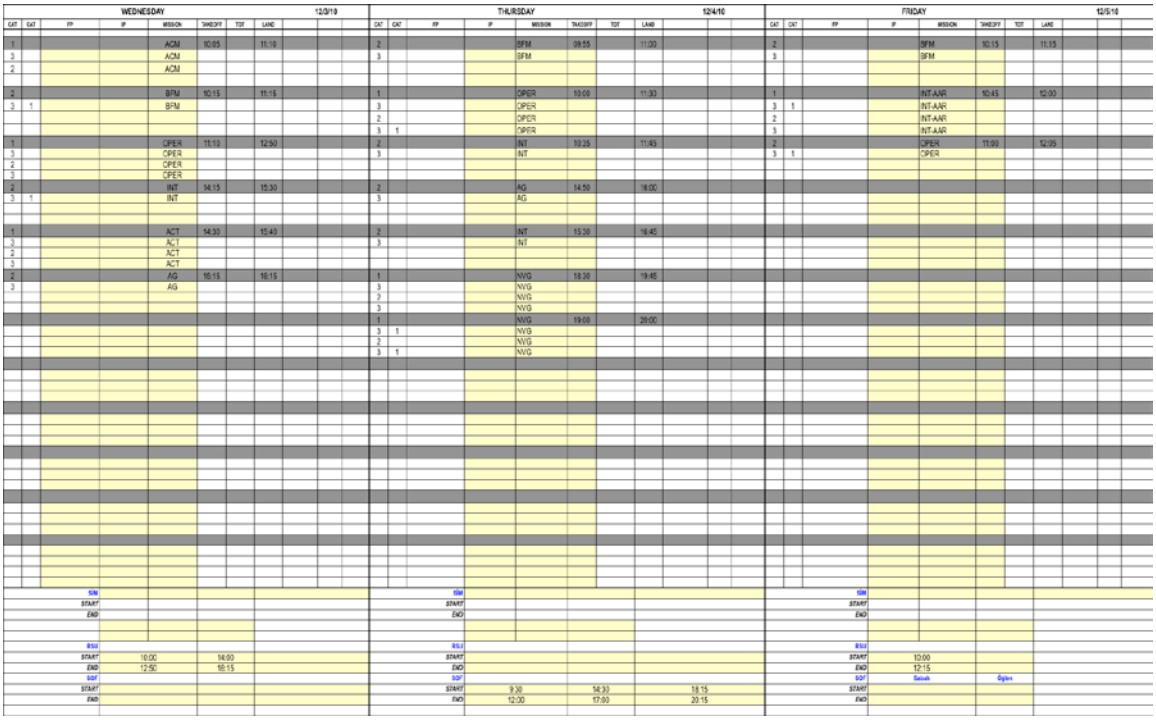


Figure 9. Wednesday through Friday Flights in the Generated Schedule

4.3.2 Assumptions

Before running the code for four weeks, the assumptions are the same as those of previous runs and are listed in Section 4.2.1. A total of 10 different mission types are included in the scenarios. For simulator duties, only Wingmen are allowed to be assigned.

4.3.3 First Scenario

For the first scenario, all last flight days are set to 11/3/2010 and the first day of the first week is dated to 12/01/2010. All pilots are assumed to be available for all flight days during the month. The code is run for four consecutive weeks and the results for each week are shown in Table 6. According to Table 6, sorties and other duties are nearly equally distributed among the pilots. The standard deviation of number of sorties per pilot

is lower than the standard deviation of the real squadron schedule scenario in Section 4.2.2.2.

In Table 6, NOS, SIM, RSU and SOF represent the number of sorties, the number of simulator duties, the number of RSU duties and the number of SOF duties, respectively. At the end of the month, an average of 14 sorties are flown by the pilots and the standard deviation is 0.763. Not only the sorties but also the other duties are distributed equitably. At the end of the fourth week, many of the pilots approach the currency limit of different missions as shown in Table 7. The reason for this is the lack of sorties to keep all the pilots within their currency limit.

Table 6. Number of Sorties and Other Duties for the First Scenario

I	1st week				2nd Week				3rd Week				4th Week				TOTAL			
PILOT	NOS	SIM	RSU	SOF	NOS	SIM	RSU	SOF	NOS	SIM	RSU	SOF	NOS	SIM	RSU	SOF	NOS	SIM	RSU	SOF
1	3	0	0	1	3	0	0	0	4	0	0	0	4	0	0	0	14	0	0	1
2	3	0	0	0	3	0	0	1	4	0	0	1	4	0	0	0	14	0	0	2
3	4	0	0	1	3	0	0	0	3	0	0	0	4	0	0	1	14	0	0	2
4	4	0	0	0	2	0	0	1	4	0	0	0	3	0	0	1	13	0	0	2
5	3	0	0	0	4	0	0	2	3	0	0	0	4	0	0	0	14	0	0	2
6	3	0	0	1	3	0	0	0	4	0	0	0	4	0	0	1	14	0	0	2
7	4	0	0	0	3	0	0	1	3	0	0	0	3	0	0	0	13	0	0	1
8	3	0	0	0	3	0	0	0	4	0	0	1	4	0	0	0	14	0	0	1
9	2	0	0	0	4	0	0	1	4	0	0	0	4	0	0	1	14	0	0	2
10	3	0	0	0	4	0	0	0	3	0	0	1	4	0	0	0	14	0	0	1
11	4	0	0	1	3	0	0	0	3	0	0	1	4	0	0	0	14	0	0	2
12	2	0	0	0	4	0	0	0	4	0	0	2	3	0	0	0	13	0	0	2
13	2	0	0	1	4	0	0	0	4	0	0	0	4	0	0	1	14	0	0	2
14	2	0	0	1	4	0	0	0	4	0	0	0	4	0	0	1	14	0	0	2
15	6	0	0	0	4	0	0	0	4	0	1	0	2	0	0	0	16	0	1	0
16	5	0	0	0	5	0	0	0	3	0	1	0	3	0	0	0	16	0	1	0
17	7	0	0	0	4	0	0	0	2	0	1	0	2	0	0	0	15	0	1	0
18	3	0	0	0	4	0	0	0	4	0	1	0	3	1	0	0	14	1	1	0
19	4	0	1	0	3	1	0	0	4	0	0	0	3	0	0	0	14	1	1	0
20	5	1	0	0	3	0	1	0	3	0	0	0	3	0	1	0	14	1	2	0
21	4	0	0	0	3	0	1	0	4	0	0	0	4	0	0	0	15	0	1	0
22	4	0	0	0	3	0	1	0	4	0	0	0	4	0	1	0	15	0	2	0
23	3	0	1	0	4	0	0	0	3	1	0	0	5	0	0	0	15	1	1	0
24	4	0	1	0	4	0	0	0	3	0	1	0	4	0	0	0	15	0	2	0
25	4	1	1	0	4	0	0	0	3	0	0	0	4	0	1	0	15	1	2	0
26	3	0	0	0	4	0	1	0	4	1	0	0	3	0	1	0	14	1	2	0
27	3	0	1	0	4	0	0	0	4	0	0	0	3	1	1	0	14	1	2	0
28	3	0	0	0	4	1	1	0	4	0	0	0	4	0	0	0	15	1	1	0
															AVRG :		14.29	0.73	1.43	1.71
															STDV :		0.763	0.47	0.51	0.47

Table 7. Remaining Days to the Missions' Currency Limit

MISSIONS	CURRENCY	PILOTS																											
	LIMIT	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28
IF	60	57	50	36	49	43	7	7	7	7	7	7	7	35	56	7	7	7	7	7	36	7	7	7	7	43	50	7	57
INT	60	49	58	60	49	59	49	60	60	52	59	56	51	56	56	56	42	58	52	56	60	53	58	45	52	59	53	60	49
BFM	60	37	58	44	42	49	44	56	58	43	50	36	60	37	59	51	57	56	43	45	58	59	57	42	53	60	49	56	46
ACM	60	7	49	42	51	7	42	49	51	56	44	44	35	58	37	35	58	56	35	56	7	7	42	49	7	44	58	51	37
ACT	60	56	56	49	56	49	37	35	57	49	51	49	56	50	58	50	58	43	58	43	56	57	50	51	57	49	35	44	58
AG	60	50	7	7	7	56	51	43	36	37	44	59	52	49	35	58	42	45	50	49	45	59	52	57	58	51	42	43	56
OPER	60	60	44	58	58	59	60	52	50	59	57	59	44	53	51	46	52	37	51	60	50	43	46	59	59	58	44	57	52
MISC	60	7	7	7	7	7	7	57	7	50	43	50	43	36	7	57	36	7	57	50	57	7	36	43	43	50	36	7	7
NVG	45	44	30	44	44	42	44	23	30	44	42	30	44	28	37	37	23	23	42	35	35	42	44	44	23	23	44	42	44
AAR	180	163	173	180	156	177	170	180	180	163	177	156	127	163	173	156	177	166	163	166	180	173	177	156	159	156	173	180	163

For example, since there are only 3 pilot-sorties in a week for IF, most of the pilots approach their currency limit, because these pilots could not perform an IF mission during the last four weeks. Similar to the IF, there is not an adequate number of MISC and ACM flights in the schedule.

4.3.4 Second Scenario

For the second scenario, Pilot 5 is set “not available” during the fourth week, Pilot 9 is set “not available” during the third week, Pilot 17 is set “not available” on Monday through Thursday of the fourth week and Pilot 22 is set “not available” on Thursday through Friday of the first week and Wednesday through Friday of the fourth week. So, Pilot 5, Pilot 9, Pilot 17 and Pilot 22 have 18, 18, 19 and 16 available flight days respectively during the month while the other pilots have 23 available flight days. The code is run for four consecutive weeks. The number of sorties and other duties are calculated at the end of each week. Table 8 gives insight about the assignments made by the code according to the available flight days. Especially, the schedules of Pilot 5, Pilot 9, Pilot 17 and Pilot 22 should be closely monitored.

Table 8. Number of Sorties and Other Duties for the Second Scenario

II	1st week				2nd Week				3rd Week				4th Week				TOTAL			
PILOT	NOS	SIM	RSU	SOF	NOS	SIM	RSU	SOF	NOS	SIM	RSU	SOF	NOS	SIM	RSU	SOF	NOS	SIM	RSU	SOF
1	2	0	0	0	4	0	0	1	4	0	0	0	3	0	0	1	13	0	0	2
2	4	0	0	0	2	0	0	0	3	0	0	2	5	0	0	0	14	0	0	2
3	2	0	0	1	3	0	0	0	4	0	0	0	5	0	0	1	14	0	0	2
4	2	0	0	1	3	0	0	0	5	0	0	0	4	0	0	0	14	0	0	1
5	6	0	0	0	4	0	0	1	4	0	0	0	0	0	0	0	14	0	0	1
6	4	0	0	0	2	0	0	1	4	0	0	1	4	0	0	0	14	0	0	2
7	2	0	0	1	4	0	0	0	4	0	0	0	4	0	0	0	14	0	0	1
8	2	0	0	1	3	0	0	0	5	0	0	0	3	0	0	0	13	0	0	1
9	6	0	0	1	3	0	0	0	0	0	0	0	5	0	0	1	14	0	0	2
10	2	0	0	0	4	0	0	0	4	0	0	3	4	0	0	0	14	0	0	3
11	3	0	0	1	3	0	0	0	4	0	0	0	3	0	0	0	13	0	0	1
12	2	0	0	0	4	0	0	1	3	0	0	0	6	0	0	0	15	0	0	1
13	1	0	0	0	5	0	0	1	4	0	0	0	4	0	0	2	14	0	0	3
14	3	0	0	0	3	0	0	1	3	0	0	0	4	0	0	1	13	0	0	2
15	6	0	0	0	4	0	0	0	3	0	1	0	3	0	0	0	16	0	1	0
16	6	0	0	0	4	0	1	0	3	0	0	0	3	0	0	0	16	0	1	0
17	7	0	0	0	5	0	0	0	3	0	1	0	0	0	0	0	15	0	1	0
18	5	0	1	0	3	0	0	0	3	0	0	0	4	1	0	0	15	1	1	0
19	4	1	0	0	3	0	1	0	4	0	0	0	4	0	0	0	15	1	1	0
20	2	1	1	0	5	0	0	0	4	0	0	0	5	0	1	0	16	1	2	0
21	3	0	0	0	4	0	0	0	4	0	0	0	4	0	2	0	15	0	2	0
22	2	0	0	0	7	0	1	0	4	0	0	0	3	0	0	0	16	0	1	0
23	4	0	0	0	3	0	0	0	3	1	2	0	5	0	0	0	15	1	2	0
24	4	0	1	0	3	0	0	0	4	0	0	0	2	0	1	0	13	0	2	0
25	4	0	1	0	3	1	0	0	4	0	0	0	2	0	0	0	13	1	1	0
26	4	0	1	0	3	1	0	0	4	0	1	0	3	0	0	0	14	1	2	0
27	4	0	0	0	3	0	1	0	3	0	0	0	4	1	0	0	14	1	1	0
28	4	0	0	0	3	0	1	0	3	1	0	0	4	0	1	0	14	1	2	0
															AVRG :		14.29	0.727	1.43	1.7143
															STDV :		0.976	0.467	0.51	0.7263

Even though four of the pilots are not available for different time periods during the month, at the end of the fourth week an average of 14 sorties and a standard deviation of 0.976 are achieved. For example, for Pilot 22, in the second week the algorithm assigns the pilot to 7 flights in order to assure an equivalent number of sorties for all pilots at the end of the month.

4.3.4 Third Scenario

For the third scenario, almost all of the last flight dates are set to dates that are randomly distributed between the first day of the first week (12/1/2010) and 18 days before this day (11/13/2010). Some last flight days are set relatively close to the currency limit of the corresponding mission. Remaining days to the currency limit for each pilot-mission match can be seen on Table 9. Gray colored matches are defined intentionally before running the algorithm.

Table 9. Remaining Days to the Currency Limits at the Beginning of the1st Week

MISSIONS	CURRENCY	PILOTS																										
	LIMIT	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28
IF	60	42	45	46	55	56	50	55	57	53	57	47	56	52	48	48	50	48	52	53	50	51	4	53	53	57	49	46	48
INT	60	48	54	53	50	13	42	56	55	47	50	52	55	56	51	52	42	47	48	44	53	52	53	49	54	5	47	45	45
BFM	60	47	57	45	42	47	43	52	48	43	47	48	56	47	43	47	57	47	46	55	50	56	44	52	50	5	53	57	54
ACM	60	42	55	46	56	53	42	55	43	57	56	52	45	42	6	47	51	50	45	50	56	52	55	48	50	5	53	51	56
ACT	60	54	49	52	54	56	49	54	55	48	53	49	54	45	53	56	42	52	42	47	42	52	54	46	42	47	49	46	55
AG	60	44	51	50	49	54	47	42	53	51	44	50	51	51	45	46	51	49	53	13	48	45	52	49	45	45	53	47	50
OPER	60	55	55	49	53	56	52	57	45	44	55	50	43	53	51	57	54	49	42	51	51	53	16	57	56	46	42	49	57
MISC	60	49	53	50	55	16	57	48	45	50	47	42	49	53	57	54	47	44	42	55	51	51	52	44	44	43	48	47	44
NVG	45	36	33	28	40	37	27	36	36	28	7	40	38	42	35	33	35	30	33	41	41	42	34	38	33	8	37	28	33
AAR	180	173	177	172	162	174	170	166	167	165	177	173	168	170	163	168	171	100	170	174	166	164	173	172	176	167	162	173	170

The code is run for four consecutive weeks and the data is collected at the end of each week. At the end of the first week, remaining days to the currency limits are checked again. Table 10 shows the situation after scheduling the pilots for the first week. Of the 7 pilots who are close to their currency limit for certain missions at the beginning of the week, no one remains close their currency limit at the end of the first week.

Table 10. Remaining Days To The Currency Limits at the End of the 1st Week

MISSIONS	CURRENCY	PILOTS																											
	LIMIT	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28
IF	60	38	57	42	51	52	56	51	53	49	53	43	52	48	44	44	46	44	48	49	46	47	57	49	49	53	45	42	44
INT	60	56	50	56	56	60	56	56	51	43	46	58	60	56	56	59	60	59	60	56	59	60	49	45	56	59	58	41	41
BFM	60	43	53	41	58	43	39	48	44	39	56	44	52	43	39	60	58	59	59	51	58	52	56	57	46	60	49	53	50
ACM	60	38	51	42	52	49	38	51	56	53	52	48	56	38	58	58	47	46	41	46	52	48	51	56	46	58	49	47	52
ACT	60	50	56	48	50	56	45	50	51	57	49	57	50	58	56	52	58	56	56	57	38	56	50	58	38	43	45	58	57
AG	60	40	47	46	45	50	43	38	49	47	57	46	47	47	41	56	59	58	49	59	57	41	48	45	58	41	49	56	46
OPER	60	51	51	59	60	52	48	59	58	58	51	46	39	57	47	53	59	60	57	59	47	58	60	53	52	42	59	45	58
MISC	60	45	49	46	51	57	53	44	41	46	43	38	45	49	53	50	43	57	38	51	47	57	48	40	57	39	44	43	40
NVG	45	44	44	42	36	33	44	32	32	42	44	36	34	38	31	44	44	42	29	37	37	38	44	42	44	44	42	42	44
AAR	180	169	173	177	158	180	166	162	163	161	173	169	180	166	159	177	180	177	180	170	162	180	169	177	172	163	158	177	166

Table 11. Remaining Days To The Currency Limits at the End of the 2nd Week

MISSIONS	CURRENCY	PILOTS																											
	LIMIT	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28
IF	60	31	50	57	44	56	49	44	46	42	46	36	45	41	37	37	39	37	41	42	39	40	50	42	42	46	38	57	37
INT	60	60	56	56	56	53	56	60	60	36	59	56	53	59	56	52	53	56	53	60	59	53	59	56	49	58	60	34	34
BFM	60	36	58	34	51	36	57	56	59	60	49	37	45	36	58	56	51	52	52	59	57	45	49	50	60	53	58	46	56
ACM	60	31	44	58	45	42	58	56	49	46	45	56	49	31	51	51	40	39	34	39	45	41	44	49	39	51	56	58	45
ACT	60	56	49	41	57	49	58	43	58	50	42	50	57	51	49	45	56	49	56	50	31	58	43	58	31	57	38	51	57
AG	60	33	40	39	38	43	36	31	42	40	56	39	40	40	34	59	58	51	58	57	50	34	41	38	59	34	42	56	39
OPER	60	44	59	60	53	59	41	52	51	58	44	58	32	50	40	59	52	60	50	52	59	60	58	46	58	35	52	38	59
MISC	60	38	42	39	44	50	46	37	34	39	57	57	38	42	46	43	36	50	57	44	40	57	41	33	50	32	37	36	33
NVG	45	37	44	35	44	42	37	44	42	42	37	44	27	44	44	37	37	35	22	30	42	44	37	42	42	37	44	44	44
AAR	180	180	166	170	151	177	177	180	180	154	166	162	173	177	152	170	173	170	173	180	177	173	162	177	165	156	180	170	159

At the end of the second week, all the pilots who are close their currency limit for certain missions at the beginning of the first week are clear of their currency limits. Pilot 18 starts to come close to the currency limit of the NVG mission at the end of the second week but the code holds the pilot away from the currency limits as long as the flights in the schedule permit.

The results of this scenario show that the code tries to keep all pilots within their currency limits while distributing the flights and other duties as evenly as possible. Table 12 shows the number of sorties and other duties for four weeks. In this scenario, similar to the other scenarios, an average of 14 sorties per pilot and a standard deviation of 0.897 are obtained.

An unexpected situation is faced when comparing the results of the three scenarios. Pilot 15, Pilot 16 and Pilot17 have larger number of sorties compared to other pilots no matter which scenario it is. The reason for this is that these three pilots are the only 2LDs of the squadron, and since in the flight schedule there are more spots for 2LDs than there are for 4LDs or Ws, the algorithm assigns these pilots more than the others for the first week. However, as these pilots' number of sorties increase rapidly, in the following weeks the code balances the number of sorties properly.

Table 12. Number of Sorties and Other Duties

III	1st week				2nd Week				3rd Week				4th Week				TOTAL			
PILOT	NOS	SIM	RSU	SOF	NOS	SIM	RSU	SOF	NOS	SIM	RSU	SOF	NOS	SIM	RSU	SOF	NOS	SIM	RSU	SOF
1	3	0	0	1	4	0	0	0	4	0	0	0	4	0	0	1	15	0	0	2
2	4	0	0	1	2	0	0	0	6	0	0	0	3	0	0	0	15	0	0	1
3	4	0	0	0	4	0	0	1	3	0	0	0	3	0	0	0	14	0	0	1
4	3	0	0	1	4	0	0	0	3	0	0	0	4	0	0	0	14	0	0	1
5	5	0	0	0	2	0	0	1	3	0	0	1	3	0	0	0	13	0	0	2
6	3	0	0	1	4	0	0	0	3	0	0	0	4	0	0	2	14	0	0	3
7	3	0	0	1	4	0	0	0	4	0	0	1	2	0	0	0	13	0	0	2
8	2	0	0	0	5	0	0	0	2	0	0	1	4	0	0	1	13	0	0	2
9	4	0	0	0	2	0	0	0	4	0	0	2	3	0	0	0	13	0	0	2
10	2	0	0	0	3	0	0	1	4	0	0	0	5	0	0	0	14	0	0	1
11	2	0	0	1	4	0	0	0	3	0	0	0	4	0	0	0	13	0	0	1
12	1	0	0	0	5	0	0	2	5	0	0	0	3	0	0	0	14	0	0	2
13	3	0	0	0	3	0	0	1	5	0	0	0	4	0	0	1	15	0	0	2
14	2	0	0	0	3	0	0	0	5	0	0	1	3	0	0	1	13	0	0	2
15	6	0	0	0	4	0	0	0	1	0	1	0	4	0	0	0	15	0	1	0
16	7	0	0	0	3	0	0	0	2	0	2	0	4	0	0	0	16	0	2	0
17	6	0	0	0	4	0	0	0	3	0	1	0	3	0	0	0	16	0	1	0
18	3	0	1	0	5	1	0	0	4	0	0	0	2	0	1	0	14	1	2	0
19	4	0	0	0	4	0	0	0	3	1	1	0	4	0	0	0	15	1	1	0
20	4	0	1	0	2	1	0	0	4	0	0	0	5	0	1	0	15	1	2	0
21	5	0	0	0	2	0	1	0	5	0	0	0	3	0	0	0	15	0	1	0
22	4	0	0	0	3	0	1	0	4	0	0	0	4	0	0	0	15	0	1	0
23	3	0	1	0	3	0	0	0	4	0	0	0	4	1	0	0	14	1	1	0
24	3	0	1	0	5	0	0	0	4	0	0	0	2	1	1	0	14	1	2	0
25	5	0	0	0	3	0	1	0	4	0	0	0	3	0	0	0	15	0	1	0
26	3	0	1	0	4	0	0	0	2	1	0	0	5	0	1	0	14	1	2	0
27	3	1	0	0	4	0	1	0	3	0	0	0	5	0	1	0	15	1	2	0
28	3	1	0	0	5	0	1	0	3	0	0	0	3	0	0	0	14	1	1	0
															AVRG :		14.29	0.727	1.429	1.71
															STDV :		0.897	0.467	0.514	0.61

4.4 Conclusion

In this chapter, the algorithm is tested under two main instances. The first instance is the weekly flight schedule provided by one of the Turkish Air Force fighter squadrons and the second one is an arbitrarily built weekly flight schedule. For both instances and for all the tests, GRASP reacts well and creates feasible pilot schedules while providing an equivalent number of sorties for each pilot and keeping each pilot within the currency

limits as much as possible given the limiting factors of each instance. Every one of the four weekly schedules is created in less than 25 seconds (24.1 on average).

The next chapter discusses the results of the research and provides recommendations for future research.

5. CONCLUSIONS AND RECOMMENDATIONS

5.1 Summary of the Research

In the first chapter of this research, the problem and the objective of the research are stated along with the scope of the research. The necessary assumptions are stated at the beginning of the research. The specifications of the research problem are also introduced in the first chapter.

Existing research on different types of flight scheduling problems are studied in Chapter 2. The research problem is compared with other flight scheduling problems. General heuristic implementations and the use of GRASP in scheduling problems are examined.

In Chapter 3, the methodology and its implementation are introduced. The GRASP process including the grading steps and the inputs to the problem are explained. Mission types and their specifications are examined. A pseudo code is presented for the general flow of the model.

An analysis phase is presented in Chapter 4. The methodology is analyzed with the help of the Turkish Air Force 192nd Fighter Squadron's weekly schedules and a generated schedule. Different sub-scenarios are analyzed through these schedules.

In this chapter, the conclusions based on the results found in the analysis phase are explained. Also, recommendations for future research on the same or similar problems is introduced.

5.2 Conclusions

At the beginning of this research, the defined objective was to build an automated tool that creates feasible weekly pilot schedules for Turkish Air Force F-16 squadrons in a short amount of time and that provide equally distributed flights and other duties for the pilots while keeping the pilots within the currency limits of the missions.

In the analysis phase, the real life schedules show that the squadron scheduler can create an infeasible weekly schedule and the squadron scheduler must spend a huge amount of time to fix these infeasibilities. Even with the infeasible weekly schedules, assigning each pilot to an equivalent number of flights in a month cannot be accomplished. During the time period of the real life schedules, there are a considerable number of pilots coming from outside the squadron for flights which means a large number of manual inputs. Using the construction phase of the GRASP algorithm, the code decreases the standard deviation of the number of flights per pilot from 4.5 to 2.6 while keeping the pilots in their currency limits. This shows the success of the algorithm.

In the scenarios with the arbitrarily created weekly schedule, it is seen that the standard deviations of number of flight and number of duties per pilot remain minor. In the third scenario, the code keeps all pilots within the currency limits of the missions by assigning the pilots who are closer to the currency limit of a mission to a flight of this mission type. All of this is accomplished with an average of 24.1 seconds per weekly schedule which provides the squadron scheduler a huge amount of time savings compared to the 10 hours spent per weekly schedule. Even though the schedule created by the code may not be the perfect schedule for the scheduler, it represents an initial feasible solution and the scheduler can make desired changes to the schedule by

interchanging the pilots, changing the takeoff and landing times of the missions or the type of the missions. In fact, the squadron scheduler can make many of these changes and rerun the algorithm.

The analysis shows that the code can be used for creating weekly flight schedule and satisfies both objectives. Also, by putting all other days' flight in the schedule manually, feasible daily schedules can be created in a short amount of time.

In the second part of the analysis, it is seen that in the first weeks, 2LDs can be assigned to more flights than the other pilots. Even though the code balances the number of flights in the following weeks, this situation may not be acceptable to the scheduler. In order to prevent this, the status multiplier for flying one status below the pilot's status can be increased for the flight schedule with not enough flights for some status or for the squadrons that have an unbalanced number of leaders (4LD and 2LD).

5.3 Future Recommendations

For future research on the same problem or similar problems, a comprehensive Design of Experiment (DOE) study can be done for defining proper numbers for status and category multipliers, alpha value or any constant number that is used in the calculation of grades.

In this research, only the construction phase of GRASP is used. An improved model using the local search by interchanging pilots in the schedule that is created by the first phase of GRASP can be created. Also, the algorithm might be made faster and a huge number of iterations could be run in short time periods.

An advanced model with a user friendly interface can be created. This interface can let the scheduler define the limits or the restrictions of the weekly schedule such as

allowing shorter briefing or debriefing times. To input the required data can become easier with this kind of user friendly interface.

Using GRASP, weekly flight schedules can be created and pilots can be assigned to the flights in these schedules. Thus, time savings for schedulers increase.

APPENDIX A: BLUE DART

Optimizing an F-16 Squadron Weekly Pilot Schedule for the Turkish Air Force

Many of the countries in the world spend billions of dollars to have a powerful Air Force, since there may always be an attack to the national territory and these attacks may have to be responded instantly. Air Forces have great capabilities to have immediate responses to possible attacks such as precision and speed thanks to cutting edge technology in the aviation industry. However, technology is not the only factor to make the Air Forces attractive in national defense because the technology must be used effectively by pilots. Therefore, flight training activities occur every day to develop new tactics, to train new pilots, and to be ready for all possible missions.

However, Air Forces have limited budgets. The most effective flight training activities with the least amount of money spent is always desired. More importantly, Air Forces have also physical restrictions since the pilots are humans and they should have enough rest before performing flight missions. There are also some necessary flight hours for pilots to be able to continue flying without having additional training. Therefore, an effective flight schedule which saves resources in terms of flight hours and fuel, gives every pilot enough time to rest, and decreases the number and hours of additional training is desired.

Flight schedules are prepared by a flight scheduler in the squadron, which is a unit in the Air Force that every pilot lives in. First, the flight scheduler assigns the operational missions to the pilots coming from the headquarters. The operational missions are the most important missions while preparing a flight schedule. No matter how close a pilot to need to take an additional training, the scheduler should assign the best pilots to the

operational missions. Next, the required number of missions for each type and the pilots are matched. The most important point in preparing a flight schedule is that few only a few of pilots will need o take an additional training and the number of missions per pilots should be approximately similar. In other words, as many pilots as possible should fly their associated required hours of flights for each type; and the workload on pilots should be roughly the same.

This is not easy task as long as the scheduler has no automated tool to maintain at least an initial flight schedule since every pilot affects the other and preparing a schedule by hand requires hours. However, an automated tool, which provides a good initial schedule, helps the scheduler to have an optimum flight schedule in a short time (i.e., 2 minutes).

The model developed in this research performs the similar job as the scheduler prepares a flight schedule by hand. The model chooses best pilot-mission matches first. Then, it tries to find the best schedule in such a way that the additional training hours are minimized and every pilot has approximately the same workload. The scheduler uses an Excel spreadsheet inputting the required data such as the necessary missions to prepare a flight schedule. The Excel spreadsheet provides the best environment for the scheduler to prepare a schedule because the scheduler also uses the same tabular environment in preparing a schedule by hand. The only thing that the scheduler should do after inputting the required data is to hit the button. Then, an initial flight schedule is provided in less than a minute. The scheduler can play with the initial schedule to maintain the final schedule based on the instructions coming from the headquarters.

The initial solutions produced by the model, which is developed in this research, outperform the final schedules prepared by hand. For all the scenarios included in this research, the model creates schedules that provide equitable number of sortie for each pilot which means evenly distributed workload for the pilots.

APPENDIX B: QUAD CHART

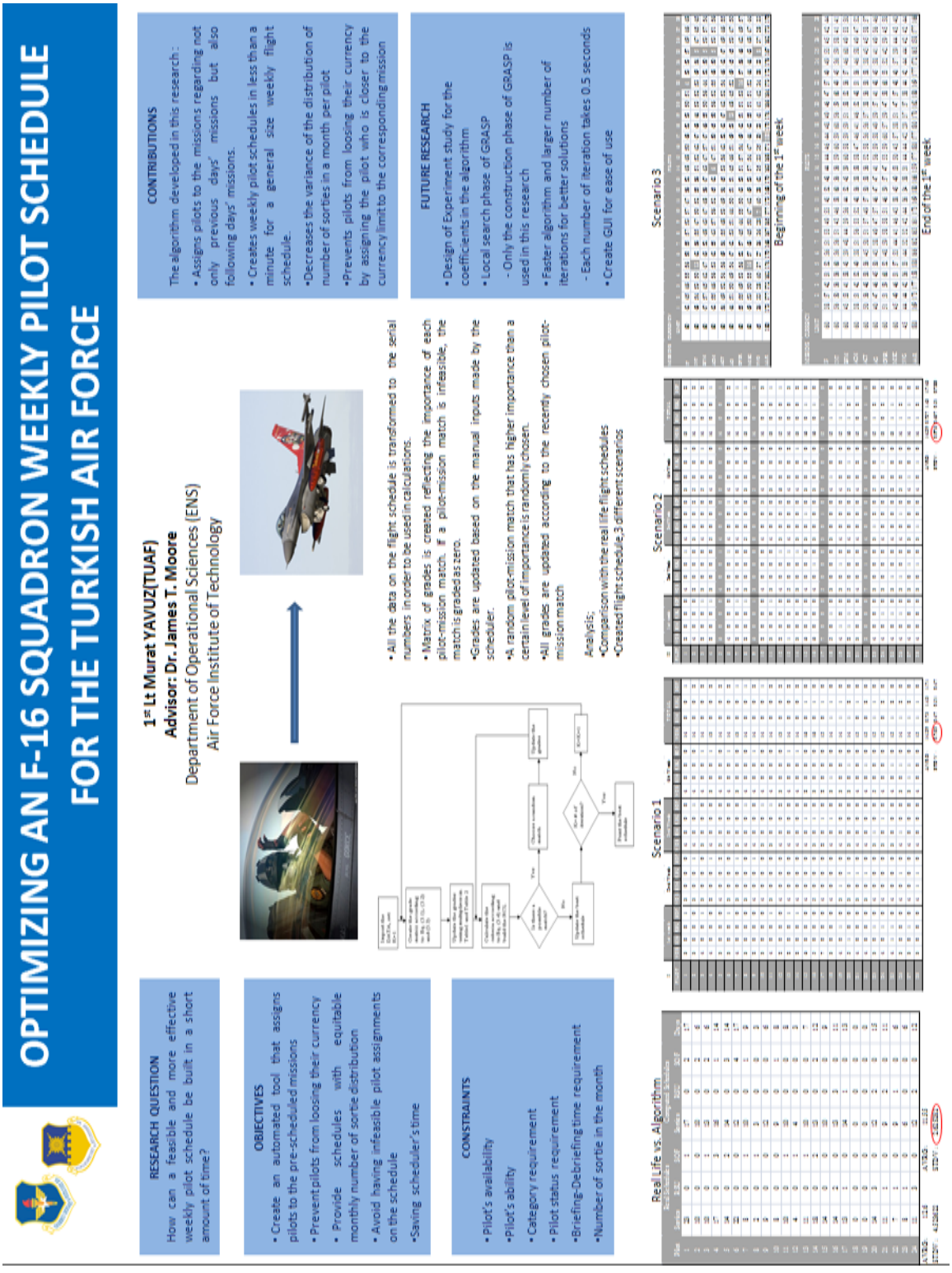


Figure 10. Story Board

BIBLIOGRAPHY

1. Aickelin Uwe, White Paul, "Building Better Nurse Scheduling Algorithms", *Annals of Operation Research*, 128 pp 159-177, 2004.

2. Aickelin Uwe, Dowsland Kathryn A., " An Indirect Genetic Algorithm for a Nurse Scheduling Problem", *Computer & Operations Research*, 31(5) pp 761-778, 2004

3. Aslan, Davut, "A Decision Support System for Effective Scheduling in an F-16 Pilot Training Squadron", MS thesis, AFIT/GOR/ENS 03-01. Graduate School of Management and Engineering, Air Force Institute of Technology (AU), Wright-Patterson AFB OH, March 2003.

4. Boyd, Jay A. H., Case A. Cunningham, Darren P. Gray, and John L. Parker, "Network Flow Model For Optimizing Fighter Squadron Scheduling", Graduate Research Project, AFIT/ENV ISL-06J. Graduate School of Management and Engineering, Air Force Institute of Technology (AU), Wright-Patterson AFB OH, June 2006.

5. Combs, Todd E., Moore, James T. "A Hybrid Tabu Search/Set Partitioning Approach to Tanker Crew Scheduling", *Military Operations Research*, Volume 9 Number 1, 43-56, 2004.

6. Dowsland Kathryn A. "Nurse Scheduling with Tabu Search and Strategic Oscillation", *European Journal of Operational Research*, 106 (2-3) 393-407, 1998.

7. Gokcen, Osman B., “Robust Aircraft Squadron Scheduling in the Face of Absenteeism”, MS thesis, AFIT/GOR/ENS 08-06. Graduate School of Management and Engineering, Air Force Institute of Technology (AU), Wright-Patterson AFB OH, March 2008.

8. HKY 164-1B *Muharip Jet ve Muharip Jet Eğitim Birlikleri Yönergesi*, TUAF, 2007.

9. Nawaz, M., Enscore, E.E., and Ham, I.,”A Heuristic Algorithm for the m-Machine, n-Job Flow-shop Sequencing Problem”, *Omega, The International Journal of Management Science,* 11(1), pp 91-95, 1983.

10. Newlon, Thomas M., ”Mathematical Programming Model for Fighter Training Squadron Pilot Scheduling”, MS Thesis, AFIT/GOR/ENS 07-17. Graduate School of Engineering and Management, Air Force Institute of Technology (AU), Wright-Patterson AFB, OH., March 2007.

11. Nguyen, Cuong T. “An Interactive Decision Support System for Scheduling Fighter Pilot Training”, MS thesis, AFIT/GOR/ENS/02-12. Graduate School of Management and Engineering, Air Force Institute of Technology (AU), Wright-Patterson AFB OH, March 2002.

12. Pinedo, Michael. *Scheduling: Theory, Algorithms, and Systems*, 3rd Edition, 2008

13. Prabharan G., Shaul Hamid Khan B., Rakesh L., “Implementation of GRASP in Flow Shop Scheduling”, *Springer-Verlag London Limited*, 2005.

14. Resende M.G.C., Ribeiro C.C., “Greedy Randomized Adaptive Search Procedures”, Handbook of Metaheuristics, pp-219-249, 2003.

15. Rojanasoonthon Siwate, Bard Jonathan, “A GARSP for Parallel Machine Scheduling with Time Windows”, *INFORMS Journal on Computing*, Vol. 17 No.1, pp 32-51 2005.

16. Stojkovic Mirela, Soumis Francois. “An Optimization Model for the Simultaneous Operational Flight and Pilot Scheduling Problem”, *Management Science*, Vol 47, No.9, p. 1290-1305, September 2001.

Vita

First Lieutenant Murat YAVUZ was born in İzmir. He graduated from Air Force Academy in İstanbul, in 2002 and he earned the degree of Bachelor of Science in Electronic Engineering. In the same year, he began his flight training in the 2nd Main Jet Base in İzmir. In 2005, after graduating from F-16 Basic Training Program, he was assigned to the 192nd Squadron, Balıkesir as a wingman. He entered Graduation School of Engineering and Management, Air Force Institute of Technology in 2008.

REPORT DOCUMENTATION PAGE	Form Approved OMB No. 074-0188

1. REPORT DATE *(DD-MM-YYYY)*	**2. REPORT TYPE**	**3. DATES COVERED** *(From – To)*
25-03-2010	**Graduate Research Project**	Sep 2008 – Mar 2010

4. TITLE AND SUBTITLE

OPTIMIZING AN F-16 SQUADRON WEEKLY PILOT SCHEDULE FOR THE TURKISH AIR FORCE

5a. CONTRACT NUMBER

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)

Murat YAVUZ, 1st Lt, TUAF

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAMES(S) AND ADDRESS(S)

Air Force Institute of Technology
Graduate School of Engineering and Management (AFIT/EN)
2950 Hobson Street, Building 642
WPAFB OH 45433-7765

8. PERFORMING ORGANIZATION REPORT NUMBER

AFIT-OR-MS-ENS-10-11

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

INTENTIONNALY LEFT BLANK

10. SPONSOR/MONITOR'S ACRONYM(S)

11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT

APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.

13. SUPPLEMENTARY NOTES

14. ABSTRACT

Fighter squadrons in the Turkish Air Force build flight schedules for weekly periods. This process requires a great deal of time and does not seek optimality. Schedules are built with feasibility concerns. The Turkish Air Force doesn't have an automated tool for flight scheduling. Many constraints including crew rest, number of sorties flown in a month, and duty currencies affect the schedule. Providing an automated scheduling tool may help schedulers save time for other squadron tasks including mission preparation, briefing, and debriefing. In this research, a heuristic approach to the problem is developed. Greedy Randomized Adaptive Search Procedures (GRASP) is applied to the weekly pilot scheduling problem. Manual scheduler inputs are allowed. A code for GRASP implementation is written in MATLAB.

Two different approaches are used in the analysis. First, the code is run for four weekly schedules taken from an F-16 squadron of the Turkish Air Force and second, a weekly flight schedule is created randomly. In the second approach, the created flight schedule is used for three different scenarios which represent possible real life situations.

For all scenarios and real schedules, GRASP performed well and smaller standard deviations in sortie numbers are obtained while keeping all pilots within the currency limit of each mission.

15. SUBJECT TERMS

Pilot Scheduling, Heuristic in Scheduling, Greedy Randomized Adaptive Search Procedure (GRASP)

16. SECURITY CLASSIFICATION OF:			**17. LIMITATION OF ABSTRACT**	**18. NUMBER OF PAGES**	**19a. NAME OF RESPONSIBLE PERSON**
a. REPORT	**b. ABSTRACT**	**c. THIS PAGE**			James T. Moore, Dr. (ENS)
U	**U**	**U**	**UU**	85	**19b. TELEPHONE NUMBER** *(Include area code)* 937) 255-3636 ext:4528