ABSTRACT

Title of Document:	A GENETIC ALGORITHM-BASED COLUMN GENERATION APPROACH TO THE PASSENGER RAIL CREW SCHEDULING PROBLEM
	Mindy Wang Liu, Master of Science, 2009
Directed By:	Professor Ali Haghani Department of Civil & Environmental Engineering

The goal of this thesis was to develop and apply a genetic algorithm-based column generation heuristic to solve a passenger rail crew scheduling problem. The crew scheduling problem minimized the total cost of payment to crew members based on the hours on-board, hours away from a crew base, number of nights of lodging, and number of on-board and away meals. Payment regulations also dictated an overtime payment and a guaranteed salary per week. Additional problem constraints included restrictions on the maximum number of continuous working hours, maximum number of days worked per week, and minimum hours of rest. The proposed heuristic produced solutions with improvements of total cost ranging from 3.0 percent to 27.9 percent.

GENETIC ALGORITHM-BASED COLUMN GENERATION APPROACH TO THE PASSENGER RAIL CREW SCHEDULINGN PROBLEM

By

Mindy Wang Liu

Thesis submitted to the Faculty of the Graduate School of the
University of Maryland, College Park, in partial fulfillment
of the requirements for the degree of
Master of Science
2009

Advisory Committee:
Professor Ali Haghani, Chair
Professor Paul Schonfeld
Professor Cinzia Cirillo

Dedication

For my parents.

Acknowledgements

My sincerest gratitude goes to my advisor, Dr. Ali Haghani, for his guidance and encouragement, and to my defense committee members, Dr. Paul Schonfeld and Dr. Cinzia Cirillo.

Thank you to the research group and other colleagues for valuable advice. Thank you also to Shahab Toobaie, who introduced me rail crew scheduling problems and assisted in the data collection for this work.

I am very grateful to the Civil & Environmental Engineering faculty, those whom inspired me and helped develop my interest in transportation and engineering.

Finally, thank you to Eric Yeh. This thesis would not have been possible without you.

Table of Contents

List of Tables

List of Figures

Chapter 1: Introduction

1.1 Background

The crew scheduling problem (CSP) is the problem of assigning work to crew members to create a minimal cost work schedule. In this thesis, a genetic algorithm-based column generation heuristic is applied to the CSP for the Northeast Corridor of the North American passenger railway corporation, Amtrak. Amtrak's busiest corridor is the Northeast Corridor, which stretches from the Canadian border to North Carolina. Within the corridor, the most heavily used portion is between Boston, Massachusetts and Washington, D.C., which includes stations in New York City and Philadelphia, Pennsylvania. Amtrak crew is comprised of trainpersons and engineers (T&E crew) and on-board services crew. For this thesis, the CSP focuses on solely the T&E crew, consisting of conductors, assistant conductors, engineers, and firemen. The CSP for the Northeast Corridor contains 4338 total work duties that must be covered by these crew members.

Most of the research on rail crew scheduling in the North America has focused on the freight rail industry rather than passenger rail. Although the lessons learned from research on CSPs in either industry may be applicable to other industries, there are also important differences between freight and passenger rail crew scheduling.

First, the North American passenger rail industry has special payment rules with guaranteed weekly or biweekly salaries. The union regulations generally require that workers be paid a minimum weekly or biweekly salary regardless of the number of actual hours worked. A simple example is a crewperson assigned 10 hours of work in a week but guaranteed pay for 40 hours per week. This guaranteed salary results in inefficiency since crew are essentially paid for not working. While these schedules may be necessary for some crew members in order to cover all work requirements, it is likely that current crew schedules are not operating at maximized efficiency due to this regulation. This special guaranteed payment rule is unique to Amtrak and is not found in many other industries.

Second, the passenger rail crew payment is based on shifts rather than individual duties. In this respect, the payment scheme is similar to many long-haul airline crew payment rules in which crew members are paid based on the time away from their home rather than individual duties. These two special payment rules are not typically found in the freight rail industry and significantly complicate the crew scheduling problem. For Amtrak's Northeast Corridor in particular, the single train trips may be relatively short compared to train trips in other corridors in the Midwestern and Western portions of the country. These shorter trips may result in more time away from a crew member's crew base since more trips are required to fill a single crew member's weekly or biweekly schedule.

A third difference between freight and passenger rail crew scheduling is that passenger rail crew normally have set work duties. These work duties are typically defined by the train timetable whereby a certain number of crew must work every train trip listed in the timetable. However, in freight rail, work duties may change based on shipping demand. Since shipping demand frequently changes based on many different factors, the trips are often not repeated weekly, biweekly, or even monthly as they are in transit and passenger rail. The scheduling cycles in passenger rail applications mean that schedules can be optimized for a single cycle, for instance, a weekly schedule, and that any savings in cost can be multiplied to produce large annual savings. This nature of passenger rail makes these CSPs good candidates for optimization and improvements. Nevertheless, as previously stated, most research on CSPs in North America has focused on the freight rail industry. The differences between freight and passenger rail crew scheduling highlight the need for more research on North American passenger rail CSPs.

On an international level, both passenger and freight rail crew scheduling have been more widely studied than in North America. Studies on passenger rail crew scheduling in the last decade include applications at Netherlands Railways (Kroon & Fischetti 2001), Australian National Rail Corporation (Ernst 2001), and Deutsche Bahn in Germany (Bengtsson et al. 2007). In each study, crew scheduling at these passenger rail corporations produced significant savings in operating costs. These applications show that there is potential for improving crew schedules at Amtrak as well. While the solution approaches used in these studies provided effective crew scheduling systems, tailoring these solution methods to the Amtrak CSP would prove to be a difficult task. The numerous differences between working and payment regulations across various passenger rail corporations facilitate the need for case-specific solution approaches.

Outside of the rail industry, the CSP has been applied in the air and transit industries. In fact, CSPs have been most extensively studied in the airline industry. While the underlying goals of the air and transit CSPs are similar to those in rail, the major difference here is in the work duties. For airlines, the work duties might consist of all flights in a hub network, whereas in rail, the work may include a much larger set of train trips. The train trips may also occur much more frequently than flights. A second difference between these CSPs is air crew may be allowed to work continuously for a longer period of time to allow for longer distance, or long-haul, flights. For transit applications, the work may be daily bus or subway trips in a far smaller network than rail or air. The transit work regulations may require that crew have resting periods more frequently, but the time horizon of a crew schedule is also typically shorter. The transit CSP also does not need to consider crew away time lodging and meals, and all crew must return home at the end of each day. In these respects, the rail CSP can prove to be a more difficult problem to solve than many airline and transit CSPs.

1.2 Problem Statement

The purpose of the work in this thesis is to apply a genetic algorithm-based column generation heuristic to solve the passenger rail crew scheduling problem for Amtrak's Northeast Corridor. The crew in this problem consists of trainpersons and engineers who are responsible for piloting trains and handling passengers. More specifically, this crew includes conductors, assistant conductors, engineers, and firemen based in the Northeast Corridor region in the railway system.

The objective of the problem is to minimize the cost of salary payment to the crew based on the number of hours worked on-board a train, the number of hours held away from the crew base when not on-board, the number of nights of lodging provided, and the number of on-board and away meals provided. The constraints of this problem are the work regulations, as defined in the union rules, and the payment regulations defined by Amtrak. These constraints set minimum rest times, maximum working days per week, overtime rates, and guaranteed salary. The ultimate goal of this thesis is to determine if there is potential for savings in operating costs for Amtrak by improving crew schedules.

1.3 Motivation

The primary motivation of this thesis is to determine if there are potential cost-savings in optimizing Amtrak crew schedules. In 2008, Amtrak has seen increased ridership due to various factors, including the rise of fuel prices and the resulting attractiveness of mass transit. However, for many years, Amtrak has also faced criticism for not generating enough revenue to support operations and depending heavily on government subsidies and funding. The need for an efficient business drives the motivation for researching cost-savings in improving crew scheduling. Despite this need, the passenger rail crew scheduling problem has not been extensively studied in the United States.

The complexity of crew scheduling constraints also makes the Amtrak problem more difficult to solve than many other CSPs. This thesis aims to develop an approach to solve this passenger rail crew scheduling problem for Amtrak's Northeast Corridor within a reasonable amount of time and computer memory resources. Due to the large size of CSPs, finding optimal solutions is often impossible or impractical. As a result, finding an improved solution within a short amount of time is an important component of a CSP solution method.

While Amtrak has optimized crew schedules for the Northeast Corridor in the past, it is uncertain if there is room for improvement in the current schedule. The ultimate goal of this work is to show if there are any potential cost-savings in optimizing the Northeast Corridor crew scheduling.

The existing schedule implemented by Amtrak covers a total of 43 routes and 469 trains. Of these routes, 15 are long distance, and 28 are corridors and state sponsored routes. For the month of April 2007, 2,207,146 passengers rode on Amtrak trains. The revenue collected from this particular month reached $128,644,681. The financial information for that entire fiscal year reported 25.8 million passengers and $1.5 billion in revenue. Because of the repeating nature of crew scheduling, minor improvements to schedules which reduce cost each week or month can lead to a significant annual savings. This makes crew schedule improvement an ideal application for cost-savings studied in this thesis.

1.4 Overview of Research

The goal of this thesis was to develop and apply a genetic algorithm-based column generation heuristic to solve a 4,338 duty passenger rail crew scheduling problem in North America. A review of relevant research on passenger rail CSPs and CSPs in general revealed the need for further research on crew scheduling for rail crew schedule in the United States. Various techniques and solution methods have been applied to CSPs, including mathematical programming, heuristics, and constraint programming. For large, real world-size problems, a combination of these approaches has proven to be useful for rail crew scheduling applications.

Crew schedule and work requirement data for this thesis were obtained from Amtrak. The datasets were for the T&E crew in the Northeast Corridor. As part of this thesis, the data was processed and prepared for the crew scheduling application. In order to reduce the size of the problem, the entire network of trains was partitioned into smaller problems by crew type and by region. The partitioning resulted in four separate CSPs. In addition, three small problems were created based on the real datasets.

The main contribution of this thesis was the development and application of a genetic algorithm-based column generation heuristic on the passenger rail crew schedule problem. This heuristic was applied to the three small, generated problems, the four partitioned CSPs as well as two larger CSPs for the entire Northeast Corridor network. The new solutions for these problems were compared, and the sensitivity of the overall improvement, calculation time, and memory usage was examined. The heuristic was able to successfully show improvements in the crew schedules, which indicated that there is potential for savings in cost by improving the Northeast Corridor crew schedules.

1.5 Outline of Thesis

The thesis is divided into the following seven chapters:

Chapter 1 – Provides a brief introduction to the passenger rail CSP, Chapter 2 – Discusses important definitions and a literature review of past works on CSPs with emphasis on the main solution approaches applied to CSPs,

Chapter 3 – Describes Amtrak CSP and the data obtained from Amtrak,

Chapter 4 – Explains in detail the solution methodology and the heuristic approach,

Chapter 5 – Discusses analysis of the solution results, and

Chapter 6 – Summarizes the conclusions and offers extensions on the passenger rail CSP.

Chapter 2: Literature Review

2.1 Introduction

This chapter introduces the crew scheduling problem (CSP) and provides definitions of important terminology related to the CSP. A review of relevant literature organized by solution approach follows. For each solution approach, important works in the rail, air, and transit industries are described.

2.2 The Crew Scheduling Problem

The CSP is the problem of assigning work duties to crew members while covering all required work at minimum cost. Because crew scheduling terminology is not standard throughout the literature, it is first important to define the terminology used in this thesis:

- **Duty –** In rail crew scheduling, a *duty* corresponds to a single train trip or one day of yard work or extra board. For example, a single duty may be an engineer duty for the train trip from Washington, D.C. to New York City at 9 AM on Monday.
- **Yard Work –** Crew on *yard work* at the crew base station is responsible for maintenance or other work at the train station. Yard work duties do not

require traveling away from the station, thus, do not require away time, lodging, or meals. Typically, certain crew members are responsible for yard work, and most of these crew members' schedules consist of only yard work.

- **Extra Board –** *Extra board* duties are when crew members are on back-up duty for other crew. These crew members may remain at home and are not on-duty unless unforeseen circumstances force them to be called to work. For instance, if a crew member takes leave due to illness, a crew member on extra board will fulfill the duties. Crew members on extra board are paid the regular hourly salary regardless of whether he or she is called to work.
- **Shift –** One or many duties make up a *shift*. A shift is a chain of duties that begins and ends at a crew base. For example, a single shift may be from the crew base in New York City to Philadelphia, from Philadelphia to Washington, D.C., and from Washington, D.C. back to New York City.
- **Crewbase –** The *crewbase* is the location at which a crew member goes on and off duty and is usually near the crew member's residence.
- **Pairing -** One or many shifts make up a *pairing*, or a sequence of trips that starts and ends at a single crew base for a given time horizon. In the Amtrak CSP for this thesis, a pairing consists of all duties for a single crew member over a one week period. A pairing may consist of train duties, yard work, and/or extra board duty.
- **Away –** During a shift, a crew member may travel to an *away* location, a location that is not the crew base and in which a crew change is permitted. When at an away location, the crew may have rest time and may be

compensated for meals and/or lodging depending on the length of the away time.

- **Deadhead –** When a crew *deadheads* a duty, the crew is not on duty during that train trip. Deadheading may be required to transport a crew to a location for starting a shift or to transport a crew back to the crew base at the end of a shift.

The planning horizon of crew scheduling may vary. *Long-term* planning involves large changes to the crew schedule every few years. In the long-term planning, the crew scheduling has a large impact on the operations and business of the transportation system. In *short-term* planning, crew scheduling changes are marginal and may be done biannually or seasonally. Lastly, *tactical* planning involves crew scheduling responses to real-time changes in work demand or disruptions. Solving CSPs for tactical planning is a difficult task that requires collection of real-time information and powerful solution methods that can produce efficient results.

2.3 Early Works on the CSP

Early research in crew scheduling was primarily applied to the transit and air industries. In the transit industry, one of the earliest works in computerized crew scheduling involved using a simple local search approach to develop crew schedules for trains in Adelaide, Australia (Bennett & Potts 1968). Wren also provides a brief review of early works in transit and an application of a constructive heuristic for bus driver scheduling at London Transport (Wren 1981). In the airline industry, the CSP was solved by enumerating pairings, reducing the number of pairings by heuristics, and then optimizing a reduced linear program (LP) problem using the branch-and-bound method (Arabeyre, Fearnley, Steiger, & Teather 1969). A comprehensive summary of early CSP cases and solution methods is provided by Bodin et al. (1983). The focus on airline and transit applications is perhaps because these areas have larger opportunities for profit from solving crew scheduling problems.

2.4 Applications of the CSP

Although earlier works were not applied in the rail industry, there are similiarities between the CSP in these different applications. The rail CSP is most similar to the problem in the airline industry for several reasons. First, the time horizons are similar in that both cases involve short-haul and long-haul trips. The transit case differs because crew scheduling horizons are daily and crew members return home at the end of the day. With respect to scheduling time horizons, the air and rail CSP is more complex because crew may be required to stay at an away location. The CSP costs then involves not only costs associated with lodging and meals at away locations, but also far more complex rest requirements. Union and work regulations require that crew members have specific minumum rest times. In contrast to the airline application of the CSP, the rail application may be more difficult in the magnitude of the problems. In the CSP for the Italian railways, there were over 5000 train trips and one million duties per day (Caprara et al. 1999). The researchers noted that this magnitude is one to two times larger than a typical airline CSP.

Most works on the rail CSP has been conducted in Europe. The large body of research done in Europe may have been a response to the deregulation and privatization of the rail industry in European countries. In addition, there has been research in other areas that have larger markets for rail such as in Asia and Australia In the United States, a recent work solved the CSP for a major freight rail company There is more draw for work to be done in the freight industry over the passenger rail industry because there is more opportunity for profit from finding solutions for freight rail carriers. At this time, there has been little research on the rail passenger CSP in the United States.

2.5 Solution Approaches

2.5.1 Set Partitioning Problem & Set Covering Problem Approaches

The most common method of solving the CSP in the literature has been through modeling the CSP as a set covering problem (SCP) or set partitioning problem (SPP). In both of the SCP and SPP formulations of the CSP, the decision variable, x_j, is a binary integer variable that represents whether or not a pairing is selected as a work duty for one crew member. The constraints consist of a matrix of binary values a_{ij} that indicate if a pairing j covers a work duty i. Each row in this matrix shows which pairings cover a single work duty. Each column corresponds to one possible pairing or the work for an individual crew member over the defined time horizon. The SPP models the CSP as a problem of finding a minimum cost subset of pairings that exactly covers the work requirements. The SPP formulation is:

Minimize $$\sum_j c_j x_j$$

Subject to $$\sum_i a_{ij} x_j = 1$$

$$x_{ij} \in \{0,1\}$$

Where: $c_j =$ cost of pairing

$$x_j = \begin{cases} 1, & \text{pairing j is selected} \\ 0, & \text{otherwise} \end{cases}$$

$$a_{ij} = \begin{cases} 1, & \text{pairing j covers work requirement i} \\ 0, & \text{otherwise} \end{cases}$$

The SCP formulation of the CSP is similar to that of the SPP, except the problem is more flexible, allowing for over coverage of the work requirements. The SCP is the problem of finding the minimum cost subset of pairings that covers the work requirements. The SCP formulation of the CSP is:

Minimize $$\sum_j c_j x_j$$

Subject to $$\sum_i a_{ij} x_j \geq 1$$

$$x_{ij} \in \{0,1\}$$

Where: $c_j =$ cost of pairing

$$x_j = \begin{cases} 1, & \text{pairing j is selected} \\ 0, & \text{otherwise} \end{cases}$$

$$a_{ij} = \begin{cases} 1, & \text{pairing j covers work requirement i} \\ 0, & \text{otherwise} \end{cases}$$

Depending on the CSP application, the SPP or SCP formulation may be more appropriate. In some cases, exact coverage may be needed because over coverage would result in a large cost relative to the regular schedule costs. Formulating this problem as a SCP would then result in a suboptimal solution. In this thesis, the SPP is the appropriate formulation since exact coverage is needed. Over coverage would result in excess crew members assigned to duties.

Earlier works using the SCP or SPP approaches added side constraints and were application oriented. Rubin formulated an airline CSP as an SCP with constraints that specified the number of crew at each crew base (Rubin 1973). The researcher did not solve the SCP, but instead used a current known schedule as an initial solution for a heuristic to solve the problem. A review of other earlier works using the SPP model of the CSP can be found in Marsten and Shepardson (1981). This paper reviewed the use of Lagrangian relaxation, subgradient optimization, and heuristics to solve the CSP for several airlines and a transit system.

A major difficulty with the SPP and SCP formulations of the CSP is that of determining all possible pairings for these models. In the full SPP and SCP formulations, the matrix of constraints contains columns for every possible pairing. For larger CSPs, this results in an extremely large number of columns. For a CSP with thousands of trips, there can easily be an unmanagebly large amount of possible pairings. In this regard, the first problem is the time-consuming task of enumerating all these possible pairings. In the rail CSP, complex work and payment rules make it difficult and inefficient to determine the set of all possible and legal pairings. Then the second problem is that even if all pairings can be enumerated, the resulting SPP or SCP, which is a binary integer program, is NP-hard.

Much of the past work in CSPs has tackled the difficulty of solving the SPP and SCP formulations for the CSP. Some researchers have attempted to use constraints that force the linear programming (LP) relaxation of the problem to have integer solutions (Ryan and Foster 1981). In a later work, the researchers showed how an LP relaxation of the integer SPP produced integer or near-integer solutions when each duty must be followed by the next available work requirement.

Another approach used in the SPP and SCP formulations of the CSP is the use of constructive heuristics methods to reduce the problem size. Smith and Wren introduced heuristics for reducing the number of feasible pairings to a manageable size and solved the smaller problem with a branch-and-bound approach (1988). Another study introduced decomposing the CSP into subproblems by starting time (Falker and Ryan 1992). Then, heuristics are used to remove duties that are not likely to appear in good solutions. Finally, smaller sized SPPs were solved.

In another work, the researchers used constructive heuristics with relaxation techniques to solve the CSP. Caprara et al. (2001) divided the CSP for an Italian railway into three parts: pairing generation, pairing optimization, and roster optimization. For the pairing generation step, the researchers enumerated all feasible pairings and then proceeded to use heuristics to reduce the set of feasible pairings. In the pairing optimization step, the researchers used a Lagrangian relaxation technique to help solve the set covering problem. Finally, a constructive heuristic was used to match pairings together to create crew rosters. The final crew schedule solution is improved by iterating between the pairing optimization and roster optimization steps (Caprara et al. 2001).

For very large CSPs, many researchers have adopted a column generation technique to help in solving the SPP or SCP. Desrocher and Soumis (1989) presented a column generation constrained shortest path subproblem for adding new pairings. The researchers solved an urban mass transit CSP as a SCP with side constraints for work shift regulations using dynamic programming. The use of the shortest path algorithm has been used successfully in other works as well.Column generation is used with pricing strategies, which is a method called branch-and-price.

The simplest way to generate these new pairings is through enumeration (Garfunkel and Nemhauser 1970; Marsten 1974). However, enumeration is memory-intensive and not efficient for large problems. Methods that combine enumeration with bounding have also been used (Makri & Klabjan 2004).

Other research has approached the subproblem of pairing generation by using heuristics for generation promising but not necessarily optimal pairings (Klabjan et al. 2001). One specific heuristic method is constraint programming, which uses computer programming techniques to reduce the feasible domain of solutions to the subproblem (Fahle et al. 2002; Sellman 2004). Such heuristics provide fast approaches to finding near-optimal solutions to the subproblem.

Many works have formulated the pricing subproblem as a constrained shortest path problem (Desrochers & Soumis 1989; Lavoie et al.1988). In this approach, dynamic programming is usually used to solve a multi-label, resource-constrained shortest path problem. This approach, however, is not able to efficiently model more complex constraints.

Similar to branch-and-bound for integer programming problems, branch-and-price uses a branching tree to solve for integer solutions. In the branch-and-price method, column generation is used at each node in the branch-and-bound tree. Constraint branching rules are used to solve a pricing subproblem at each node in the branch-and-bound tree. Branch-and-price involves generating new pairings, or columns, while solving an IP, whereas the column generation heuristic method generates new pairings in between solving LP relaxations problems. A detailed description of branch-and-price for airline CSPs can be found in Barnhart et al. (1998). The branch-and-price method has also been used in mass transit (Fores and Proll 1998) and rail CSPs (Freling et al. 2004).

Another method for solving large, real-world CSPs formulated as a SCPs or SPPs is the branch-and-cut method. Branch-and-cut involves branch-and-bound with cutting planes to solve large integer programming problems. Hoffman and Padberg (1993) introduced the branch-and-cut method for solving a SPP in airline crew scheduling with 8,600 columns by 800 rows and one million columns by 145 rows. A similar study also applied the cutting plane method to solve the CSP for United Airlines (Graves et al. 1993). In the rail industry, Ernst et al. solved the crew scheduling and roster problems with complex regulations and crew quality of life restrictions for an Australian railway (2001).

Researchers have also extended on the general CSP. Willers et al. examined the CSP with dual objective functions for cost and number of crew (1995). The study presents new construction heuristics and a model for the dual objective function. Other works have integrated the CSP with vehicle scheduling. In the general CSP, the vehicle schedule has been determined and is used as an input into the CSP. However, researchers have worked on the integrated vehicle and crew scheduling problems as a way of better optimizing transportation operations. Haase and Friberg presented a SPP formulation of the integrated mass transit vehicle and crew scheduling problem with a single crew base. The problem was solved using branch-and-bound with column generation (1999). An extension the work incorporated a cutting plane scheme in the solution approach (Haase et al. 2001). Another way of dealing with the CSP and vehicle scheduling has been to examine crew scheduling before vehicle scheduling and routing. Klabjan et al. solved the airline CSP before the aircraft routing problem to evaluate a new way to find cost savings (2001). This work considered the CSP with time windows and number of aircraft constraints in order to ensure a feasible solution that precedes the aircraft routing problem.

Recent works in CSP have also focused on the crew rescheduling problem, or crew scheduling as part of disruption management. The goal crew rescheduling problem is to make changes to a predetermined scheduling while minimizing costs caused by disruptions such as weather, technical problems, or absence of crew. There especially has been interest in the crew rescheduling problem in the airline industry (Nissen and Haase 2006, Kohl et al. 2007). In the rail industry, researchers have examined the impacts of crew rescheduling, adjustments in timetables, and rolling stock in Denmark (Jespersen-Groth et al. 2007).

With recent advancements in technology and improved solution techniques, researchers have solved large real-world railway CSPs using the SPP and SCP methods. Kroon and Fischetti solved CSPs for a Dutch railway with 2,500 through

9,000 trips (2001) using a combination of column generation, Lagrangian relaxation, and heuristic methods. Another more recent study of the Dutch railway NS Reizigers presented solution techniques for a CSP with over 14,000 trips and 1,000 duties per day (Abbink et al. 2004). The researchers used similar methods to find a crew schedule solution with savings of $4.8 million per year.

2.5.2 Network Flow Approach

Another approach to the CSP is through modeling the problem as a network flow problem. Mellouli introduced a time-space network representation of a German rail CSP (2001). The advantage of the network approach is that the resulting problem is usually smaller than the corresponding SPP or SCP. In this approach, the difficulty lies in the construction of the network model. Yan and Tu used a pure network flow problem and network simplex to solve an airline CSP for China Airlines (2002). Another study introduced the airline CSP as an integer multicommodity flow problem (MFP) (Cappanera et al. 2004). The researchers develop a model that incorporates complex regulations on working times, crew absence, training, and union activities. The MFP approach was able to solve a CSP that could not be solved by a traditional SPP approach. Vaidyanathan et al. (2007) also used a MFP approach for a freight rail CSP. The researchers developed a time-space network model of the problem and also introduced an algorithm for incorporating the seniority and bidding aspect of the freight rail crew system.

2.5.3 Metaheuristic Approach

A different approach to the CSP is through the use of metaheuristics. Metaheuristics provide a framework of steps to solve optimization problems and can produce near-optimal or optimal solutions. These strategies typically allow worse solutions in subsequent iterations in hopes of avoiding local optima. One metaheuristic that has been applied to the CSP is simulated annealing (SA). In SA, the heuristic framework uses principles from the energy process of cooling metals to find new solutions. SA metaheuristics are generally good for finding acceptable solutions in a defined amount of time. Emden-Weinert and Proksch (1999) solved an airline CSP using a SA approach. The work concluded that the SA produced good quality solutions, but required longer processing times than simpler heuristics. An SA approach was also used to solve a multi-objective CSP for airline pilots (Lucic and Teodovoric 1999). In the rail industry, SA has been used to solve the CSP with complex crew compensation rules (Ernst et al. 2001). Overall, SA solution approaches to the CSPs have produced acceptable solutions, but have not been shown to be as effective as other methods.

Another metaheuristic applied to CSPs is tabu search (TS), which exploits local or neighborhood searches and a list of "taboo" moves to drive the solution search and avoid local minima. Cavique et al. (1999) used a TS approach for solving a CSP with regulations for working time and meal breaks for Lisbon Underground. TS has also been used in transit CSPs, but produced solutions that were inferior to other methods (Shen & Kwan 2001). The researchers, however, show that refining a TS procedure has potential to produce better solutions. Tabu search methods treat the CSPs as problems similar to vehicle routing problems (VRP), in which crew (or drivers in the VRP) must be on specific trains (or at specific locations in a network for the VRP) at the given time. Instead of routing vehicles in a network, the CSP “routes” crew members through a network of train trips. Tabu search methods have not been extensively studied for applications in the CSP. However, the success of tabu search metaheuristics for VRPs with various constraints indicates that they may be applicable to CSPs as well.

Genetic algorithms (GA) have also been used to solve CSPs. GAs use the evolutionary behavior of genes in chromosomes to produce new populations of possible solutions from existing parent solutions. GA approaches have been used in several transit CSP applications (Clement and Wren 1995; Wren & Wren, 1995; Kwan and Wren 1996; Kwan et al. 1999). The GA methods produced good quality solutions, but cannot guarantee optimal solutions. Another application of GA showed that the GA solutions were inferior to exact algorithms for most problem instances and did not produce feasible solutions for some problems (Levine 1996). The GA approach does, however, allow incorporating complex work regulations and can produce good solutions for very large problems that cannot be solved efficiently with exact solution methods.

2.6 Summary of Literature Review

The CSP aims to reduce inefficiencies in crew schedules for airline, rail and transit systems. By reducing these inefficiencies, transportation operators can significantly reduce operating costs. Additionally, CSPs solutions can help balance workload for crew members.

Past work on passenger rail CSPs is limited. The literature has focused on airline applications and rail applicccations in Europe and Asia. The rail CSP research in North America has especially been limited, which prompts the need for further work on passenger rail CSPs, such as the Amtrak CSP in this thesis.

The review of relevant literature presented three general approaches to solving the CSP:

1. **SPP & SCP Approach --** This approach is the most widely used and has been applied to the largest, real world sized problems. Due to the problem formlation in this approach, complex contraints can be incorporated. Since this approach formulates the CSPs as a binary IP and requires enumeration of a large set of feasible variables, the problem is NP-hard. As presented in this chapter, reseachers have used various relaxation methods, heurstics, column genration, and branching strategies to solve this problem. However, approaches that have been applied to real world problems are not ideal for solving the Amtrak CSP because each method is tailored to a specific problem and constraints. In particular, the Amtrak CSP has complex payment and allowance schemes as well as complex work regulations and union rules to which a schedule must adhere.
2. **Network Flow Approach –** The network flow approach has been used in more recent research on the CSP in the last decade. This approach formulates the CSP as a network flow problem, which has been successful for moderately-sized, real world problems. This approach is limited, however, because it can be difficult to model more complex constraints and to solve very large problems.

3. **Metaheuristic Approach –** The metaheurstic approach has been shown to be successful in finding feasible, although not necessarily near-optimal, solutions for large CSPs. Metaherustics allow complex cosntraints to be incorporated and produce solutions with fast computation times. This approach is also useful for making marginal changes to a crew schedule.

The work in this thesis aims to combine the SPP approach with a GA metahuerstic to solve the large, real world Amtrak Northeast Corridor CSP. The SPP formulation of the problem handles the complex constraints of the CSP, and the goal of the GA is to drive a column generation heuristic that can produce solutions within a reasonable compution time.

Chapter 3: Problem Description

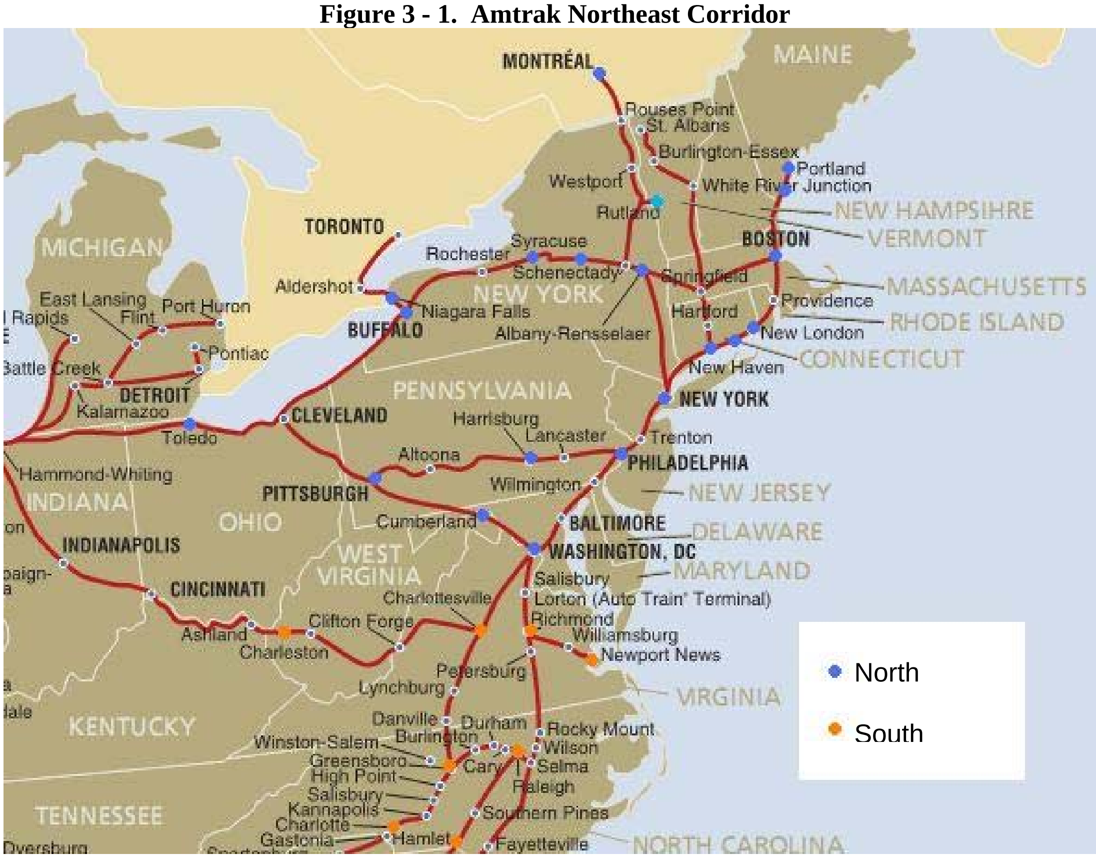
The objective of this project is to apply a genetic algorithm-based column generation heuristic to the passenger rail crew scheduling problem. This chapter describes the problem, input data, and constraints in detail.

3.1 Description of Data

Data on crew requirements and schedule for Amtrak's Northeast Corridor were obtained from Amtrak for the summer operating season of 2008. The requirements indicate which duties need to be fulfilled by the crew, and the existing schedule provides a basis for determining the existing cost of the crew schedule. A sample dataset is provided in the Appendix. Full datasets are not provided due to the excessive size of all data.

Amtrak routes cover 13 zones across the United States. For the purposes of this thesis, the Northeast Corridor zone was studied. The Northeast Corridor is the busiest corridor in the Amtrak network with the heaviest ridership between Boston and Washington, D.C. This zone covers 32 different stations, allowing for billions of unique schedules (Figure 3-1). Within a week's time, the schedule must be covered by a specific number of crew members. For instance, a single train may require two conductors, one engineer, and one fireman. A train with more cars or heavier ridership may require more crew members.

Figure 3 - 1. Amtrak Northeast Corridor



The crew requirements dataset contains the duties required to fulfill all trains for 32 total release stations and 16 crew bases (Table 3-1). The release stations are stations at which crew may go on or off duty and may be released for rest. There are many other stations in the Amtrak Northeast Corridor that are not included since they are not release points.

These duties in the requirements dataset includes train work, extra board, and station yard work. The dataset provides information on the date and time of when a duty begins and ends, the starting and ending locations of the duty, the train number, the number of crew needed for each train, and the type of crew required. The crew type may be one of four positions:

1. Conductor,
2. Assistant Conductor,
3. Engineer, or
4. Fireman.

Of the 4,338 current duties, 2,093 are engineer and firemen positions, and 2,245 are conductor positions and assistant conductor positions. The engineer and firemen positions can be filled by the same type of personnel as these crew types share the same responsibilities. The conductor and assistant conductor positions can also be filled by the same type of personnel. However, engineering and firemen positions are not interchangeable with conductor and assistant conductor positions. This conveniently allows the route information to be divided among those two groups for data processing and solving. By partitioning the single, full network problem as two independent problems, the problem size is dramatically reduced. The full Northeast Corridor network has 4,338 duties. After partitioning this problem into one for the engineers and firemen and one for the conductors and assistant conductors, the two new problems have 2,093 and 2,245 duties, respectively. These two crew-specific problems will be referred to as EF, for engineers and firemen, and AC, for conductors and assistant conductors.

Table 3 - 1. Amtrak Northeast Corridor Stations and Crew Bases

Code	***Station Location***
ALB*	Albany, NY
BON*	North Station, Boston, MA
BOS*	South Station, Boston, MA
BUF*	Buffalo, NY
CLT	Charlotte, NC
CUM	Cumberland, MD
CVS	Charlottesville, VA
FLO	Florence, SC
GRO	Greensboro, NC
HAM	Hamlet, NC
HAR*	Harrisburg, PA
HRB*	Harrisburg, PA
HUN*	Huntington, WV
MTR	Montreal, QC
NFL	Niagara Falls, NY
NFS	Niagara Falls, ON
NHV*	New Haven, CT
NLC	New London, CT
NPN	Newport News, VA
NYP*	Penn Station, NY
NYZ*	Penn Station, NY
OSB	Old Saybrook, CT
PGH	Pittsburgh, PA
PHL*	Philadelphia, PA
POR*	Portland, ME
RGH	Raleigh, NC
RUD	Rutland, VT
RVR*	Richmond, VA
SPG*	Springfield, MA
SYR	Syracuse, NY
TOL	Toledo, OH
UCA	Utica, NY
WAS*	Washington, DC
WDC*	Washington, DC
WWS*	Washington, DC

*crew base

The schedule dataset lists all current work schedules for each individual crew member. For each individual crew member's schedule, the data indicates the duties, pairings, and away time for a single week. The current Amtrak schedule in electronically documented form takes up over four megabytes of memory. In order to further reduce the complexity in coding required to process this large amount of data efficiently, the Northeast Corridor zone was partitioned into a North and South region (Figure 3-1). The current train network has a single station at the Washington, D.C. crew base that divides the network into two sections. All trains originating from north of Washington, D.C. must travel through this station in order to arrive at a station south of Washington, D.C. As a result, North region for this thesis was designated as all stations north of Washington, D.C., and the South region was designated as all stations south of Washington, D.C. Additionally, within the existing schedule provided by Amtrak, the crew base for the Washington, D.C. station was coded as two distinct crew bases. One Washington, D.C. crew base only served stations north of the station, and the other crew base only served stations south of the station.

Table 3 - 2. Description of Problems

Problem	***Crew Type***	***Region***	***Code***	***Number of Duties***
1	Engineers & Firemen	North	EFN	1044
2	Conductors & Assistant Conductors	North	ACN	1178
3	Engineers & Firemen	South	EFS	1049
4	Conductors & Assistant Conductors	South	ACS	1067
5	Engineers & Firemen	All	EF	2093
6	Conductors & Assistant Conductors	All	AC	2245

The current schedules connecting the north and south regions are assumed to be static throughout the study, which makes the two partitions into independent networks. This partitioning in conjunction with the previously described specialized crew assignments creates a total of four separate problems for the Northeast Corridor (Table 3-2):

1. ACN - conductors and assistant conductors in the north partition,
2. ACS - conductors and assistant conducts in the south partition,
3. EFN - engineers and firemen in the north partition, and
4. EFS - engineers and firemen in the south partition.

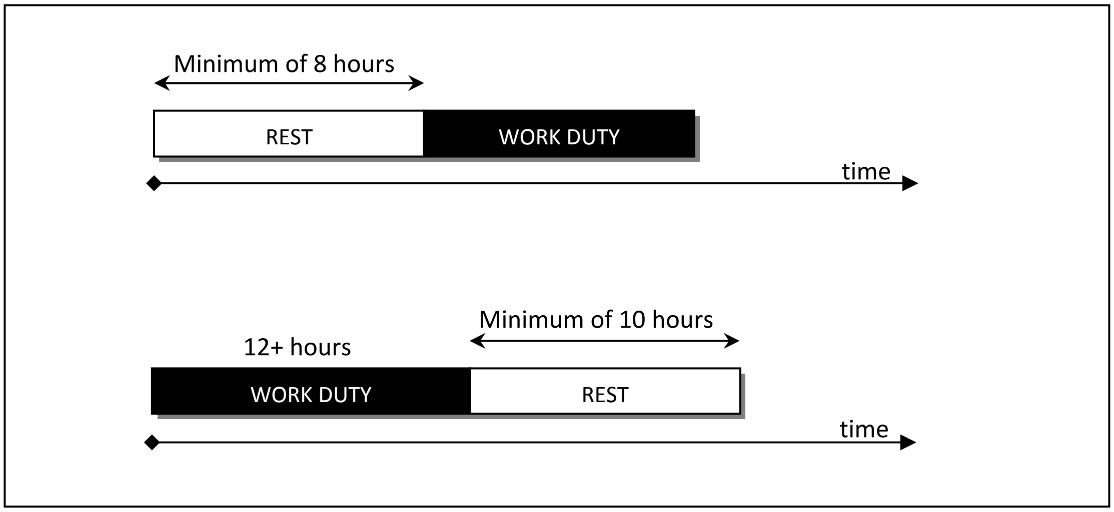
3.2 Description of Constraints

3.2.1 Crew Work Regulations

Amtrak T&E crew members are subject to specific work regulations. First, T&E crew may be required to stay at away locations between trips or overnight, but must return home at the end of a shift. Second, T&E crew schedules permit deadheading of crews. Deadheading may require additional cost for transporting crew to the appropriate locations, but is necessary for some cases in order to meet all work requirements.

In accordance with union regulations, T&E crew are guaranteed minimum rest times (Figure 3-2). Before working a shift, crew must be allowed at least eight consecutive hours of rest during the 24 hours prior to working. After being on duty for 12 consecutive hours or more, the crew must have at least 10 consecutive hours of rest. In addition, crew members must have at least one entire day without work per week.

Figure 3 - 2. Minimum rest time regulations



3.2.2 Crew Payment Regulations

In this project, the regular hourly rate paid to the crew is assumed to be $33.60, which is based on the real Amtrak crew cost. The “one-and-one-half” overtime rate is thus equal to $50.40. The categories of payment regulations include:

- Shift work payment,
- Meal allowance,
- Lodging allowance, and
- Weekly 40-hour guarantee.

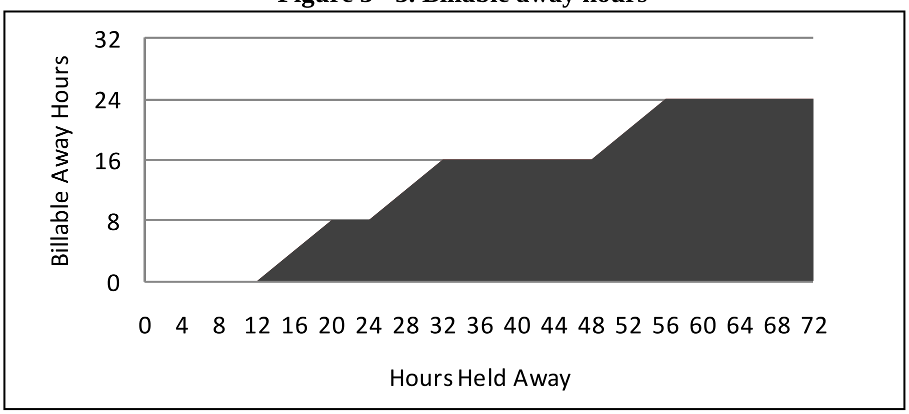
Shift work payment includes pay for hours on-duty and billable away hours. The on-duty hours are time spent on a train trip, yard work, or extra board hours. Thus, the duty hours for a duty is simply the starting date and time of the duty subtracted from the ending date and time of the duty:

$$T_{TR} = \text{train hours for duty i} = t_{end_i} - t_{start_i}$$

When held away, the number of billable away hours is a function of the total number of hours held away; crew are only paid for certain hours held away (Figure 3-3). Crew are not paid for the first 12 consecutive hours away. After being held away for 12 consecutive hours, Amtrak pays the crew for up to eight hours every 24 consecutive hours away. For the first 30 hours away, the billable away hours are:

$$T_{AW} = \text{billable away hours} = \begin{array}{lccccc} 0, & 0 & \leq & (t_{start_{(i+1)}} + t_{end_i}) & < & 12 \\ (t_{start_{(i+1)}} + t_{end_i}) - 12, & 12 & \leq & (t_{start_{(i+1)}} + t_{end_i}) & < & 20 \\ 8, & 20 & \leq & (t_{start_{(i+1)}} + t_{end_i}) & < & 24 \\ (t_{start_{(i+1)}} + t_{end_i}) - 4, & 24 & \leq & (t_{start_{(i+1)}} + t_{end_i}) & < & 30 \end{array}$$

Figure 3 - 3. Billable away hours



Calculation of shift payment is based on entire shifts rather than individual work duties. The Amtrak regulation states that the first eight hours of a shift are paid at the regular hourly rate, and any additional hours in the shift above eight hours are paid at the overtime rate. Therefore, the cost to Amtrak for a single shift, where T_s is the total number of paid hours in a shift, is:

$$C_s = \begin{cases} 33.6 * T_s, & 0 \leq T_s \leq 8 \\ 50.4 * T_s, & 8 \leq T_s \end{cases}$$

The second payment category covers meal allowances for crew during duties and while away. If a train duty is at least five hours long, Amtrak provides crew members an allowance of $3.50 every five hours. When held away for at least four hours, the company provides an allowance of $6.00 and another $6.00 once every eight hours afterwards. The cost of meal allowances for a duty is:

$$C_M = 3.5 * \left\lfloor \frac{t_{A_i} - t_{D_i}}{5} \right\rfloor + 6 * \left\lfloor \frac{t_{D_{i+1}} - t_{A_i} + 4}{8} \right\rfloor$$

The third category of payment reimburses crew for lodging when held away for at least four hours. Lodging allowance is assumed to be $70.00, and additional lodging allowance is provided once every 24 hours after the first four hours of away time. Thus, the cost of lodging for a duty is:

$$C_L = 70 * \left\lfloor \frac{t_D - t_A + 20}{24} \right\rfloor$$

The last payment category guarantees that all crew members are paid for at least 40 hours of work each week. If the total working time and paid away time is less than 40 hours, Amtrak pays for the additional hours at the regular rate of $33.60 dollars per hour. This case is disadvantageous to Amtrak since the company essentially pays the crew member for hours not worked. If the total working time and paid away time is over 40 hours, however, Amtrak pays for hours over 40 hours at the one-and-one-half overtime rate. Again, paying crew for additional overtime is not of direct benefit to Amtrak. However, good crew schedules may include many pairings with additional overtime because these pairings cover more duties. Efficient crew schedules have a good balance between the cost of paying the 40-hour guarantee and the benefit of that pairing covering more duties.

3.3 Current Crew Schedule

In order to manipulate and study the current crew schedules from Amtrak, a consistent set of rules governing the data must be present among all different problems studied. This means that certain duties need to consistently be removed across all groups of data so that the remaining data can be compared. The following are rules for data removal:

- *Duties on train outside of the defined corridor in this thesis.* Since different operating units in Amtrak have different definitions of the "Northeast Corridor", the corridor defined in this thesis is limited to trains along the main corridor between Boston, New York, and Washington, D.C. Some duties on train outside of the defined region were not included in the analysis.
- *Duties that do not serve Amtrak trains but are included in the dataset for operations purposes*. For example, local agency-operated commuter trains that may share tracks with Amtrak are listed in the raw dataset.
- *Duties with missing data*. Some duties had missing data for starting or ending locations and times, and thus, these duties could not be included in the analysis.

The current crew schedules for the North portion of the Northeast Corridor after the above mentioned duties are removed show potential for savings in cost. The easiest way to identify that schedules can be improved is the varying length of the pairings. Within the data, there currently exists many short (2-3 duties) pairings and many long (10 or more duties) pairings. This condition suggests that crew members are likely paid for hours not worked due to the cost guarantee or overtime. The current weekly costs of the crew schedules for each problem are shown in Table 3-3.

Table 3 - 3. Current Crew Schedule Costs

Crew Schedule	*Weekly Cost ($)*
ACN	$426,879
EFN	$237,782
ACS	$631,892
EFS	$451,120
AC	$1,110,099
EF	$709,902

Chapter 4: Methodology

This chapter describes the methodology used to solve the passenger rail crew scheduling problem.

4.1 Model Formulation

The problem is formulated as the general set partition problem (SPP) used in crew scheduling, described in Section 2.5.1. This basic formulation was selected because it is arguably the most versatile operations research approach to the crew scheduling problem (CSP). Since the specific problem in this thesis requires complex constraints, the SPP allows the constraints to be simply modeled as part of the cost and binary constraint matrix. The drawback of the SPP formulation is that the entire set of feasible pairings cannot be generated and solved efficiently. The total number of feasible pairings grows exponentially with the number of duties required. For a set of n duties, there are 2^n possible pairings, and any number can be feasible, depending on the constraints. The following solution approach aims to efficiently create a subset of "good" pairings so that the entire set of pairings does not need to be generated and solved.

4.3 Data Processing

Prior to solving the problem, the raw data from Amtrak was processed to prepare it for input into the computer program. Processing the data involved preparing the two input files for the computer program. The first input file stored the required duties for the problem as a two-dimensional array of n by 4 entries. In this file, each row represents a duty, where the first two entries are the starting and ending station location, and the last two entries are the starting and ending times (Figure 4-1).

Figure 4 - 1. Sample Required Duties Input File

	0 – From	***1 – To***	***2 – Start Time***	***3 – End Time***
0 – Duty 1	10	0	360	629
1 – Duty 2	0	10	820	1181
…				
n-1 – Duty n	10	3	1230	1416

The station locations are coded as distinct integers representing the stations in Table 4-1. The duty starting and ending times are coded in minutes. This file is sorted by start time, which allows duty information to be expressed chronologically.

The second input file stores the current crew schedule as an array of the duties and crewbases. All of the duties from the first file are mapped to numerical identifiers for efficient storage. For each crewperson's schedule, the duties are listed in numerical form, followed by "999" to signify the end of the duties, then the crewperson's crewbase (Figure 4-2). The crewbase must be listed in the input file since the cost of a schedule depends on how long a crewperson is away from the crewbase. The array ends with the integer "9999" to confirm the end of the file.

Table 4 - 1. Crewbase Identification

Northern Partition	
Crewbase	*ID*
ALB	0
ALP	1
BON	2
BOS	3
BUF	4
MTR	5
NFL	6
NFS	7
NHV	8
NLC	9
NYP	10
OSB	11
POR	12
RUD	13
SPG	14
SYR	15
TOL	16
UCA	17

Southern Partition	
Crewbase	*ID*
CLT	0
CUM	1
CVS	2
FLO	3
GRO	4
HAM	5
HAR	6
HUN	7
NPN	8
NYP	9
PGH	10
PHL	11
RGH	12
RVR	13
WAS	14

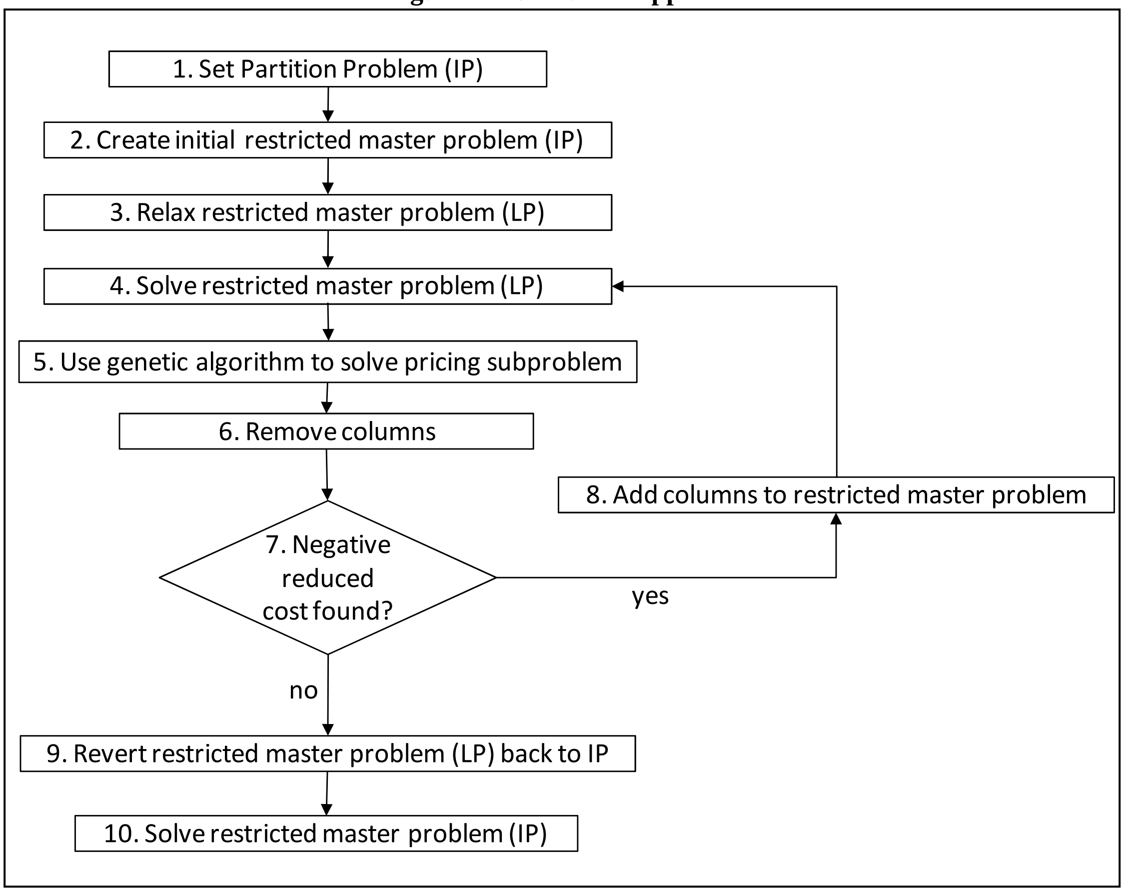
Figure 4 - 2. Sample Crew Schedule Input File

```
5
7
16
56
999
5
22
59
89
110
999
2
…
9999
```

4.2 Solution Approach

The solution approach used to solve the problem is a genetic algorithm-based column generation heuristic. Figure 4-3 provides an overview of this solution approach.

Figure 4 - 3. Solution Approach



4.2.1 Master Problem

The master problem is the full integer program (IP): the SPP formulation of the crew scheduling problem. This master problem is never completely generated due to the excessively large number of pairings. The master problem serves as a starting point of for the rest of the solution method.

4.2.2 Restricted Master Problem

In large CSPs, it is not efficient to enumerate all possible pairings or columns, so a restricted subset of the set of all feasible pairings is created. This restricted SPP is also an IP. Following a general column generation algorithm, the LP relaxation of the IP is used for the pricing subproblem described in the next section.

In each iteration, the relaxed restricted master problem is solved by calling the commercial optimization package software CPLEX through the CPLEX Callable Library. The library is used in code written in the C programming language within Microsoft Visual C++ 6.0. The computer program uses the two input files previously described and first establishes the cost of each pairing based on the complex set of constraints. Using this data, the program calls CPLEX to solve the relaxed, restricted LP. The purpose of solving the restricted LPs is to find the dual variables, or shadow prices, corresponding to the duties in the problem for each restricted subset of pairings. These values are used to calculate the reduced cost of new pairings in the pricing subproblem.

4.2.3 Pricing Subproblem

The pricing subproblem is used to determine which new pairings should be generated and added to the restricted problem. This step is executed within the C program and does not require the use of CPLEX.

According to LP duality theory, a solution improves when variables with negative reduced cost enter the basis. As a result, the pricing subproblem aims to identify pairings (which are also columns or variables) that improve the LP relaxation master problem. Using the dual variables from the LP solution, the reduced cost, $\bar{c}$, of each pairing can be calculated using the following equation from LP duality theory:

$$\overline{c_j} = c_j - \sum_i \lambda_i a_{ij}$$

Where: $\overline{c_j}$ = reduced cost of pairing j

c_j = cost of pairing j

λ_i = dual variable of duty i

$$a_{ij} = \begin{cases} 1, & \text{pairing j covers work requirement i} \\ 0, & \text{otherwise} \end{cases}$$

The pricing subproblem effectively "prices out" newly generated pairings to determine a good candidate for adding to the restricted problem. The goal is to generate new pairings that improve the LP relaxation of the restricted master problem, which in turn, is likely to improve the IP master problem. Thus, the objective of this pricing subproblem is to minimize the reduced cost of pairings not already in the restricted subset. If the solution to this pricing subproblem is negative or zero, then the corresponding pairing should enter the restricted subset. In order to make the iterations more efficient, many pairings with negative reduced costs can be brought into the restricted subset in each iteration. If the solution is non-negative, then there is no additional pairing that can improve the LP relaxation of the master problem.

4.2.4 Pairing Generation

The solution approach uses a genetic algorithm (GA) metaheuristic to generate new pairings for the pricing subproblem. A GA is a global search heuristic that uses evolutionary algorithms to find approximate solutions to optimization problems. GAs work by using the notion of biological reproduction to create candidate solutions that evolve into improved solutions. The evolution initiates with a population of individuals, which in this case, is the restricted subset of pairings.

In this metaheuristic, a pairing is stored as a binary chromosome (Figure 4-4). In Figure 4-4, the sample pairing for a crew based at station 9 is required to work duty 1, 3, and 5, before returning the crewbase. If the problem has five total duties, the chromosome has five genes. Then, a "1" in a gene represents a selected duty, and a "0" represents non-selected duty. This chromosome representation of a pairing corresponds to a column in the binary matrix of the SPP formulation of the CSP. The fitness of the pairing is equal to the reduced cost of the pairing, which is calculated from the cost, dual variable, and assigned duties. In addition to the chromosome, a separate array keeps record of the crew base of each pairing.

Figure 4 - 4. Chromosome Representation Sample

Pairing:

Crew Base 9 → Duty 1 → Duty 3 → Duty 5 → Crew Base 9

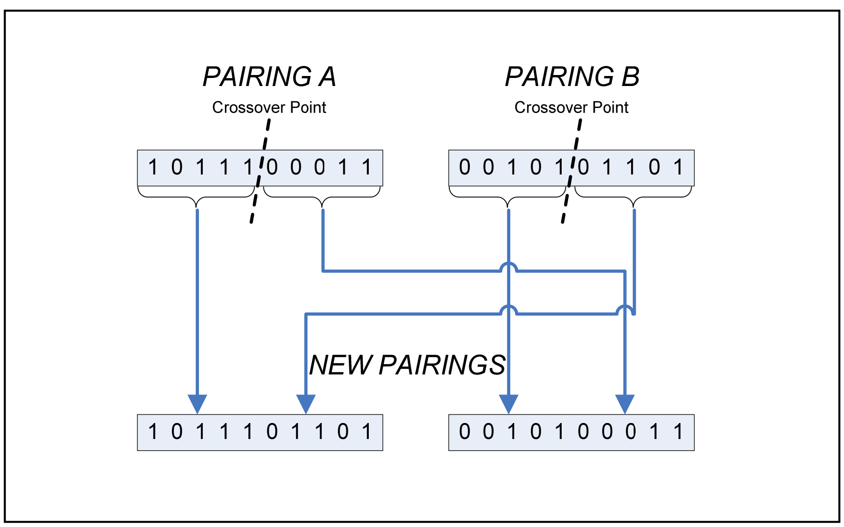
Chromosome:

Duty 1	Duty 2	Duty 3	Duty 4	Duty 5
1	0	1	0	1

As previously stated, the genetic algorithm initializes with the current subset of pairings in the restricted problem. The two pairings with the lowest fitness value are selected from the population to be “parents”.   Once parents are selected, the parents reproduce to create “children”. In this solution approach, reproduction occurs through a simple one-point crossover (Figure 4-5). The crossover point is selected randomly for this approach. In Figure 4-5, two pairings, Pairing A and Pairing B, are the parents. The crossover point is in between the fifth and sixth gene in the chromosome for this example. The two new pairings that are created then share genes from each parent.

The crossover creates a possible pairing, and a subsequent feasibility operator tests to ensure that the pairing is feasible. A pairing is considered feasible if it is feasible to the overall crew scheduling problem, and it is not already in the subset of pairings in the restricted problem. In the application of the GA to the CSP in this thesis, a large number of pairings are infeasible due to the constraints. An additional mutation operator facilitates the genetic diversity of subsequent populations. This reproduction process iterates to create many new populations of pairings for the restricted problem.

Figure 4 - 5. Example of One-Point Crossover



The genetic algorithm terminates when a fixed number of feasible pairings are created. The population size, n, is limited to a defined number to prevent the problem from becoming excessively large. These pairings are added to the restricted problem. The metaheuristic also terminates if no new feasible pairings are found within a given amount of time. The latter stopping criterion makes the solution approach more efficient and prevents the program from searching for feasible pairings for an excessive amount of time.

4.2.5 Pairing Removal

The pairing generation process creates a large number of new pairings. In order to prevent an excessive number pairings in the pricing subproblem, pairings can be removed. After each iteration of solving the restricted program, the new reduced costs of each pairing can be calculated. The pairings that are not selected to be in the restricted LP solution and with a positive reduced cost are removed since these are less likely to improve a subsequent LP solution. The removal of these less-optimal pairings reduces the overall size of a subproblem, which reduces run-time and memory usage requirements.

4.2.6 Final Solution

When no further pairings with negative reduced cost can be found by the genetic algorithm, the relaxed, restricted master problem can be solved to optimality or near-optimality. The final relaxed, restricted master problem is reverted back to an IP. Then, this IP is solved to optimality using CPLEX. The resulting solution from this IP is the final solution to the CSP. In terms of the Amtrak schedules, this means that the resulting solution ensures that each train has the same amount of workers required to operate, but the overall cost of operation is reduced.

4.5 Testing Strategy

In order to test the proposed heuristic solution approach, three small sample problems were created based on the actual data. These problems have 26, 28, and 30 duties. The duties for the 26 duty problem are based on the EFN data and are shown in Table 4-2. The duties for the 28 and 30 duty problems are based on the EFS data and are shown in Table 4-3 and 4-5. The existing crew schedules for these problems are shown in Table 4-6.

While these problems are small in comparison to the real-world problems with approximately 1,000 to 2,000 duties, the number of possible pairings for these small problems is large (Table 4-1). For a problem with 30 duties, there are 1,073,741,824 possible pairings, although the actual number of *feasible* pairings is smaller due to the complex constraints of the problem. The goal of using the heuristic method is to drastically reduce the large set of possible pairings and only select "efficient" pairings to solve in the SPP.

The small problems were first formulated as SPP IPs and solved to optimality using CPLEX. The results served as a base for comparison. The small problems were then solved using the proposed solution approach, and the results were compared.

Next, the real-world problems were solved using the proposed solution approach. A sensitivity analysis was then conducted to determine the effects of changes in cost, schedule, and genetic algorithm population size on the solution.

Finally, conclusions were drawn from the results. Each of the four large problems, ACN, EFN, ACS, EFS, was solved using the heuristic with a population size, n, of 2,000, 4,000, and 20,000 pairings. Finally, the entire AC and EF Amtrak Northeast Corridor problems were solved separately using the heuristic.

Table 4 - 2. Generated Small Problem with 26 Duties

Duty	*From*	*To*	*Depart*	*Arrive*
0	0	4	605	1270
1	0	4	605	1270
2	4	0	1890	2525
3	4	0	1890	2525
4	0	4	2045	2710
5	0	16	2160	2322
6	16	0	2417	2605
7	4	0	3330	3965
8	0	4	3485	4150
9	0	4	3485	4150
10	0	16	3600	3762
11	16	0	3857	4035
12	4	0	4770	5405
13	4	0	4770	5405
14	0	4	4925	5590
15	4	0	6210	6845
16	0	4	6365	7030
17	0	4	6365	7030
18	0	16	6480	6642
19	16	0	6737	6915
20	4	0	7650	8285
21	4	0	7650	8285
22	0	4	7805	8470
23	0	4	7805	8470
24	4	0	9090	9725
25	4	0	9090	9725

Table 4 - 3. Generated Small Problem with 28 Duties

Duty	*From*	*To*	*Depart*	*Arrive*
0	5	0	457	693
1	0	5	482	753
2	0	5	1093	1314
3	5	7	1309	1469
4	7	5	1886	2045
5	5	0	1853	2156
6	5	0	2464	2765
7	0	5	2515	2749
8	5	7	2754	2936
9	7	5	3318	3515
10	0	5	3352	3660
11	5	7	4186	4370
12	7	5	4752	4933
13	5	0	4740	5039
14	5	0	5358	5631
15	0	5	5378	5634
16	5	7	5649	5811
17	7	5	6198	6360
18	0	5	6262	6496
19	5	7	7088	7251
20	7	5	7621	7828
21	5	0	7630	7898
22	5	0	8259	8519
23	0	5	8279	8555
24	5	7	8505	8704
25	7	5	9052	9268
26	5	0	9052	9359
27	0	5	9141	9387

Table 4 - 4. Number of Possible Pairings for Small Problems

No. of Duties	*No. of Possible Pairings*
26	67,108,864
28	268,435,456
30	1,073,741,824

4.5.1 Evaluation of Convergence and Cost

The solutions produced by the proposed iterative heuristic were studied after the resulting final costs have converged. This ensured that the maximum benefit of the method is achieved when compared to alternative methods. Many of the changeable parameters when using the proposed heuristic have a direct effect on the number of iterations needed for convergence. The population size, costs, schedule complexity, and method of generated pairings are some of these parameters. The number of pairings generated per iteration has a direct affect on cost. Varying these factors allows for the full analysis of the proposed heuristic and full optimization of the heuristic.

4.5.2 Evaluation of Calculation Time

Calculation times were determined using a Windows based machine with dual core processor and two gigabytes of random access memory. Each iteration time as well as total runtime using the heuristic or the full subset of pairings was logged for study. Calculation time using the heuristic was calculated by the summing the running times of each iteration. The relationship between calculation time and iteration number can be determined within solving each single problem. The relationship between calculation time and the number of pairings generated per iteration will require varying the number of pairings. These values can then be compared to the cost improvement from using the heuristic to determine optimal parameters for efficiency.

Table 4 - 5. Generated Small Problem with 30 Duties

Duty	From	To	Depart	Arrive
0	0	10	0	360
1	1	0	10	420
2	2	0	10	995
3	3	10	3	1230
4	4	3	10	1799
5	5	10	0	1800
6	6	10	0	2412
7	7	0	10	2435
8	8	10	3	2670
9	9	3	10	3239
10	10	0	10	3300
11	11	10	3	4110
12	12	3	10	4679
13	13	10	0	4680
14	14	10	0	5292
15	15	0	10	5315
16	16	10	3	5550
17	17	3	10	6119
18	18	0	10	6180
19	19	10	3	6990
20	20	3	10	7559
21	21	10	0	7560
22	22	10	0	8172
23	23	0	10	8195
24	24	10	3	8430
25	25	3	10	8999
26	26	10	0	9000
27	27	0	10	9060
28	28	10	0	9612
29	29	0	10	9635

4.5.2 Evaluation of Memory Usage

Similar to calculation time, memory management was also logged during each CPLEX iteration. The C program managing all CPLEX calculations and pairing information stores only relevant route identification and does not store unused route information. While more extensive program techniques could reduce memory usage, the memory usage of this program should give good indication of how memory usage would scale on a typical, modern personal computer with parameters beyond the scope of what was studied. This allows for an approximation of a possible maximum generated pairs per iteration and approximate calculation time for convergence.

Table 4 - 6. Small Problem Existing Crew Schedules

26 Duty Problem			28 Duty Problem			30 Duty Problem		
Pairing	*Home*	*Duties*	*Pairing*	*Home*	*Duties*	*Pairing*	*Home*	*Duties*
1	16	8 12 16 20	1	5	5 3 4 11 12 19 20	1	10	0 2 14 18 22 27
2	0	0 2 18 19 22 24	2	5	1 6 10 21 23	2	0	1 6 10 21 23 28
3	0	4 7 14 15	3	0	8 9 16 17 24 25	3	10	5 7 13 15 26 29
4	4	5 6 9 13 17 21	4	0	5 7 13 15 26	4	10	8 9 16 17 24 25
5	0	1 3 10 11 23 25	5	5	2 14 18 22 27	5	10	3 4 11 12 19 20

Chapter 5: Results & Analysis

This chapter presents the results and provides analysis of the improved solution. First, the small generated problem results are presented, followed by the results for the four partitioned problems. Lastly, the full network problems are presented.

5.1 Small Generated Problem Results

In order to determine whether or not the proposed heuristic can improve the crew schedule, results for the small problems were compared. The data for the real world Amtrak crew schedules are too large to generate a complete list of feasible pairings. The time required to first generate the feasible pairings is excessive. Then, solving such a large integer program within a reasonable time would be impossible using modern computers. As a result, three small problems based on the real world data were generated. These small problems had 26, 28, and 30 duties each. As the number of duties increases, the number of possible pairings grows exponentially. The small problem with 28 duties is significantly larger than the 26 duty problem, and the 30 duty problem is again much larger than the 28 duty problem. By solving each of these problems to optimality using the traditional integer programming solution techniques in CPLEX and comparing the results to those obtained by the heuristic, it was found that the heuristic does improve the crew schedules.

The smallest problem with 26 duties was solved to optimality by the heuristic in less than 16 seconds (Table 5-2). The solution converged in two iterations. The final solution was 17.7 percent improvement over the original total cost.

Table 5 - 1. Small Generated Network Comparison 26 Duties

Iteration	*Total Cost ($)*	*Improvement (%)*	*Calculation Time (sec)*
0	8593	NA	NA
1	7074	17.7%	0.9
2	7074	17.7%	15.0

The next problem with 28 duties was solved by the heuristic in about 22 seconds and within four iterations (Table 5-3). The final solution for this problem had an 18.4 percent improvement over the original total cost. Compared with the smaller 26 duty problem, this larger problem required more time solve and more iterations to reach convergence. This result was expected due to the larger number of possible pairings and larger number or feasible pairings.

Table 5 - 2. Small Generated Network Comparison 28 Duties

Iteration	*Total Cost ($)*	*Improvement (%)*	*Calculation Time (sec)*
0	8920	NA	NA
1	9594	14.9%	1.1
2	7074	16.2%	4.0
3	7280	18.4%	8.2
4	7280	18.4%	24.6

Table 5 - 3. Small Generated Network Comparison 30 Duties

Iteration	*Total Cost ($)*	*Improvement (%)*	*Calculation Time (sec)*
0	9776	NA	NA
1	9594	1.9%	1.2
2	7074	19.8%	7.0
3	7834	19.9%	31.0
4	7834	19.9%	31.0

The last small problem with 30 duties was solved in about 31 seconds, with convergence at the fourth iteration (Table 5-4). The total improvement was 19.9 percent over the original total cost. Again, the larger number of possible and feasible pairings for this problem made the calculation time longer.

In all three small problems, the initial iteration produced a smaller improvement, while subsequent iterations produced the largest improvements. These results may have occurred because the first iteration creates “good” parent pairings that may not improve the overall solution dramatically, but create “good” children pairings in the later iterations. All three small problem results also indicated that the calculation times of later iterations were longer. This result was also expected since the later iterations have a larger number of pairings to solve in the integer program in CPLEX. Solving these larger integer programs is a more intensive process.

The heuristic results for these three small problems are then compared to the optimal results obtained from CPLEX. The problems were formulated as SPPs and solved the optimality. The results are shown in Table 5-5.

Table 5 - 4. Comparison of Small Problem Heuristic Solution and Optimal Solution

No. of Duties	Total Cost ($)		Improvement (%)		Calculation Time (sec)	
	Heuristic	Optimal	Heuristic	Optimal	Heuristic	Optimal
26	7074	7074	17.7%	17.7%	15	79
28	7280	7280	18.4%	18.4%	22	889
30	7834	7792	19.9%	20.3%	31	6284

For the 26 duty problem, the heuristic solved the problem to optimality in 15 seconds, which is far less than the 79 seconds required for the full SPP formulation solution. The heuristic computation time was 81 percent less time than the traditional SPP IP solution method. The heuristic also solved the 28 duty problem to optimality in 22 seconds, compared with the 889 seconds required for the SPP solution method.

The heuristic, however, did not solve the largest small problem with 30 duties to optimality. The heuristic solution improvement was 19.9 percent, while the optimal solution improvement was 20.3 percent. The total cost of the final heuristic solution was within 0.5 percent of the optimal solution. The proposed heuristic also reached a final solution in 31 seconds, compared with 6,284 seconds for the optimal solution using the traditional SPP IP formulation method.

As the number of duties increases beyond 30 duties, it is expected that the proposed heuristic solution would be increasingly less optimal. However, the savings in calculation time appears to increase exponentially as the number of duties increases. Based on only these three small problems, the total calculation time using the heuristic also appears to increase linearly as number of duties increases. The calculation time for solving the traditional SPP IP formulation to optimality using CPLEX increases exponentially as the number of duties increases. Based on this the calculation times of the small problems, a problem with 35 duties would conservatively require several days to solve. The excessive calculation time required to find optimal solutions for the real world-sized problems demonstrates the need for more a efficient solution method. For the problems in this work with over 1,000 duties, generating all feasible duties and then solving the problem to optimality becomes nearly impossible due to computer memory and calculation time constraints.

5.2 ACN Results

5.2.1 ACN Results for n = 2,000

The results for the ACN problem solved with a population size, n, of 2,000 are shown in Table 5-1 and Figure 5-1. The final solution for the ACN problem was a

4.45 percent improvement over the original cost of $426,879 dollars per week, as determined by calculating the cost of the existing schedule. Over one year, the total savings in cost would be $763,308 dollar per year. Since there were some simplification assumptions in the work for this thesis, the actual savings could differ. However, the results indicate that there are potential cost-savings in improving the crew schedules.

The total cost for this problem is reduced linearly after each iteration before the results converge at the thirteenth iteration. Afterwards, no cost reduction is observed. Memory usage varied based on the average length of the pairings generated, but did not show a consistent trend from iteration to iteration. It varied based on the collective lengths of the pairings generated in each iteration.

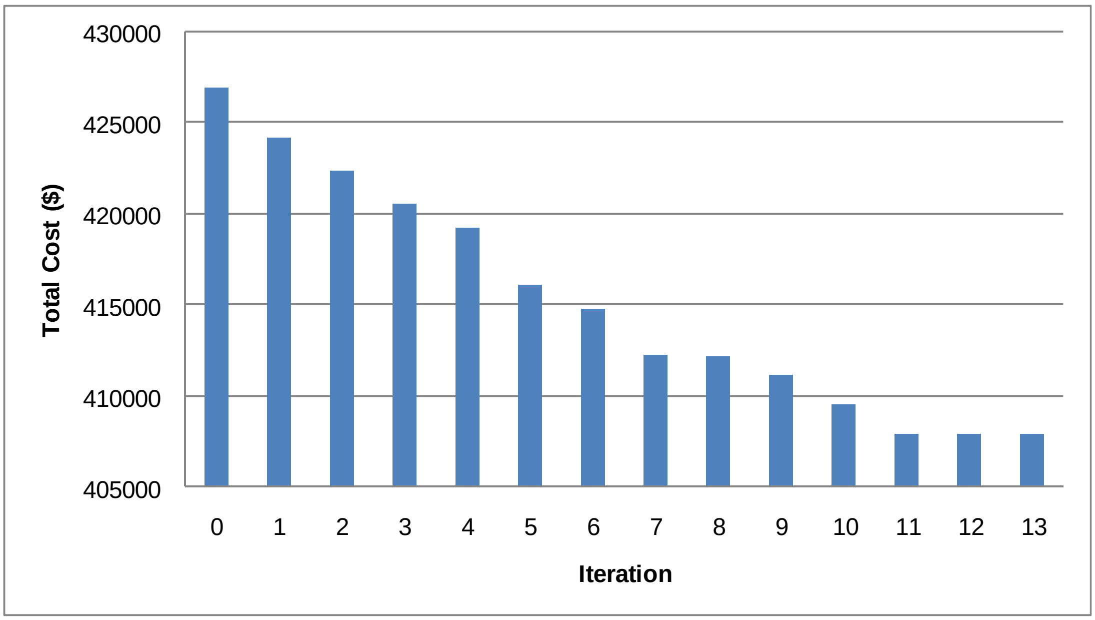
Table 5 - 5. Improvement in ACN Total Cost for n = 2,000

Iteration	Total Cost ($)	Improvement (%)	Calculation Time (sec)	Memory Usage (MB)
0	426879	NA	NA	NA
1	424192	0.63%	2966	4.2
2	422390	1.05%	2869	3.3
3	420600	1.47%	3229	5.5
4	419250	1.79%	2328	6.0
5	416086	2.53%	2512	5.1
6	414812	2.83%	2470	3.8
7	412276	3.42%	3124	3.9
8	412200	3.44%	2376	4.8
9	411184	3.68%	2657	4.8
10	409500	4.07%	2540	3.7
11	407868	4.45%	2565	6.2
12	407868	4.45%	2722	5.0
13	407868	4.45%	2653	5.0

Running with this number of pairings per iteration does not have a significant impact on the memory resources of a modern computer. The calculation time is more dependent on the data rather than computer resources.

Iteration 8 has minimal cost reduction compared to those from previous iterations. The solution ultimately converges at iteration 13 because all resulting iterations provide no changes in selected pairings and no cost reduction. The total calculation time when the solution converged was 9.72 hours. Memory usage of solving the problem was not intensive for each iteration, and there were no issues with memory limits during the solution process. The ACN problem had 1,178 duties and was the largest of the four partitioned problems.

Figure 5 - 1. Improvement in ACN Total Cost for n = 2,000



5.2.2 ACN Results for n = 4,000

The results when population size, n, is set at 4,000 are similar to when n is 2,000. However, for the ACN problem, there is a larger improvement in each iteration (Table 5-2 and Figure 5-2). This suggests that allowing a larger population size provides more opportunity for improvement.

The final solution for the ACN problem with a limit of 4,000 pairings shows a 7.4 percent improvement over the original total cost of $426,879 dollars per week. The potential annual savings in cost is $1,641,640 dollars per year.

In the solution process for this problem, there are certain iterations that produce larger improvements in total cost. For example, there is a relatively large improvement between iteration 6 and 7. These improvements suggest that the GA is producing good candidate pairings for the restricted problem. The improvements may result from the GA selecting "good" parents in previous iterations that produce better populations in the subsequent iterations. Convergence is then observed at iteration 15. This behavior is similar to when n is set to 2,000 and similar to the majority of the rest of the data sets.

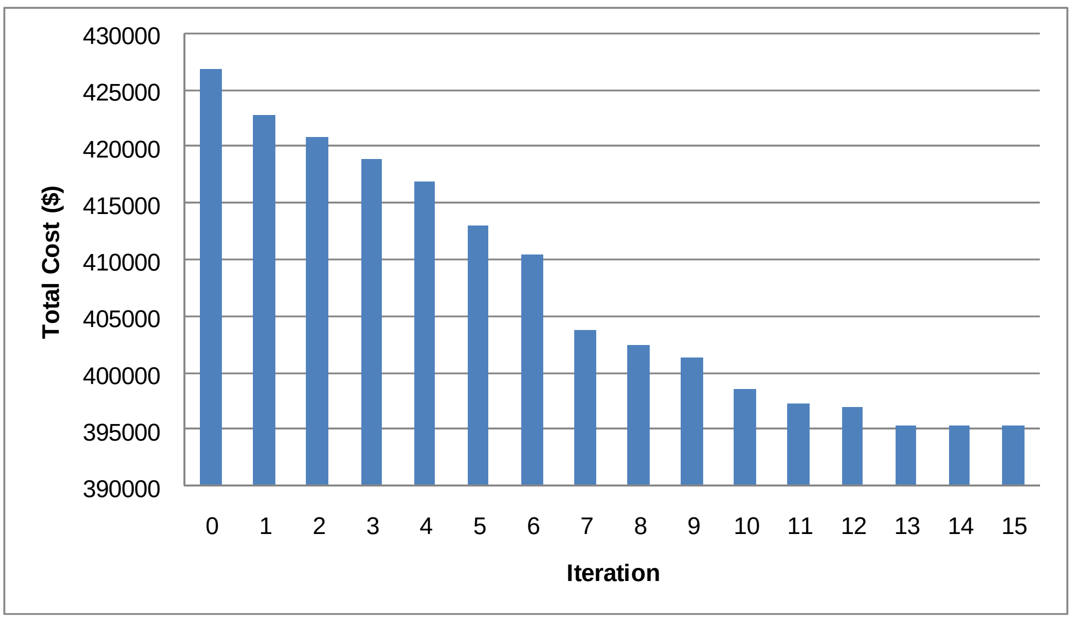
Table 5 - 6. Improvement in ACN Total Cost for n = 4,000

Iteration	*Total Cost ($)*	*Improvement (%)*	*Calculation Time (sec)*	*Memory Usage (MB)*
0	426879	NA	NA	NA
1	422742	0.97%	6114	8.4
2	420794	1.43%	4684	6.6
3	418828	1.89%	4840	11.0
4	416960	2.32%	6114	12.0
5	413014	3.25%	13792	10.2
6	410466	3.85%	5941	7.6
7	403770	5.41%	9959	7.8
8	402502	5.71%	6036	9.6
9	401319	5.99%	5796	9.6
10	398631	6.62%	6739	7.4
11	397263	6.94%	7273	12.4
12	396947	7.01%	6769	10.0
13	395309	7.40%	6742	10.0
14	395309	7.40%	8326	14.0
15	395309	7.40%	6412	12.0

As the population size doubled from 2,000 to 4,000, the final improved solution improves. However, while increasing the number of pairings results in a better solution, the running time of the heuristic is slower. The total calcalution time before convergence when n is 4,000 is about 29.3 hours.

As expected, the memory usage for this problem also increases as more pairings are stored per iteration. This calcalution time is significantly larger compared to when n is 2,000. The memory usage again was relatively constant for each iteration.

Figure 5 - 2. Improvement in ACN Total Cost for n = 4,000



5.2.3 ACN Results for n = 20,000

The best solution for the ACN network was found when population size is 20,000. This parameter produced a crew schedule with a 15.8 percent cost reduction compared to the original. The percentage of improvement is nearly four times that of the problem solved when n is 2,000. The new crew schedule cost is $359,257 dollars per week, compared to $426,879 dollars per week of the original schedule. This result is a savings of $67,622 dollars per week, or $3,516,344 per year. This is a significant amount of savings in operational cost.

The results for when n is 20,000 are less linear than when there are less pairings per iteration (Table 5-3 and Figure 5-3). Initially, the results follow almost an inverse relationship with respect to iteration.

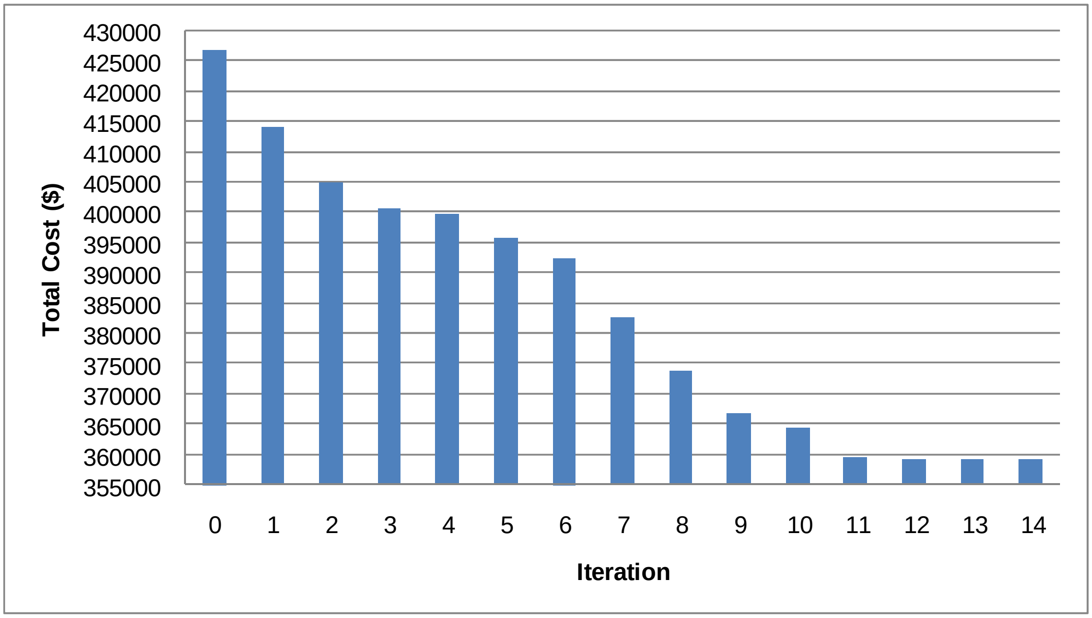
Table 5 - 7. Improvement in ACN Total Cost for n = 20,000

Iteration	*Total Cost ($)*	*Improvement (%)*	*Calculation Time (sec)*	*Memory Usage (MB)*
0	426879	NA	NA	NA
1	414166	2.98%	21015	42.0
2	404754	5.18%	27275	33.0
3	400482	6.18%	28118	54.5
4	399845	6.33%	40699	30.0
5	395834	7.27%	321298	51.0
6	392277	8.11%	124007	38.0
7	382670	10.36%	41836	38.0
8	373945	12.40%	89141	48.0
9	366639	14.11%	123059	48.0
10	364364	14.64%	98058	37.5
11	359484	15.79%	59005	62.0
12	359257	15.84%	70345	50.0
13	359257	15.84%	57528	50.0
14	359257	15.84%	45415	50.0

As the number of pairings generated per iteration increases, the total cost decreases as expected. However, the calculation time also increases dramatically. The total calculation time before convergence when n is 20,000 is about 318.6 hours, which is about 12 continuous days. With the memory usage occasionally above 50 megabytes, running the heuristic when n is 20,000 requires a dedicated machine for efficient performance. Increasing the number of pairings per iteration for the heuristic further would eventually require more random access memory or a more efficient method of data storage in order to generate a converged solution.

While significantly more time and memory resouces are required to solve this problem with the larger population size, the results provide a larger savings in cost as well. For longer-term crew scheduling, these results may be preferred, despite the longer calcalution time.

Figure 5 - 3. Improvement in ACN Total Cost for n = 20,000



5.3 EFN Results

5.3.1 EFN Results for n = 2,000

The general behavior of the heuristic results for the EFN data when the population size is limited to 2,000 is largely the same as that of the ACN data. The final solution of $209,378 dollar per week was almost 12 percent better than the original total cost. Over one year, this solution provides a savings of $1,477,008 dollars per year. Compared to the ACN problem, these EFN results have a greater reduction in cost.

The heuristic for the EFN data with a 2,000 population limit converges at iteration 13. As with the previous results, there is steady improvement through each iteration. After iteration 13, no cost reduction is observed.

Table 5 - 8. Improvement in EFN Total Cost for n = 2,000

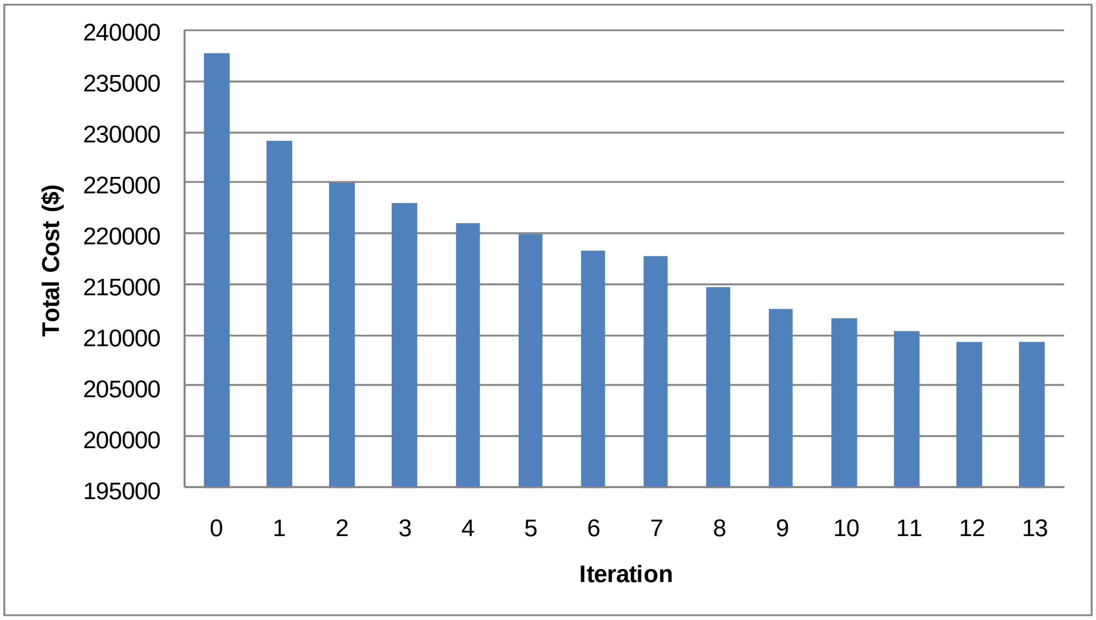
Iteration	*Total Cost ($)*	*Improvement (%)*	*Calculation Time (sec)*	*Memory Usage (MB)*
0	237782	NA	NA	NA
1	229224	3.60%	4830	2.1
2	225144	5.31%	3787	3.7
3	223059	6.19%	1446	1.3
4	221095	7.02%	5561	3.3
5	220022	7.47%	3295	1.5
6	218381	8.16%	9351	1.8
7	217917	8.35%	3883	1.1
8	214853	9.64%	4534	2.0
9	212645	10.57%	7448	2.9
10	211725	10.96%	2586	2.1
11	210513	11.47%	1045	3.3
12	209378	11.95%	4463	2.9
13	209378	11.95%	1225	2.3

The total calculation time for convergence was about 14.8 hours. Although the EFN problem, with 1,044 duties, is the smallest of the four partitioned problems, the calculation time is longer than that for the corresonding ACN problem. The longer calculation time may result from the less optimized EFN crew schedule. Since there is more room for improvement, the heuristic uses more time to find a better solution.

Memory usage for the EFN problem with a 2,000 population size limit is also similar to the ACN results. The memory usage varied based on the average length of the pairings generated, but did not show a consistent trend from iteration to iteration.

The memory usage generally remained relatively low, and there were no issues with memory shortage througout the solution process.

Figure 5 - 4. Improvement in EFN Total Cost for n = 2,000



5.3.2 EFN Results for n = 4,000

As the population size limit is increased to 4,000 for the EFN problem, the cost reduction increases. The new final solution is $188,925 dollars per week, which is a 21.0% improvement over the original solution. The corresponding annual savings is $2,540,564 dollars per year. This cost savings is significantly more than that of the corresponding ACN problem, which again suggest that the original EFN problem was less optimized.

The solution for this problem reached convergence after 15 iterations. Again, the improvements are steady over each iteration, with larger improvements initially and smaller improvemntes until convergence.

The calculation time for this problem was approximately 42.6 hours, over twice the calculation time for the the same problem with a 2,000 population size limit.

The relatively long calculation time would not be appropriate for short-term crew scheduling and planning. However, this calculation time may be acceptable for longer-term scheduling decisions.

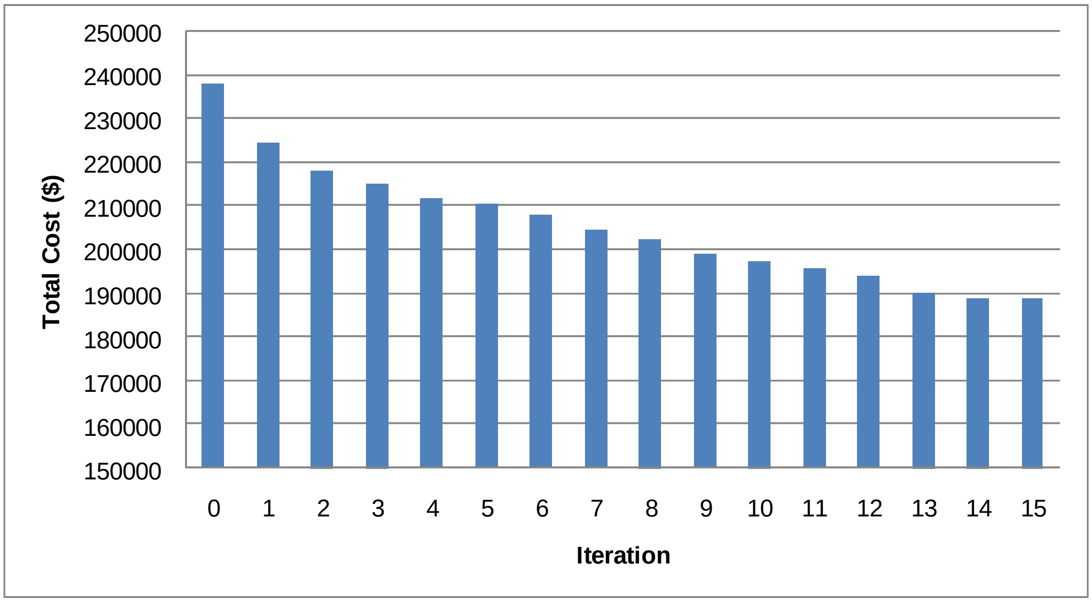
The memory usage for the EFN problem with a 4,000 pairing limit ranged from 1.1 to 10.9 megabytes per iteration. Memory usage was again not as issue for the computer throughout the solution process.

Table 5 - 9. Improvement in EFN Total Cost for n = 4,000 pairings

Iteration	***Total Cost ($)***	***Improvement (%)***	***Calculation Time (sec)***	***Memory Usage (MB)***
0	237782	NA	NA	NA
1	224511	5.58%	8752	4.8
2	218177	8.24%	15240	5.8
3	214955	9.60%	9102	3.3
4	211922	10.88%	12574	3.0
5	210254	11.58%	8786	5.6
6	207699	12.65%	10438	3.0
7	204605	13.95%	8675	4.4
8	202239	14.95%	7220	3.5
9	198811	16.39%	14382	1.7
10	197375	16.99%	7628	4.7
11	195500	17.78%	8819	4.0
12	193734	18.52%	10836	1.1
13	190253	19.99%	12397	1.1
14	188929	20.55%	11103	10.9
15	188925	20.55%	7428	10.8

The heuristic found the solution with a 12 percent reduction in total cost by the sixth iteration and within 18 hours. In comparison the problem with a 2,000 pairing limit found the solution with a 12 percent reduction in approximately 15 hours. However, the larger limit allowed for more iterations before convergence, and ultimately a 20.6 percent improvement. These results suggest that while the larger population size limit requires a longer calculation time and more intensive memory usage, the larger set of generated pairings results in a better final solution.

Figure 5 - 5. Improvement in EFN Total Cost for n = 4,000



5.3.3 EFN Results for n = 20,000

As was the case for the ACN problem, the final solution for the EFN problem with a 20,000 population size limit provides the largest cost reduction. The total improvement over the original cost is 25.7 percent. The final solution for this problem is $176,781 per week, which results in $3,172,052 dollars saved per year.

Convergence of the solution for this problem was reached within 15 iterations. There was again steady improvement over each iteration, with larger improvements initially and smaller improvements until convergence.

The calculation time for this problem was approximately 330.9 hours. The longer calculation time suggests that because more pairings need to be generated and stored, the heuristic also finds many more infeasible pairings. An examination of the pairings created during the solution process confirms that many more infeasible pairings are created and removed when the population size limit is larger. This long calculation time would not be appropriate for short-term crew scheduling, but may still be acceptable for long-range scheduling.

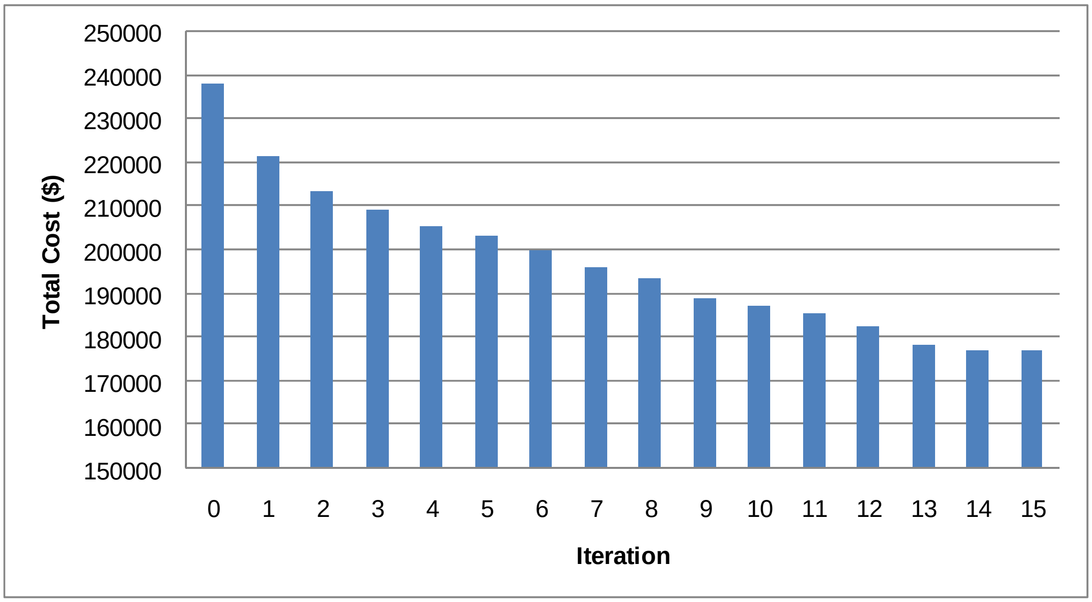
The memory usage in calculating the solution for this EFN problem with a 20,000 population size limit was not as intenstive as that of the correspnding ACN problem. The smaller memory usage may be a result of the smaller problem size.

Table 5 - 10. Improvement in EFN Total Cost for n = 20,000

Iteration	*Total Cost ($)*	*Improvement (%)*	*Calculation Time (sec)*	*Memory Usage (MB)*
0	237782	NA	NA	NA
1	221545	6.83%	68819	18.6
2	213360	10.27%	42877	11.3
3	209278	11.99%	73045	7.9
4	205482	13.58%	73144	13.9
5	203381	14.47%	59625	10.5
6	199885	15.94%	88067	7.1
7	195849	17.64%	106279	18.6
8	193622	18.57%	81091	21.4
9	188764	20.61%	54666	12.6
10	187140	21.30%	107976	10.6
11	185302	22.07%	109732	20.1
12	182354	23.31%	63469	14.1
13	178233	25.04%	117309	18.6
14	176781	25.65%	92790	8.1
15	176781	25.65%	52471	16.7

Increasing the population size limit from 2,000 to 4,000 resulted in a total cost improvement from about 12 percent to 21 percent. However, further increasing the population size limit to 20,000 only increased the improvement in total cost to about 26 percent. This problem also found the final solution for the 2,000 population limit problem within four iterations and the solution for the 4,000 population limit problem within 10 iterations. These solutions also took longer to find using this larger population size limit, which is due to the large number of pairings that must be generated and stored.

Figure 5 - 6. Improvement in EFN Total Cost for n = 20,000



5.4 ACS Results

5.4.1 ACS Results for n = 2,000

The results for the ACS problem solved with a population size, n, of 2,000 are shown in Table 5-7 and Figure 5-7. The final solution for the ACN problem was a 3.02 percent improvement over the original cost of $631,892 dollars per week. This savings corresponds to an annual savings of $991,952 dollars per year. The improvement for this ACS problem is smaller than that of the corresponding ACN problem, which suggests that the AC schedule in the northern partition has more potential for improvement. These results also imply that the original AC crew schedule for the southern partition was less optimized.

As with all previous results, the improvements in total cost increase over each iteration. This problem converged at the sixteenth iteration. There was also near convergence between the sixth and seventh iteration, which may suggest that there may be further improvements if the heuristic continues to run past the defined convergence criterion.

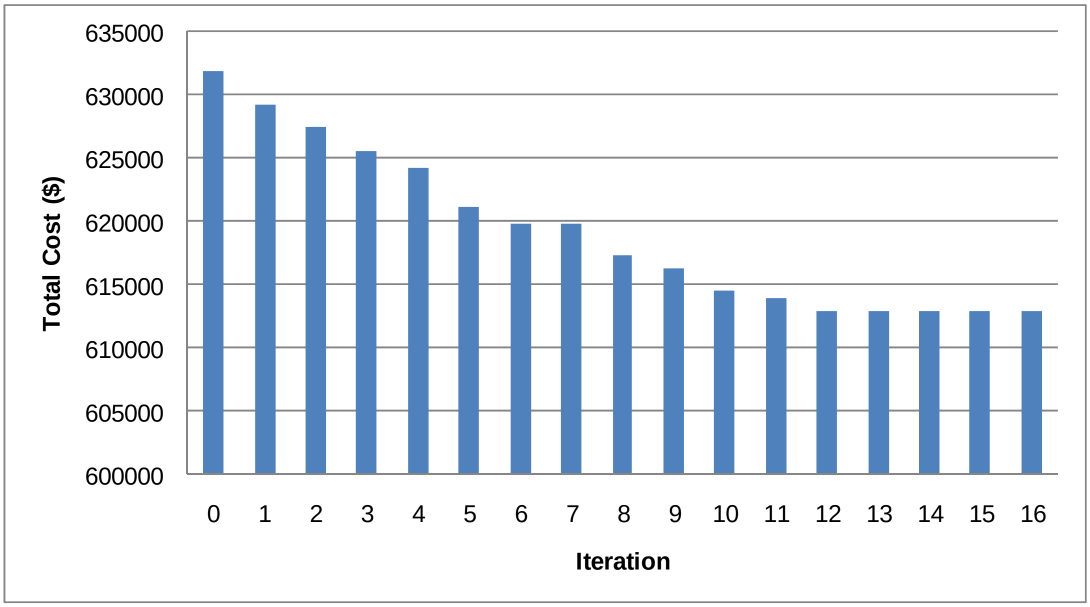
The calculation time for this problem was approximately 18.8 hours, longer than the 9.72 hours for the ACN problem. Although this problem was smaller in size, it may have required a longer calculation time because there was less potential for improvement, and the heuristic spent more time searching for new, feasible pairings.

The memory usage for the ACS problem with a 2,000 pairing limit ranged from 4.0 to 5.7 megabytes per iteration. Memory usage was again not as issue for the computer throughout the solution process.

Table 5 - 11. Improvement in ACS Total Cost for n = 2,000

Iteration	***Total Cost ($)***	***Improvement (%)***	***Calculation Time (sec)***	***Memory Usage (MB)***
0	631892	NA	NA	NA
1	629204	0.43%	4324	5.2
2	627457	0.70%	4546	5.4
3	625612	0.99%	4174	4.4
4	624272	1.21%	4397	5.5
5	621120	1.70%	3853	5.5
6	619841	1.91%	4580	5.6
7	619750	1.92%	4229	5.7
8	617240	2.32%	3915	4.5
9	616257	2.47%	4620	5.3
10	614544	2.75%	4410	5.6
11	613912	2.85%	4337	5.4
12	612902	3.01%	4143	4.5
13	612882	3.01%	3568	5.1
14	612816	3.02%	4134	4.1
15	612816	3.02%	4303	4.0
16	612816	3.02%	4303	4.0

Figure 5 - 7. Improvement in ACS Total Cost for n = 2,000



5.4.2 ACS Results for n = 4,000

As the population size limit is increased to 4,000 for the ACS problem, the final solution becomes $580,140 dollars per week. This solution is an improvement of 8.2 percent, or $2,691,104 dollars saved per year. This cost savings is more than that of the corresponding ACN problem, which suggests that the original ACS problem was less optimized.

The solution for this problem reached convergence after 15 iterations. Again, the improvements are steady over each iteration, with larger improvements initially and smaller improvements until convergence.

The calculation time for this problem was approximately 34.2 hours, nearly twice the calculation time for the the same problem with a 2,000 population size limit. The relatively long calculation time would not be appropriate for short-term crew scheduling and planning. However, this calculation time may be acceptable for long-term scheduling.

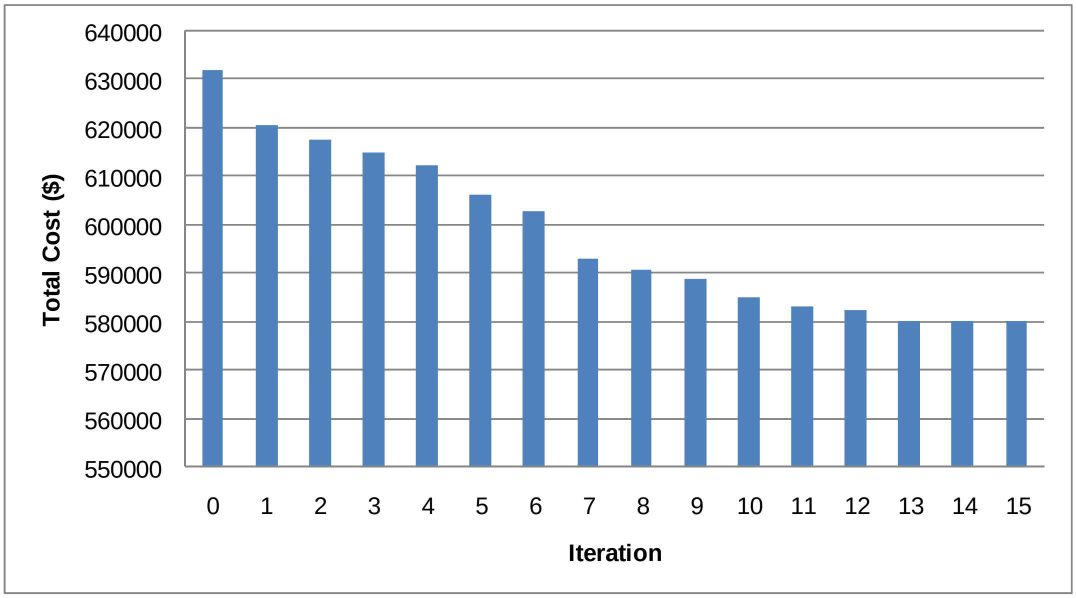
Table 5 - 12. Improvement in ACS Total Cost for n = 4,000

Iteration	*Total Cost ($)*	*Improvement (%)*	*Calculation Time (sec)*	*Memory Usage (MB)*
0	631892	NA	NA	NA
1	620518	1.80%	9293	10.4
2	617738	2.24%	9174	10.8
3	615020	2.67%	8450	8.6
4	612240	3.11%	9495	11.1
5	606174	4.07%	7747	11.1
6	602699	4.62%	9922	11.1
7	592778	6.19%	8739	11.4
8	590629	6.53%	8412	8.8
9	588860	6.81%	8990	10.7
10	584879	7.44%	9689	11.2
11	582920	7.75%	8756	10.7
12	582415	7.83%	8557	8.9
13	580140	8.19%	7645	10.2
14	580140	8.19%	8218	8.3
15	580140	8.19%	8535	8.3

The memory usage for the EFN problem was more intensive than the same problem with a 2,000 pairing limit. The memory usage ranged from 8.3 to 11.2 megabytes per iteration. Although the memory usage was higher, it was not an issue for the test computer throughout the solution process.

The heuristic found the solution from the 2,000 pairing limit problem within four iterations and in 34.2 hours. This calculation time was faster than that of the 2,000 pairing limit problem but required more memory usage. These results may have been because the heuristic created more “good” feasible pairings in a shorter amount of time.

Figure 5 - 8. Improvement in ACS Total Cost for n = 4,000



5.4.3 ACS Results for n = 20,000

The final solution for the ACS problem with a 20,000 population size limit provides the largest cost reduction. The total improvement over the original cost is 18 percent. The final solution for this problem is $518,467 per week, which results in $5,898,100 dollars saved per year.

Convergence of the solution for this problem was reached within 13 iterations. There was again steady improvement over each iteration, with larger improvements initially and smaller improvements until convergence.

The calculation time for this problem was approximately 62.7 hours. The longer calculation time seems to confirm that because more pairings need to be generated and stored, the heuristic also finds many more infeasible pairings. The heuristic spends more time creating and removing these infeasible pairings. This long calculation time would not be appropriate for short-term crew scheduling, but may still be acceptable for long-range scheduling.

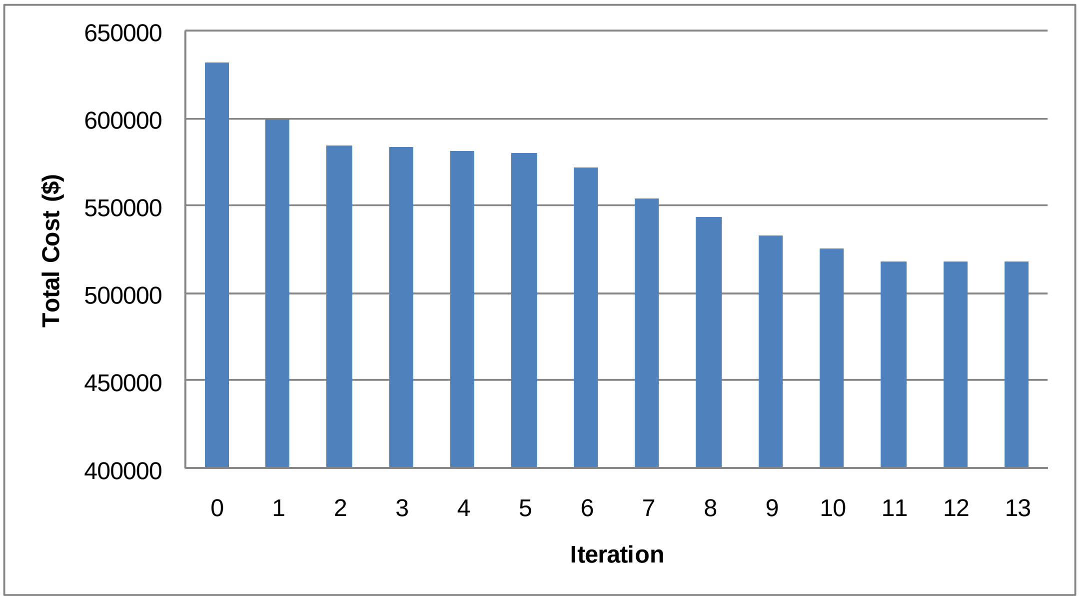
Table 5 - 13. Improvement in ACS Total Cost for n = 20,000

Iteration	*Total Cost ($)*	*Improvement (%)*	*Calculation Time (sec)*	*Memory Usage (MB)*
0	631892	NA	NA	NA
1	598907	5.22%	20464	21.0
2	584311	7.53%	12289	16.5
3	582857	7.76%	18205	27.5
4	580772	8.09%	16110	30.0
5	580077	8.20%	14237	25.5
6	571862	9.50%	16275	19.0
7	553980	12.33%	9203	19.5
8	543048	14.06%	14153	24.0
9	532748	15.69%	17548	24.0
10	525608	16.82%	16656	18.5
11	518467	17.95%	23549	31.0
12	518467	17.95%	23549	25.0
13	518467	17.95%	23549	25.0

The memory usage in calculating the solution for this EFN problem with a 20,000 population size limit was not as intenstive as that of the correspnding ACN problem. The smaller memory usage may be a result of the smaller problem size.

Increasing the population size limit from 2,000 to 4,000 resulted in a total cost improvement from about three percent to eight percent. Further increasing the population size limit to 20,000 resulted in a larger improvement in total cost of about 18 percent. This problem also found the final solution for the 2,000 population limit problem within one iteration and the solution for the 4,000 population limit problem within four iterations. These solutions did take longer to find using this larger population size limit, which is due, in part, to the large number of pairings that must be generated and stored.

Figure 5 - 9. Improvement in ACS Total Cost for n = 20,000



5.5 EFS Results

5.5.1 EFS Results for n = 2,000

The EFS problem with a limit of 2,000 pairings had a reduction in total cost of 10.2 percent. The final solution was $405,200 dollars per week, equivalent to $2,387,840 dollars saved per year. The improvement for this EF problem was again greater than that for the corresponding AC problem. These results seem to confirm that there is more room for improvement for the EF CSPs.

The solution for this problem converged in thirteen iterations. As was the case for the other problem runs, there were larger improvements initially and steady cost reductions until convergence.

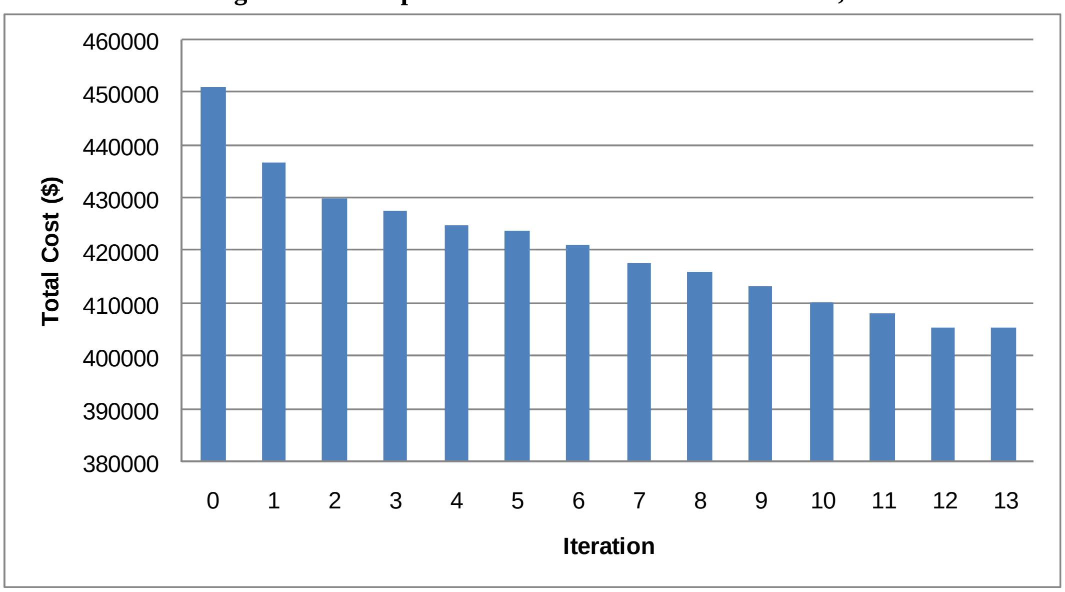
The calculation time for this problem was about 11.2 hours. Although relatively fast compared to the larger problems, this calculation time would only be appropriate for longer-term crew scheduling.

The memory usage for this EFS problem ranged from 2.0 to 3.1 megabytes per iteration. These memory usages are relatively low, and there were no memory issues during the running of these problems.

Table 5 - 14. Improvement in EFS Total Cost for n = 2,000

Iteration	*Total Cost ($)*	*Improvement (%)*	*Calculation Time (sec)*	*Memory Usage (MB)*
0	451120	NA	NA	NA
1	436859	3.16%	2733	2.6
2	429811	4.72%	2559	2.6
3	427455	5.25%	3843	2.4
4	424801	5.83%	614	2.7
5	423703	6.08%	3380	2.6
6	421195	6.63%	4732	2.5
7	417493	7.45%	1018	2.9
8	415738	7.84%	5797	3.1
9	413269	8.39%	6177	2.4
10	409951	9.13%	2775	2.2
11	408127	9.53%	378	2.0
12	405200	10.18%	4121	2.3
13	405200	10.18%	2015	2.9

Figure 5 - 10. Improvement in EFS Total Cost for n = 2,000



The EFS problem has 1,049 duties, similar to the problem sizes of the EFN and ACS problems, with 1,044 and 1,067 duties, respectively. The results for this problem had a larger improvement and shorter calculation time the ACS problem. This problem also had a similar, although smaller, improvement and shorter calculation time than the EFN problem. These results may suggest that there is larger potential for improvement for this EFS problem.

5.5.2 EFS Results for n = 4,000

As the population size limit is increased to 4,000 for the EFS problem, the final solution becomes $368,584 dollars per week. This solution is an improvement of 18.3 percent, or $4,291,872 dollars saved per year. This cost savings is more than that of the corresponding ACS problem, which suggests that the original EFS problem was less optimized.

The solution for this problem reached convergence after 11 iterations. Again, the improvements are steady over each iteration, with larger improvements initially and smaller improvements until convergence.

The calculation time for this problem was approximately 34.2 hours, nearly twice the calculation time for the the same problem with a 2,000 population size limit. The relatively long calculation time would not be appropriate for short-term crew scheduling and planning. However, this calculation time may be acceptable for long-term scheduling.

The memory usage for the EFS problem was more intensive than the same problem with a 2,000 pairing limit. The memory usage ranged from 4.5 to 6.5

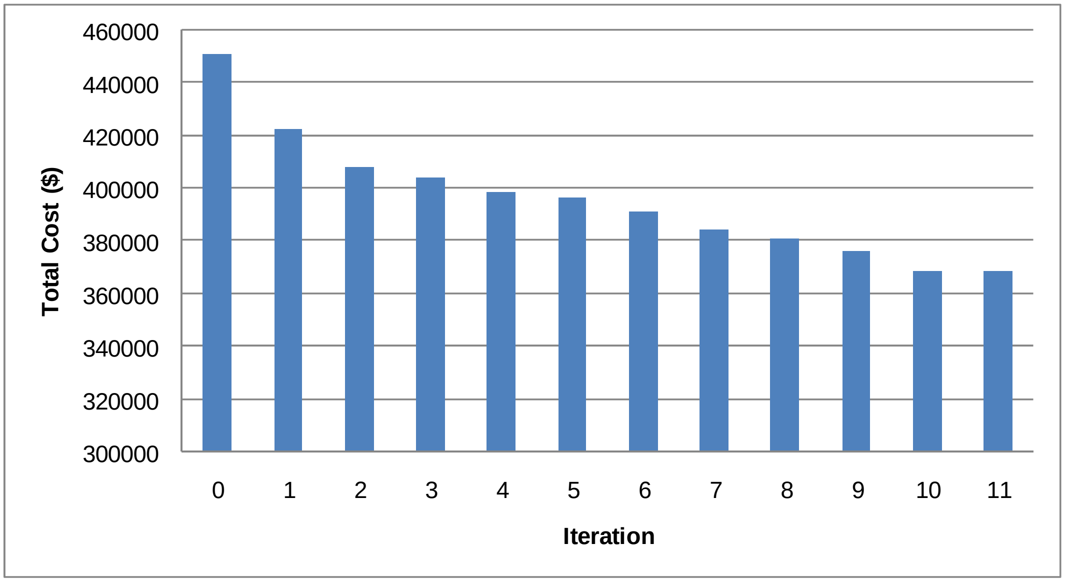
megabytes per iteration. Although the memory usage was higher, it was not an issue for the computer throughout the solution process.

Table 5 - 15. Improvement in EFS Total Cost for n = 4,000

Iteration	*Total Cost ($)*	*Improvement (%)*	*Calculation Time (sec)*	*Memory Usage (MB)*
0	451120	NA	NA	NA
1	422493	6.35%	8689	5.0
2	407980	9.56%	5348	5.7
3	403855	10.48%	5107	5.6
4	398430	11.68%	5132	6.2
5	396458	12.12%	11350	5.5
6	390862	13.36%	8868	5.2
7	383859	14.91%	9737	5.8
8	380685	15.61%	5145	6.5
9	376065	16.64%	11090	4.8
10	368584	18.30%	6684	4.5
11	368584	18.30%	11520	4.7

The heuristic found the solution from the 2,000 pairing limit problem within three iterations and in 5.31 hours. This calculation time was faster than that of the 2,000 pairing limit problem but required less memory usage. These results suggest that this problem is a good candidate for crew schedule improvements.

Figure 5 - 11. Improvement in EFS Total Cost for n = 4,000



5.5.3 EFS Results for n = 20,000

The final solution for the EFS problem with a 20,000 population size limit provides the largest cost reduction. The total improvement over the original cost is 27.9 percent, the largest improvement for all of the partitioned problems. The final solution for this problem is $325,212 per week, which results in $6,547,216 dollars saved per year.

Convergence of the solution for this problem was reached within 14 iterations. There was again steady improvement over each iteration, with larger improvements initially and smaller improvements until convergence.

The calculation time for this problem was approximately 86.3 hours, which is shorter than that of the corresponding north partition problems, but longer than the ACS problem with the 20,000 population size limit.

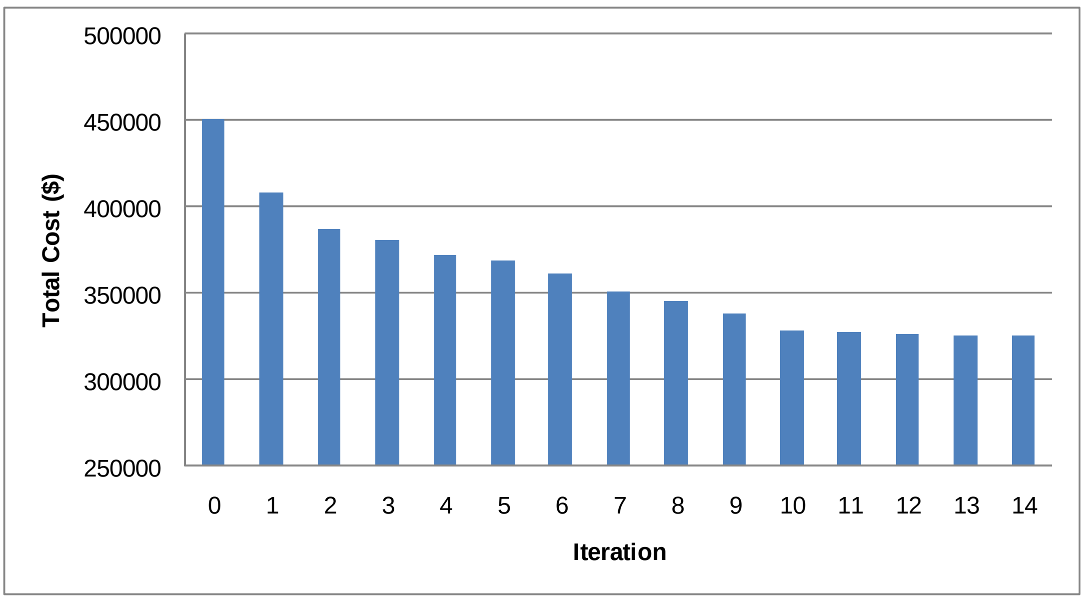
The memory usage in calculating the solution for this EFN problem with a 20,000 population size limit was not as intenstive as that of the correspnding ACN problem. The smaller memory usage may be a result of the smaller problem size.

Increasing the population size limit from 2,000 to 4,000 resulted in a total cost improvement from about 10 percent to 18 percent. Further increasing the population size limit to 20,000 resulted in a larger improvement in total cost of about 28 percent. This problem also found the final solution for the 2,000 population limit problem within two iterations and the solution for the 4,000 population limit problem within five iterations. These solutions did take longer to find using this larger population size limit, which is again due, in part, to the large number of pairings that must be generated and stored.

Table 5 - 16. Improvement in EFS Total Cost for n = 20,000

Iteration	*Total Cost ($)*	*Improvement (%)*	*Calculation Time (sec)*	*Memory Usage (MB)*
0	451120	NA	NA	NA
1	408132	9.53%	20454	37.6
2	386471	14.33%	18604	40.3
3	380637	15.62%	20197	32.1
4	372106	17.51%	12211	41.6
5	368414	18.33%	25964	37.4
6	361112	19.95%	21737	43.8
7	349903	22.44%	24118	41.3
8	345466	23.42%	17906	44.6
9	337987	25.08%	33731	30.7
10	327850	27.33%	17173	27.3
11	326759	27.57%	20907	32.2
12	325702	27.80%	22562	34.1
13	325212	27.91%	30019	39.3
14	325212	27.91%	25010	34.1

Figure 5 - 12. Improvement in EFS Total Cost for n = 20,000



5.6 Sensitivity Analysis

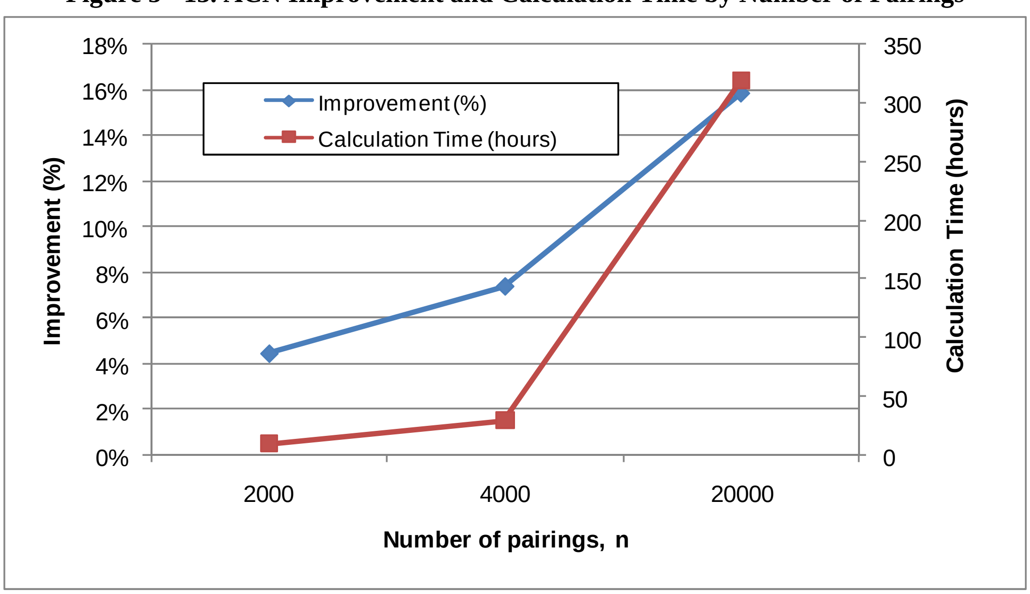
5.6.1 Description of Results

The memory usage graphs in this section suggest that as the pairings per iteration increase, memory usage and calculation time all increase as well. Further increasing the number of pairings per iteration would cause the calculation times to exceed weeks from start to convergence. While calculation times may be irrelevant in practical, long-term application of a final solution, memory usage could be a realistic limitation because stored pairings would fill the random access memory of a typical computer. Any more data that is stored would then need to be accessed from the hard disk, significantly increasing the calculation time of the program.

Table 5 - 17. Comparison of ACN Results by Number of Pairings

n	*Iterations*	*Total Cost ($)*	*Improvement (%)*	*Calculation Time (sec)*	*Memory Usage (MB)*
2000	13	407868	4.45%	35011	61.3
4000	15	395309	7.40%	105537	148.6
20000	14	359257	15.84%	1146799	341.5

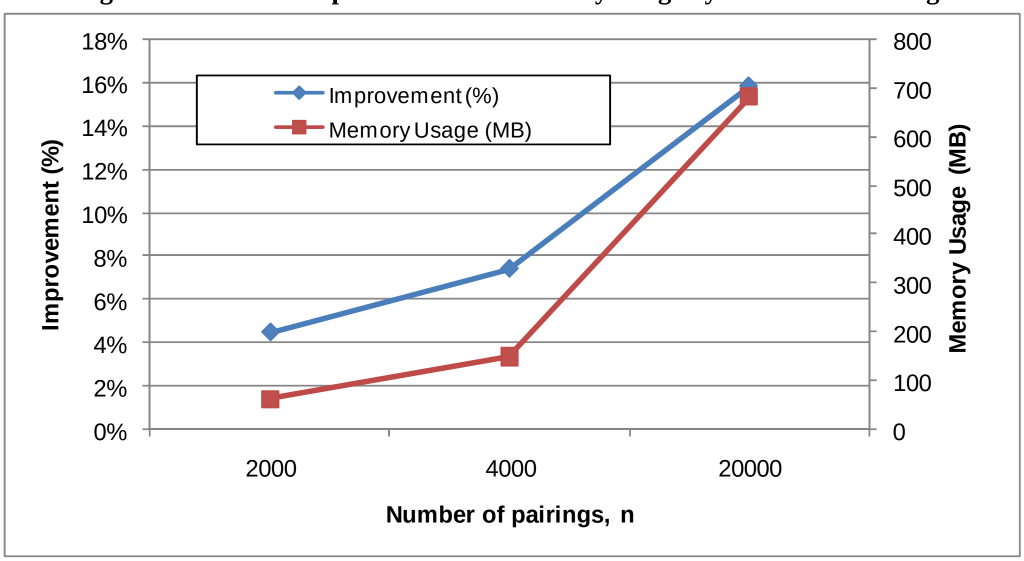
Figure 5 - 13. ACN Improvement and Calculation Time by Number of Pairings



Each of the different data sets behave somewhat differently as pairings per iteration increases. The ACN results in Table 5-17 and Figure 5-13 show that when the number of pairings is doubled from 2,000 to 4,000, the improvement in total cost is moderate. When the pairing limit is further increased by tenfold to 20,000, the improvement is less. As the improvement increases, the calculation time increases at a faster rate. If calculation time is not of concern, it appears an appropriate limit on the number of pairings is approximately 20,000 pairings.

As the number of pairings is doubled, the memory usage also has a moderate increase (Figure 5-14). However, as the number of pairings increases to 20,000, the graph shows that memory usage also increases at a faster rate than the improvement rate. An appropriate limit on the population size is also approximately 20,000 based on the memory usage results. Comparing Figures 5-13 and 5-14, it appears that the calculation time increases faster than the memory usage.

Figure 5 - 14. ACN Improvement and Memory Usage by Number of Pairings

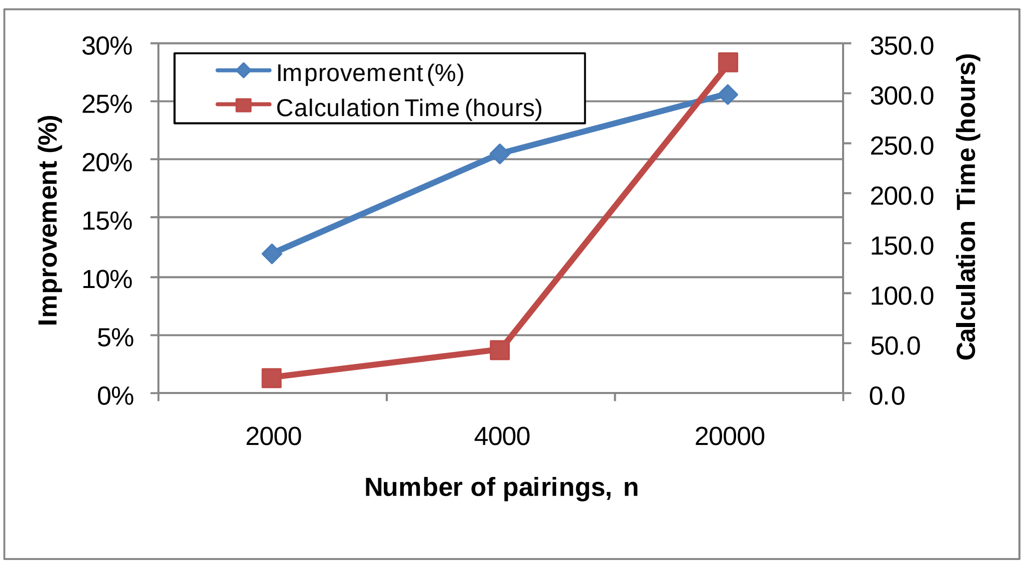


The results for the EFN problems are summarized in Table 5-18. The rate of improvement for the EFN problem is less than that for the ACN problem (Figure 5-15). Again, the calculation time increases at a faster rate than the improvement rate. Figure 5-15 suggests that a balance between better improvement in total cost and the calculation time is approximately 16,000 pairings.

Table 5 - 18. Comparison of EFN Results by Number of Pairings

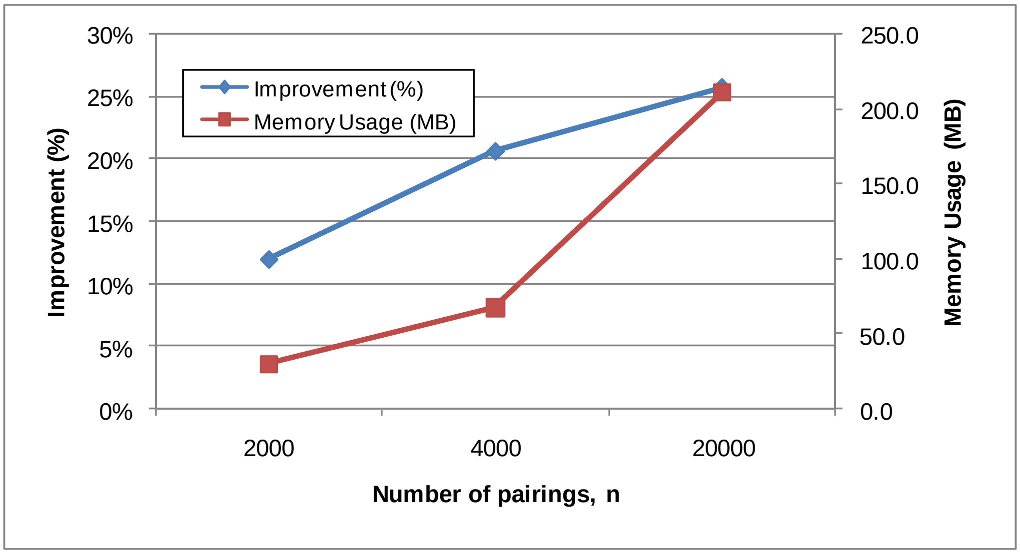
n	*Iterations*	*Total Cost ($)*	*Improvement (%)*	*Calculation Time (sec)*	*Memory Usage (MB)*
2000	13	209378	11.95%	53454	30.2
4000	15	188925	20.55%	153380	67.7
20000	15	176781	25.65%	1191360	210.1

Figure 5 - 15. EFN Improvement and Calculation Time by Number of Pairings



Similar to the EFN results for the calculation time, the EFN memory usage also increases at a faster rate than the rate of improvement. Also, similar to the ACN results, the increase in memory usage appears to be slower than the increase in calculation time as the population size limit increases. Figure 5-16 shows that a balance between the increased improvement and increased memory usage uses about 20,000 pairings as the population size limit.

Figure 5 - 16. EFN Improvement and Memory Usage by Number of Pairings

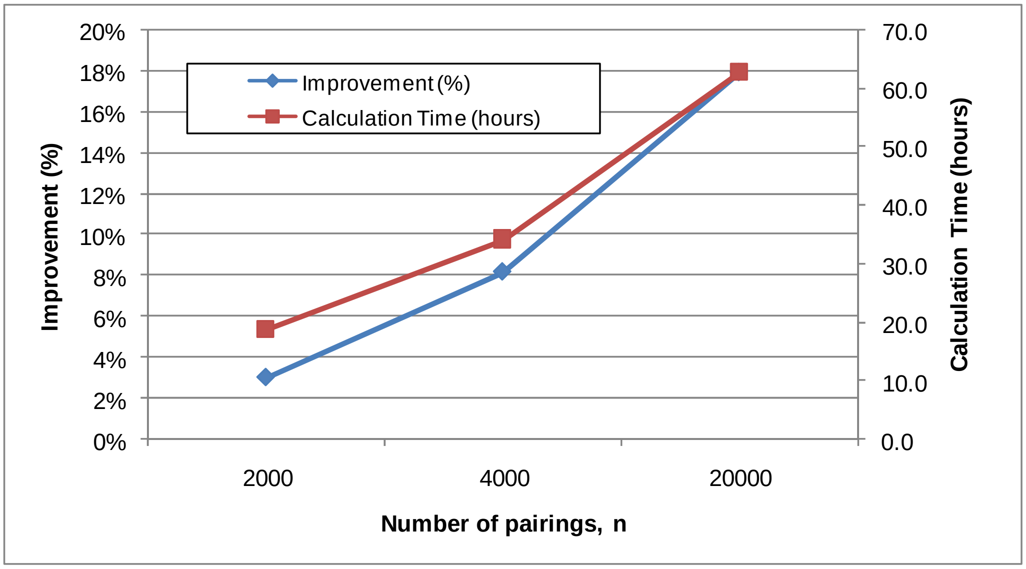


The ACS problem results are summarized in Table 5-19. As the population size limit increases, the calculation time for the ACS problem also increases (Figure 5-17). The rates of the calculation time and improvement increases are more similar for this problem. An appropriate balance between the better improvement in total cost and the longer calculation time appears occur when the population size is set at about 20,000 pairings.

Table 5 - 19. Comparison of ACS Results by Number of Pairings

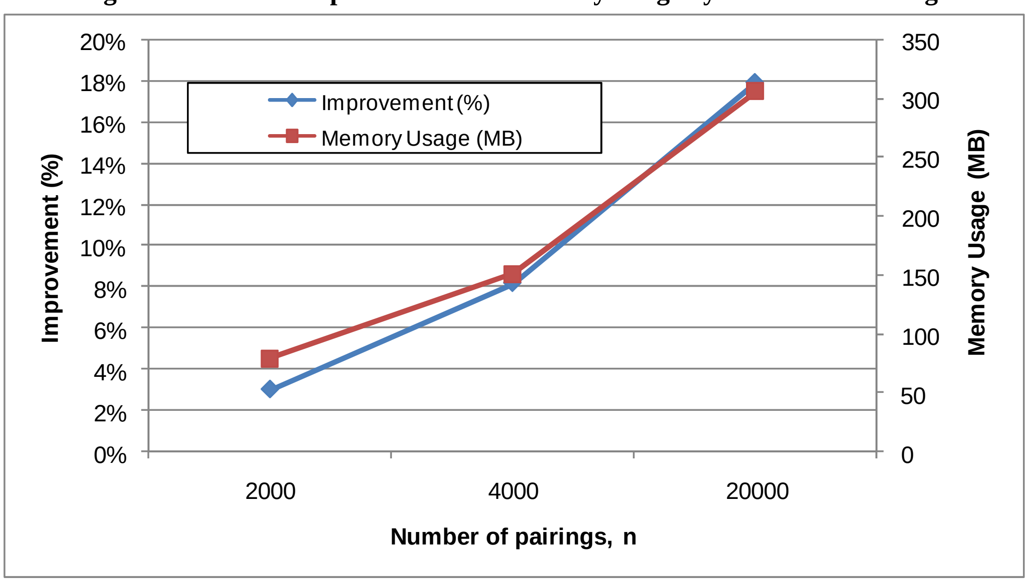
n	*Iterations*	*Total Cost ($)*	*Improvement (%)*	*Calculation Time (sec)*	*Memory Usage (MB)*
2000	16	612816	3.02%	67836	79.8
4000	15	580140	8.19%	123087	151.6
20000	13	518467	17.95%	225789	306.5

Figure 5 - 17. ACS Improvement and Calculation Time by Number of Pairings



Similarly, as the population size increases, the rate of increased memory usage is faster than the rate of better improvements in total cost (Figure 5-18). However, unlike for the ACN and EFN problems, the increase in memory usage appears to be faster than that of the calculation time. The balance of improvement increase and memory usage increase uses a population size limit of approximately 10,000 pairings.

Figure 5 - 18. ACS Improvement and Memory Usage by Number of Pairings

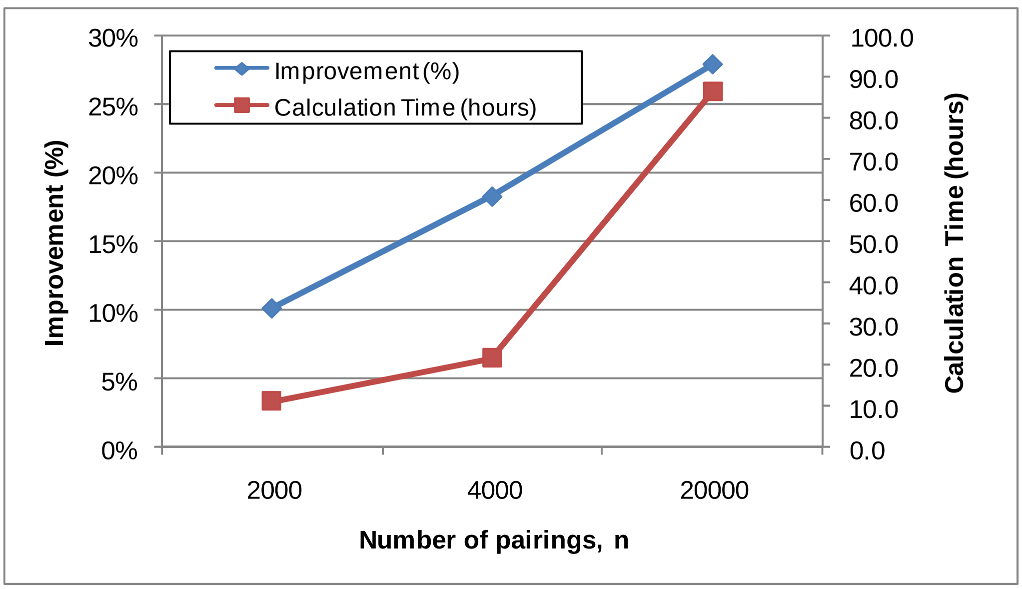


The EFS problem results are summarized in Table 5-20. For this problem, the calculation time also increases faster than the improvement when the population size is increased (Figure 5-19). The point that balances the improvement increase and calculation time increase seems to occur when the limit on the population size is above 20,000 pairings. This is the highest of all four partitioned problems.

Table 5 - 20. Comparison of EFS Results by Number of Pairings

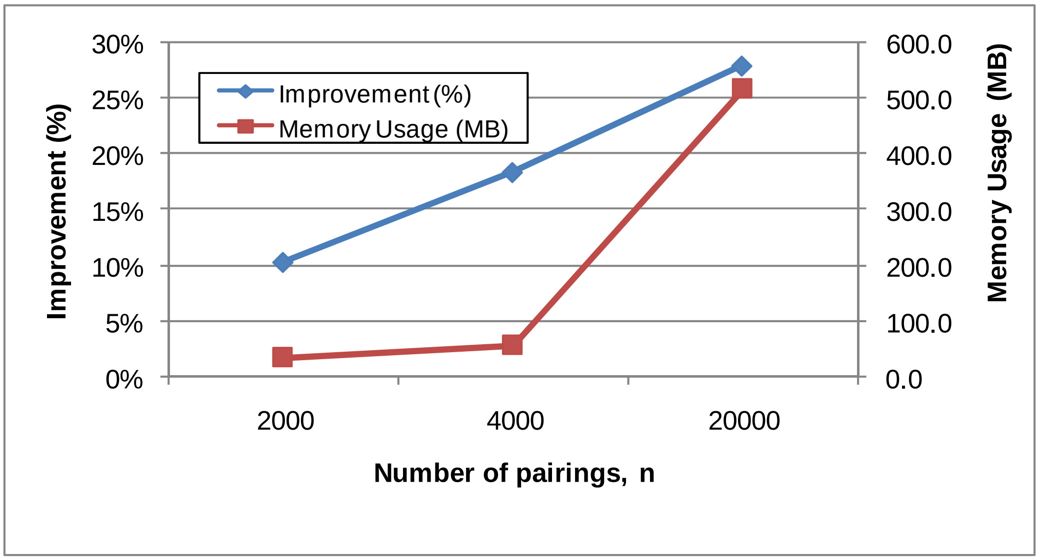
n	*Iterations*	*Total Cost ($)*	*Improvement (%)*	*Calculation Time (sec)*	*Memory Usage (MB)*
2000	13	405200	10.18%	40140	33.2
4000	11	368584	18.30%	77150	54.7
20000	14	325212	27.91%	310592	516.4

Figure 5 - 19. EFS Improvement and Calculation Time by Number of Pairings



The EFS problem memory usage also increases at a faster rate than the improvement as population size increases (Figure 5-20). The memory usage also appears to increase faster than the calculation time. As with the calculation time results, an appropriate limit on the population size based on the memory usage data is above 20,000 pairings.

Figure 5 - 20. EFS Improvement and Memory Usage by Number of Pairings



5.6.2 Population Size

As previously discussed in this section, the improvement in total cost, calculation time, and memory usage all vary with the population size, n. As the population size limit increases, the CSP becomes more difficult to solve due to several reasons. First, increasing the population size makes it more difficult to find feasible pairings. Feasible pairings are those pairings that do not violate any of the problem constraints and that are not already in the restricted subset of pairings. Since many “good” pairings are likely to have been found and have entered the restricted subset in later iterations, it becomes increasingly harder to find new pairings to enter to subset.

Second, a larger population size means that the restricted subset is also larger. The final integer program that must be solved by CPLEX has more pairings, which in turn, results in longer calculation times. In all four partitioned problems, the calculation time increased as the population size increased (Tables 5-17, 5-18, 5-19, and 5-20).

Third, since more pairings must be stored and the larger integer programs are more difficult to solve, the overall computer program requires more intensive memory usage. Again, the results from all four partitioned problems indicated increasing memory usage as the population size increased (Tables 5-17, 5-18, 5-19, and 5-20).

Although increasing the population size makes the CSPs more difficult to solve, the improvement of the solutions also improve. The ACN results suggest that the population size should be set at approximately 20,000 pairings. The EFN and ACS results show that the population size may be set at about 18,000 and 15,000 pairings, respectively. Lastly, the EFS results suggest using a population limit above 20,000 pairings. There does not appear to be a strong relationship between the number of duties and a good population size limit. It is possible that the population size depends on the existing crew schedule and how much potential there is for improvement. Based on the problems in this thesis, it is difficult to generalize what population size is appropriate for CSPs other than those studied.

5.7 Complete Network Results

The complete network results combine the north and south partitions into a single network. The crew type partitions of EF and AC crew remain relevant. As a result, the two separate CSPs were solved for these sets of crew types. The heuristic was applied to each problem with a population size of 2,000 pairings. Increasing the population size to 3,000 and 4,000 rendered the problems unsolvable using the proposed heuristic. The computer program stalled due to memory shortage issues each time the heuristic was applied with these larger population size limits.

The final solution of the AC problem provided a 2.85 percent improvement over the original total cost (Table 5-21, Figure 5-21). This improvement was less than those for the four partitioned problems. This problem required longer calculation time and more intensive memory usage.

The final solution of the EF problem was a 3.0 percent improvement over the original total cost (Table 5-22, Figure 5-22). This improvement was also less than those for the four partitioned problems, and the problem required longer calculation time and more intensive memory usage.

Table 5 - 21. Improvement in AC Total Cost for n = 2,000 pairings

Iteration	*Total Cost ($)*	*Improvement (%)*	*Calculation Time (sec)*	*Memory Usage (MB)*
0	1110099	NA	NA	NA
1	1107361	0.25%	72193	47.2
2	1107209	0.26%	63493	46.9
3	1106776	0.30%	84883	50.1
4	1105041	0.46%	73043	58.0
5	1102586	0.68%	83493	54.9
6	1099312	0.98%	105739	55.1
7	1096907	1.20%	96953	78.0
8	1094468	1.43%	87030	65.4
9	1090529	1.79%	95673	57.6
10	1086357	2.18%	97756	62.3
11	1085850	2.23%	10765	66.9
12	1082909	2.50%	92531	67.7
13	1079799	2.80%	74958	78.5
14	1079302	2.85%	75798	64.9
15	1079316	2.85%	86797	53.5

Figure 5 - 21. Improvement in AC Total Cost for n = 2,000

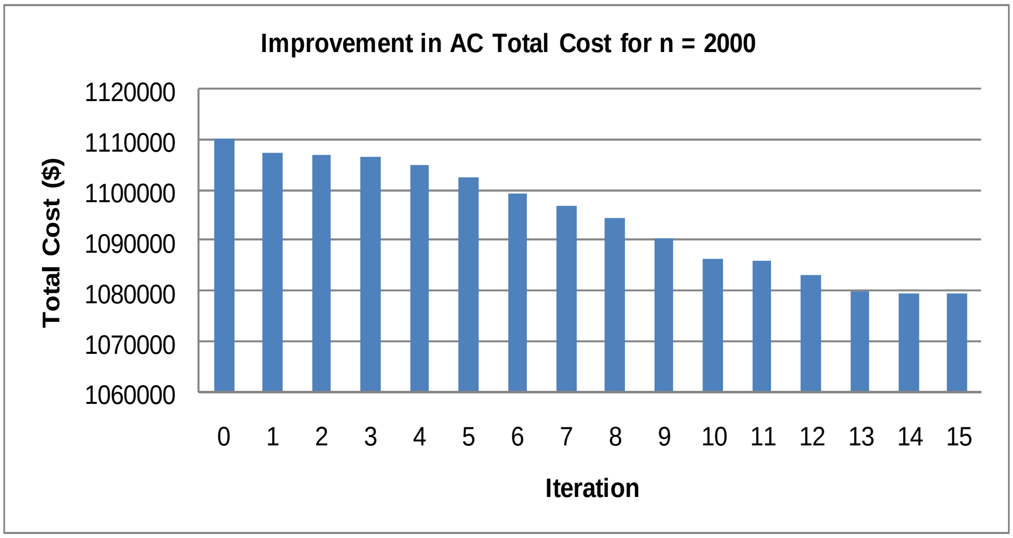
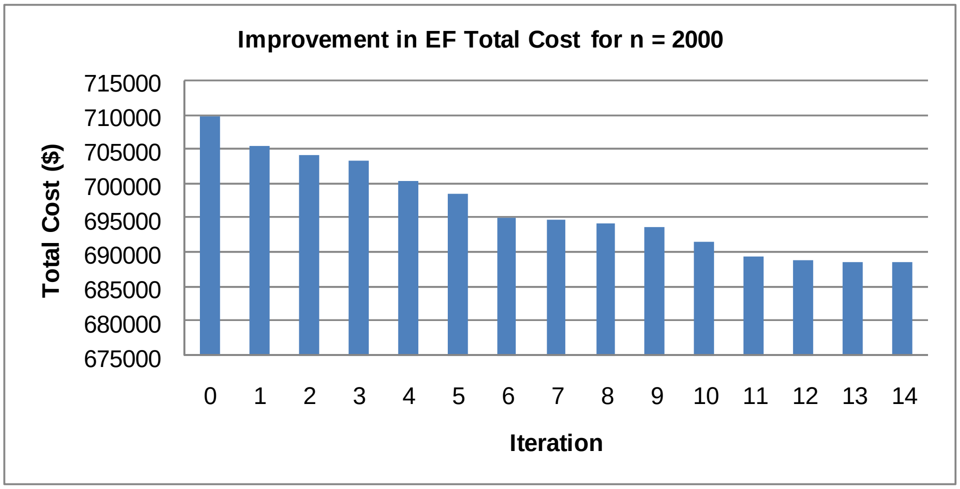


Table 5 - 22. Improvement in EF Total Cost for n = 2,000 pairings

Iteration	*Total Cost ($)*	*Improvement (%)*	*Calculation Time (sec)*	*Memory Usage (MB)*
0	709902	NA	NA	NA
1	705684.3	0.59	63715	83.0
2	704386	0.78	101421	76.1
3	703510.5	0.90	72754	52.5
4	700452.6	1.33	99301	65.9
5	698536.5	1.60	130417	72.9
6	695009.7	2.10	75379	43.0
7	694921.6	2.11	107807	50.0
8	694343.8	2.19	42961	61.4
9	693670.4	2.29	49557	69.9
10	691778.2	2.55	55344	61.0
11	689400	2.89	42536	62.0
12	688856.2	2.96	73966	48.8
13	688621.3	3.00	43646	55.0
14	688621.3	3.00	62691	83.9

Figure 5 - 22. Improvement in EF Total Cost for n = 2,000



Both AC and EF problem results had similar improvements of about three percent over the original total cost. Applying the heuristic to these full network problems did not provide as much cost reduction as solving the separate ACN, ACS, EFN, and EFS partitions. The total network data runs are inefficient because the heuristic currently cannot search across a large network and compare generated pairings before solving the restricted problems in CPLEX. A possible improvement to the heuristic could be implementing a dynamic search function that would further filter generated pairings based on location before solving the optimization problems in CPLEX.

5.8 Comparison of All Results

To summarize all of the findings from this study, Table 5-23 lists the results for all 14 problems solved. By examining partitioned problem results by crew type, it is evident that the largest improvements were seen for the EFN and EFS problems using a population size of 20,000 pairings. The partitioned EF problems also had better final solutions than the ACN and ACS for the cases with population size limits of 2,000 and 4,000 pairings. The original, partitioned EF crew schedules may have been the least optimal, and thus, had larger potential for improvements. The partitioned EF problems were also smaller than the partitioned AC problems, which may have facilitated better improvements.

When comparing the partitioned problems by region, the northern region results were better when the population size was 2,000. However, when the population size was increased to 4,000 and 20,000, the southern region results had larger improvements. The northern regions improvements may have been better for the smaller population size limit because of the larger number of duties for these problems. With more duties, the heuristic was able to find more "good" pairing combinations. However, the overall results appear to indicate that the southern region generally had larger improvements.

Table 5 - 23. Complete Table of Results

Problem	***Number of Duties***	***Population Size, n (pairings)***	***Improvement (%)***	***Calculation Time (hours)***	***Memory Usage (MB)***
ACN	1178	2000	4.5%	9.7	61.3
ACN	1178	4000	7.4%	29.3	148.6
ACN	1178	20000	15.8%	318.6	341.5
EFN	1044	2000	12.0%	14.8	30.2
EFN	1044	4000	21.0%	42.6	67.7
EFN	1044	20000	25.7%	330.9	210.1
ACS	1067	2000	3.0%	18.8	79.8
ACS	1067	4000	8.2%	34.2	151.6
ACS	1067	20000	18.0%	62.7	306.5
EFS	1049	2000	10.2%	11.2	33.2
EFS	1049	4000	18.3%	21.4	54.7
EFS	1049	20000	27.9%	86.3	516.4
AC	2245	2000	2.9%	333.6	906.9
EF	2093	2000	3.0%	266.3	885.4

As the population size increased, the calculation time and memory usage also increased. However, a direct relationship between the calculation time and memory usage was not observed. While generally problems that required longer calculation time also required more intensive memory usage, these variables were not directly related. The memory usage was found to be related to the number of duties, the population size, and the "fitness" of the random pairings generated by the heuristic.

The results from solving the full networks confirm that as the number of duties increases, the heuristic becomes less effective at improving the crew schedules. The larger AC problem had the smallest improvement, while the nearly as large EF problem had the second smaller improvement in total cost. These problems also required much longer calculation times, on the order of days, than the smaller problems with the same population size limit. As expected, the resulting memory usage for such large problems was far more intensive than the smaller, partitioned problems. Increasing the population size for these problems was ultimately limited by the computer memory constraints.

By partitioning the large problem by crew type and by region, the heuristic was successful in producing improved crew schedules within an acceptable amount of time. Examining the results indicated that the heuristic successfully replaced less efficient pairings with more efficient pairings.

Table 5-24 shows examples of inefficiencies in the current Amtrak schedules. The two cases shown in Table 5-24 are extracted from the original EFN schedule and are examples of poorly scheduled pairings. The first pairing only has two duties, which is equivalent to 6.67 hours of on-board, train work. This pairing also has only one, three hour away period. Despite working for so few hours in the weekly schedule, the crew member is guaranteed payment for 40 hours per week. As a result, the additional time must be paid for by Amtrak. Pairings with such few duties are a detriment to total cost because of the salary guarantee that Amtrak provides. An examination of the original schedules found many pairings with few duties. In the improved crew schedules, many of these small pairings can be combined with other pairings to create more efficient work schedules.

Table 5 - 24. Examples of Inefficient Pairings

Pairing 1 (from EFN schedule)

Duty	*Day*	*From*	*To*	*Departure Time*	*Arrival Time*	*Train Hours*
1	4	BOS	NHV	15.20.00	17.45.00	3.25
2	4	NHV	BOS	21.08.00	23.50.00	3.42
					TOTAL	6.67

Pairing 2 (from EFN schedule)

Duty	*Day*	*From*	*To*	*Departure Time*	*Arrival Time*	*Train Hours*
1	7	BOS	NHV	15.00.00	17.39.00	3.48
2	7	NHV	BOS	21.50.00	00.13.00	3.10
3	3	BOS	NYP	12.15.00	15.45.00	4.33
4	3	NYP	BOS	19.00.00	22.35.00	4.45
5	4	BOS	NYP	12.15.00	15.45.00	4.33
6	4	NYP	BOS	19.00.00	22.35.00	4.45
7	5	BOS	NYP	12.15.00	15.45.00	4.33
8	5	NYP	BOS	19.00.00	22.35.00	4.45
9	6	BOS	NHV	13.45.00	16.08.00	3.22
10	6	NHV	BOS	18.35.00	21.05.00	3.20
					TOTAL	39.34

The second pairing in Table 5-24 has a large amount of duties, essentially representing the opposite case from the first pairing. In this example, the total on-board, train hours is 39.34 hours, which is already almost equivalent to the regular 40 hours of work per week. When all of the paid away time and meal reimbursement is added, this crew member is being paid for over 40 hours per week, with additional hours being paid at the one-and-one-half overtime rate. Pairings such as this example with many duties are also a detriment to total cost because overtime costs. Ideally, most pairings would have the same amount of duties, and all of the pairings together would satisfy the work requirements for every Amtrak train in the network.

Chapter 6: Conclusions

6.1 Summary of Results

This thesis accomplished the goal of applying a genetic algorithm-based column generation heuristic to the passenger rail crew schedule problem for Amtrak's Northeast Corridor. As a part of the thesis, the schedule for trainperson and engineer (T&E) crew members in the Northeast Corridor was improved by applying the heuristic to a set partition problem (SPP) integer program formulation of the CSP. Specifically, the schedule pertains to conductors, assistant conductors, engineers, and firemen based in one of 16 total crew bases and serving a total of 32 release stations in the Northeast Corridor. Due to the complex work policies, union rules, and payment regulations, the crew scheduling problem (CSP) for the Northeast Corridor is highly constrained. While the constraints reduce the total number of feasible pairings, modeling the constraints becomes a difficult task. This thesis accomplished the goal of using the proposed heuristic to solve this large, real-world problem within a reasonable amount of computation time and computer memory resource.

The results of this thesis indicate that there are potential cost-savings in the Amtrak Northeast Corridor T&E crew schedule. While some crew pairings were not included in the analysis due to the data reduction steps in the process, the results show that total cost of crew payment can be reduced.

In order to confirm that the proposed heuristic is successful at improving the crew schedule, small problems based on the real-world data were developed. In addition to solving the small problems using the heuristic, the problems were formulated as full SPP integer programs and solved to optimality. The results from the comparison of the optimal and heuristic results showed that the heuristic can produce optimal or near-optimal solutions. For the larger problems, the heuristic did not find optimal solutions, although the heuristic solution was within 0.5 percent of the optimal solution. Additionally, the heuristic produced solutions in significantly less computation time and with less computer memory usage. As a result, the heuristic achieved the goal of improving the crew schedules within reasonable time and computer memory resources.

The proposed heuristic in this thesis was applied to the two separate types of crew for the two portions of the Northeast Corridor network (northern and southern region, as previously defined) as well as to the entire network. For the segmented problems (EFN, ACN, EFS, and ACS problems) with approximately 1000 duties each, the heuristic was applied using population size limits of 2,000, 4,000, and 20,000 pairings. The results indicated that the larger population size of 4,000 produced schedules with larger improvements, but at the cost of computation time and computer memory. The results with the larger population size of 20,000 confirmed that a lower cost schedule could be obtained, although computation time drastically increased. Finally, the heuristic was also applied to the full Northeast

Corridor for each set of crew types (EF and AC problems), which included approximately 2,000 duties each. Due to the large problem size and resulting calculation time, these problems were solved with a population size limit of 2,000.

When the EFN, ACN, EFS, and ACS segmented problems were solved with a limit of 2,000 pairings, the improvements in total cost ranged from three percent to 12 percent. Computation time ranged from nine to 19 hours, which is reasonable given the large size of the problem. Memory usage was also within reasonable limits, ranging from 30 to 80 megabytes. When the limit on pairings was doubled to 4,000 pairings, the improvements in total cost nearly doubled as well. The percentage improvements ranged from seven to 21 percent. However, computation time also nearly doubled to 21 to 43 hours. Even larger improvements were seen when the original limit on the number of pairings was increased tenfold to 20,000 pairings. The improvements were then ranging from 16 to 28 percent. The computation time also increased to 63 to 86 hours, or 2.6 to 13.6 days. These computation times may be acceptable in practice for long-range scheduling in the real world. However, for short-term planning, these computation times may be long. These computation times are also excessive for tactical planning.

The improvements to the crew schedule in this work were for weekly schedules. As a result, the overall annual savings was found to be significant. The results show crew scheduling optimization is a viable option for reducing operating costs for Amtrak. Especially in the Northeast Corridor, where ridership has increased in 2008 due to rising fuel costs and greater dependence on public transportation, Amtrak must find ways to reduce operating costs. The work in this thesis shows that a significant amount of crew salary cost is directed to crew payment for overtime and salary guarantees. While perfectly optimized crew schedules are not be possible due to unforeseen changes in schedules and other factors, overall costs for crew payment can be reduced to provide annual savings.

6.2 Summary of Contributions

The work in this thesis contributed the following to the area of rail crew scheduling. First, the work included the development and application of a new genetic algorithm-based column generation heuristic to a real world passenger rail crew scheduling problem. The research in passenger rail crew scheduling, especially within North America, is sparse. As previously discussed, the existing literature on passenger rail crew scheduling has primarily been for applications in Europe and Asia, where there are stronger passenger rail markets. In particular, there is limited research on crew scheduling improvement and optimization for Amtrak.

Second, many other rail crew scheduling applications did not include the complex payment regulations that were a part of the problem in this work. Few other CSPs have the case in which there is overtime pay, guaranteed salary, away pay, lodging, and meals. The constraints specific to the Amtrak problem are not applicable in many other situations. However, the improvements seen from the proposed solution method in this thesis show that genetic algorithm-based column generation heuristics have potential for solving large, real world CSPs.

Finally, the results of this thesis indicate that Amtrak can reduce operation costs by improving crew schedules. While there may be other union regulations that may restrict some possible pairings, further research is merited based on the results of this work.

6.3 Implications of Findings

The overall findings of this study were that improvements can be made to Amtrak's Northeast Corridor crew schedules. Many implementation details would need to be considered when creating and implementing new crew schedules.

First, some work regulations were not considered in the constraints of this thesis. Amtrak may allow more senior crew members to select certain schedules and duties as a way of rewarding loyal employees. Additionally, seniority may be a factor when assigning extra board duties.

The crew schedule improvements in the thesis were most applicable to long-term scheduling since the calculation times for the best solutions were on the order of days. However, shorter-term scheduling may be necessary to accommodate crew members taking leave, holidays, and other irregular absences. Scheduling would also need to take into account altered train schedules for special events or due to weather and other emergencies. These considerations lead to the area of robust crew scheduling.

Second, the comfort of crew was not considered in this work. As a result, efficient pairings that were selected for the improved crew schedules may be pairings that are excessively long in actual practice. A pairing might be good in terms of Amtrak's operating costs, but may prove to be demanding for crew members. While union regulations restrict the most undesirable work schedules, demanding schedules may still be possible. Especially in the Northeast Corridor, where many trips are relatively short in length, pairings may require a crew to travel on several short train trips in a single shift. The constant boarding, on-board work, and disembarking may be too demanding for actual schedules. Demanding schedules are not only harmful to crew health and morale, but they can also affect train safety. Overly tired crew members may not be fully capable of executing the work responsibilities. Therefore, the improved crew schedules found in this work would still need to be examined by management to determine if they are reasonable.

Third, prior to implementing new crew schedules, crew experience and skills must also be considered. Certain train duties and yard work may require specific experience or skills. Again, crew experience was not taken into consideration in the improved schedules for this thesis.

Lastly, implementation of a new crew schedule with fewer required crew members may not be possible. Significantly downsizing staff is unlikely a viable option despite savings in cost. Instead, crew scheduling may be done for future expected work requirements in which additional crew members are needed. Another option would be to incorporate an additional constraint of using all available crew members in any new crew schedules.

6.4 Future Work

The results of this research indicated potential for using a genetic algorithm-based column generation technique for crew scheduling. This study used a simple genetic algorithm with a single, randomly-selected crossover point and mutation of a single duty. Further studies may examine more complex genetic algorithms, which may result in a more efficient heuristic.

Based on the results of this research, further work may be considered for improving or optimizing Amtrak crew schedules. The work in thesis simplified the CSP by examining specifically the Northeast Corridor and by segmenting this corridor into two sections, a northern and southern section. For this thesis, the Northeast Corridor was selected because it is the most heavily used corridor in the Amtrak network. Future work may consider improving crew schedules for other corridors in the network or the entire Amtrak network as a whole. Because the trains in the Northeast Corridor travel relatively short distances, pairings for these crew schedules require more duties and possibly more time away. Amtrak trains in the Midwest and Western United States typically travel longer distances, so the crew schedule improvements and potential cost-savings may differ for these corridors.

This thesis also studied only the T&E crew members since this set of crew comprises a large portion of Amtrak employees and because the union and payment regulations for this crew set are the same. Future research in Amtrak crew scheduling may examine the on-board services crew members, which are the crewpersons that are responsible for food, cleaning, and other services on trains. The union and payment regulations are more lax for this crew set. For example, these crew members may be away from their crew base for a longer period of time and typically work on a single train rather than switching trains throughout a crew pairing.

In addition, the results from this thesis show that the proposed heuristic produces improvements within time and memory constraints for longer-term scheduling and planning. Since some of the problems required several days of running the heuristic, the solution method would not be applicable for tactical and shorter-term crew scheduling. Future work may address the tactical CSP and real-time crew dispatching. These problems differ in that crew and train information is gathered in real-time, and the new crew schedule results are needed for immediate use. This requires faster, more powerful solution methods that can produce improvements in a very short time period. The tactical CSP is an important problem in rail and transit crew scheduling due to unforeseen problems such as train delays, mechanical train problems, crew absence, among many other problems. Each of these problems requires a quick solution that minimizes the impact on and cost of the rail system. In addition, another area of research for crew scheduling is that of designing robust schedules. These CSPs aim to find crew schedules that are more flexible to emergencies and other changes in work requirements and crew availability.

Further research in passenger rail crew scheduling is important as rail becomes a more attractive option for traveling in the United States. With increasing fuel costs and greater environmental responsibility, Amtrak ridership may increase in the future. At the same time, Amtrak continues to face criticism for low revenues and high operating and maintenance costs of the system. Crew scheduling improvements offer an area for reducing costs since a large portion of the operating costs is for paying Amtrak employee salaries. Additionally, as the demand for Amtrak trains increases or decreases in different corridors, Amtrak crew requirements for trains will require adjustments. As a result, future work may also examine the effects of marginal changes in the schedule on costs and how these schedules can be improved.

Appendix

Sample of Selected Existing AC Crew Schedule Data

KEY	HOME	SCHD_EFF	DTL_RPT_LOC	DTL_RLSE_LOC	DEPART_TIME	ARRV_TIME
AXAN722	ALB	7-Apr-08	NYP	ALB	00.30.00	02.45.00
CXAN722	ALB	7-Apr-08	NYP	ALB	00.30.00	02.45.00
ANH710	NHV	12-May-08	NYP	NHV	03.15.00	05.00.00
CNH710	NHV	12-May-08	NYP	NHV	03.15.00	05.00.00
ABN713	BOS	12-May-08	NHV	BOS	05.05.00	07.52.00
CBN713	BOS	12-May-08	NHV	BOS	05.05.00	07.52.00
APB706	POR	12-May-08	POR	BON	06.00.00	08.25.00
CPB706	POR	12-May-08	POR	BON	06.00.00	08.25.00
AALB716	ALB	7-Apr-08	ALB	NYP	06.05.00	08.35.00
CALB716	ALB	7-Apr-08	ALB	NYP	06.05.00	08.35.00
ASN706	SPG	12-May-08	SPG	NHV	06.15.00	08.00.00
CSN706	SPG	12-May-08	SPG	NHV	06.15.00	08.00.00
ABN756	BOS	28-May-08	BOS	NHV	06.40.00	09.08.00
ABN708	BOS	12-May-08	BOS	NHV	06.40.00	09.08.00
CBN708	BOS	12-May-08	BOS	NHV	06.40.00	09.08.00
ANH706	NHV	12-May-08	NYP	NHV	06.55.00	08.29.00
CNH706	NHV	12-May-08	NYP	NHV	06.55.00	08.29.00
AALB713	ALB	7-Apr-08	NYP	ALB	07.15.00	09.50.00
CALB713	ALB	7-Apr-08	NYP	ALB	07.15.00	09.50.00
ASN702	SPG	12-May-08	SPG	NHV	07.20.00	08.55.00
CSN702	SPG	12-May-08	SPG	NHV	07.20.00	08.55.00
XANH706	NHV	20-Jun-07	NHV	NYP	08.11.00	09.55.00
ANH711	NHV	12-May-08	NHV	NYP	08.11.00	09.55.00
CNH711	NHV	12-May-08	NHV	NYP	08.11.00	09.55.00
APB703	POR	12-May-08	POR	BON	08.15.00	10.40.00
CPB703	POR	12-May-08	POR	BON	08.15.00	10.40.00
AALB719	ALB	7-Apr-08	ALB	NYP	08.05.00	10.35.00
CALB719	ALB	7-Apr-08	ALB	NYP	08.05.00	10.35.00
AAM703	ALB	7-Apr-08	MTR	ALB	09.30.00	17.40.00
CAM703	ALB	7-Apr-08	MTR	ALB	09.30.00	17.40.00
AALB714	ALB	7-Apr-08	NYP	ALB	08.15.00	10.40.00
CALB714	ALB	7-Apr-08	NYP	ALB	08.15.00	10.40.00
ANH712	NHV	12-May-08	NHV	BOS	08.33.00	11.10.00
ANH755	NHV	12-May-08	NHV	BOS	08.33.00	11.10.00
CNH712	NHV	12-May-08	NHV	BOS	08.33.00	11.10.00
ANFL701	ALB	7-Apr-08	NFL	ALB	08.35.00	14.50.00
CNFL701	ALB	7-Apr-08	NFL	ALB	08.35.00	14.50.00
ANFL702	ALB	7-Apr-08	NFL	ALB	08.35.00	14.50.00
CNFL702	ALB	7-Apr-08	NFL	ALB	08.35.00	14.50.00
ASN708	SPG	5-Jun-08	SPG	NHV	08.40.00	10.26.00
CSN708	SPG	5-Jun-08	SPG	NHV	08.40.00	10.26.00
ABN719	BOS	12-May-08	BOS	NHV	08.40.00	11.08.00
XABN703	BOS	12-May-08	BOS	NHV	08.40.00	11.08.00

CBN719	BOS	12-May-08	BOS	NHV	08.40.00	11.08.00
CALB718	ALB	7-Apr-08	ALB	ALB	00.00.01	00.00.01
ASN706	SPG	12-May-08	NHV	SPG	09.00.00	10.40.00
CSN706	SPG	12-May-08	NHV	SPG	09.00.00	10.40.00
AZN706	NYZ	12-May-08	NYP	NHV	09.00.00	10.44.00
AZN753	NYZ	12-May-08	NYP	NHV	09.00.00	10.44.00
CZN706	NYZ	12-May-08	NYP	NHV	09.00.00	10.44.00
ANH757	NHV	12-May-08	NHV	NYP	09.11.00	10.55.00
XANH701	NHV	7-Apr-08	NHV	NYP	09.11.00	10.55.00
XCNH701	NHV	7-Apr-08	NHV	NYP	09.11.00	10.55.00
APB706	POR	12-May-08	BON	POR	08.50.00	11.15.00
CPB706	POR	12-May-08	BON	POR	08.50.00	11.15.00
AABF703	ALB	19-May-08	BUF	ALB	09.35.00	15.40.00
CABF703	ALB	19-May-08	BUF	ALB	09.35.00	15.40.00
ABN716	BOS	12-May-08	BOS	NHV	09.40.00	12.08.00
ABN752	BOS	12-May-08	BOS	NHV	09.40.00	12.08.00
XABN701	BOS	12-May-08	BOS	NHV	09.40.00	12.08.00
CBN716	BOS	12-May-08	BOS	NHV	09.40.00	12.08.00
ANFL701	ALB	7-Apr-08	ALB	NFS	10.05.00	16.15.00
CNFL701	ALB	7-Apr-08	ALB	NFS	10.05.00	16.15.00
ANFL702	ALB	7-Apr-08	ALB	NFS	10.00.00	16.25.00
CNFL702	ALB	7-Apr-08	ALB	NFS	10.00.00	16.25.00
AZN754	NYZ	12-May-08	NYP	NHV	10.00.00	11.42.00
XAZN701	NYZ	12-May-08	NYP	NHV	10.00.00	11.42.00
XCZN701	NYZ	12-May-08	NYP	NHV	10.00.00	11.42.00
AALB712	ALB	7-Apr-08	ALB	NYP	10.05.00	12.35.00
CALB712	ALB	7-Apr-08	ALB	NYP	10.05.00	12.35.00
ANH756	NHV	12-May-08	NHV	NYP	10.41.00	12.25.00
ANH709	NHV	12-May-08	NHV	NYP	10.41.00	12.25.00
CNH709	NHV	12-May-08	NHV	NYP	10.41.00	12.25.00
ASN703	SPG	12-May-08	SPG	NHV	10.20.00	12.00.00
CSN703	SPG	12-May-08	SPG	NHV	10.20.00	12.00.00
AAM702	ALB	7-Apr-08	NYP	ALB	10.20.00	12.48.00
CAM702	ALB	7-Apr-08	NYP	ALB	10.20.00	12.48.00
AAM704	ALB	7-Apr-08	ALB	MTR	11.05.00	19.10.00
CAM704	ALB	7-Apr-08	ALB	MTR	11.05.00	19.10.00
ANH708	NHV	12-May-08	NHV	BOS	10.46.00	13.07.00
ANH754	NHV	12-May-08	NHV	BOS	10.46.00	13.07.00
CNH708	NHV	12-May-08	NHV	BOS	10.46.00	13.07.00
AANF705	ALB	7-Apr-08	NFS	ALB	10.30.00	19.00.00
CANF705	ALB	7-Apr-08	NFS	ALB	10.30.00	19.00.00
ASN702	SPG	12-May-08	NHV	SPG	11.05.00	12.45.00
CSN702	SPG	12-May-08	NHV	SPG	11.05.00	12.45.00

References

Abbink, E. and M. Fischetti, L. Kroon, G. Timmer, M. Vromans (2005). “Reinventing Crew Scheduling at Netherlands Railways”. *Interfaces* 35 (5), 393-401.

Arabeyre, J., J. Fearnley, F. Steiger, and W. Teather. (1969). “The Airline Crew Scheduling Problem: A Survey.” *Transportation Science* 3, 140–163.

Barnhart, C., E. Johnson, G. Nemhauser, M. Savelsbergh, and P. Vance. (1998). “Branch-and-Price:Column Generation for Solving Huge Integer Programs.” *Operations Research* 46, 316–329.

Bennett, B. and R. Potts. (1968). “Rotating Rosters for a Transit System.” *Transportation Science* 2, 14–33.

Bengtsson, L., R. Galia, T. Gustafsson, C. Hjorring, and N. Kohl. (2007). “Railway Crew Pairing Optimization.” *Algorithmic Methods for Railway Optimization, Lecture Notes in Computer Science* 4359, 126-144.

Bodin, L., B. Golden, A. Assad, and M. Ball. (1983). “Routing and Scheduling of Vehicles and Crews –The State of the Art.” *Computers and Operations Research* 10(2), 63–211.

Cappanera, P. and G. Gallo (2004). “A Multicommodity Flow Approach to the Crew Rostering Problem”. *Operations Research* 52(4), 583-596.

Caprara, A., M. Fischetti, P. Guida, P. Toth, and D. Vigo. (1999). “Solution of Large-Scale Railway Crew Planning Problems: The Italian Experience.” *Computer-Aided Transit Scheduling, Lecture Notes in Economics and Mathematical Systems* 430, 1–18.

Caprara, A., M. Monaci, and P. Toth. (2001). “A Global Method for Crew Planning in Railway Applications.” *Computer-Aided Scheduling of Public Transport, Lecture Notes in Economics and Mathematical Systems* 505,17–36.

Cavique, L., C. Rego, and I. Themido. (1999). “Subgraph Ejection Chains and Tabu Search for the Crew Scheduling Problem.” *Journal of the Operational Research Society* 50(6), 608–616.

Clement, R. and A. Wren. (1995). “Greedy Genetic Algorithms, Optimizing Mutations and Bus Driver Scheduling.” *Computer-Aided Transit Scheduling, Lecture Notes in Economics and Mathematical Systems*, 430, 213–235.

Desrochers, M. and F. Soumis. (1989). “A Column Generation Approach to the

Urban Transit Crew Scheduling Problem." *Transportation Science* 23(1), 1–13.

Emden-Weinert, T. and M. Proksch. (1999). "Best Practice Simulated Annealing for the Airline Crew Scheduling Problem." *Journal of Heuristics* 5(4), 419–436.

Ernst, A., H. Jiang, M. Krishnamoorthy, H. Nott, and D. Sier. (2001). "An Integrated Optimization Model for Train Crew Management." *Annals of Operations Research* 108(1/4), 211–224.

Falkner, J. and D. Ryan. (1992). "Set Partitioning for Bus Crew Scheduling in Christchurch." *Computer-Aided Transit Scheduling*: *Proceedings of the Fifth International Workshop*, 359-378.

Fahle T., U. Junker , S. Karisch, N. Kohl, M. Sellmann , B. Vaaben (2002). "Constraint programming based column generation for crew assignment". *Journal of Heuristics* 8, 59–81.

Fores, S., L. Proll, and A. Wren. (1998). "A Column Generation Approach to Bus Driver Scheduling". *Transportation Networks: Recent Methodological Advances*, 195–208.

Freling, R., and R. Lentink, A..Wagelmans. (2004). "A Decision Support System for Crew Planning in Passenger Transportation Using a Flexible Branch-and-Price Algorithm." *Annals of Operations Research* 127, Special Issue on Staff Scheduling and Rostering, 203–222.

Garfinkel R. and G. Nemhauser. (1970). "Optimal political districting by implicit enumeration techniques". *Management Science* 16(8), 495–508.

Graves, G., R. McBride, I. Gershkoff, D. Anderson, and D. Mahidhara. (1993). "Flight Crew Scheduling." *Management Science* 39(6), 736–745.

Haase, K. and C. Friberg. (1999). "An Exact Algorithm for the Vehicle and Crew Scheduling Problem." *Lecture Notes in Economics and Mathematical Systems*.

Haase, K.,G. Desaulniers, and J. Desrosiers. (2001). "Simultaneous Vehicle and Crew Scheduling in Urban Mass Transit Systems." *Transportation Science* 35(3), 286-303.

Hoffman, K. and M. Padberg. (1993). "Solving Airline Crew Scheduling Problems by Branch-and-Cut." *Management Science* 39(6), 657–682.

Jesperson-Groth, J., D. Potthoff, D. Huisman, L. Kroon, G. Maroti, and M. Nielsen.

(2007). “Disruption Management in Passenger Railway Transportation.” IMM Technical Report, Informatics and Mathematical Modelling, Technical University of Denmark.

Klabjan D, E. Johnson E, and G. Nemhauser. (2001). “Solving large airline crew scheduling problems: random pairing generation and strong branching”. *Computational Optimization and Applications* 20, 73–91.

Kohl, N., A. Larsen, J. Larsen, A. Ross, and S. Tiourine. (2007). “Airline disruption management—Perspectives, experiences and outlook.” *Journal of Air Transport Management* 13(3): 149-162.

Kroon, L. and M. Fischetti. (2001). “Crew Scheduling for Netherlands Railways Destination.” *Computer-Aided Scheduling of Public Transport, Lecture Notes in Economics and Mathematical Systems*, 505: 81–202.

Kwan, A., R. Kwan, and A. Wren. (1999). “Driver Scheduling Using Genetic Algorithms with Embedded Combinatorial Traits.” *Computer-Aided Scheduling of Public Transport, Lecture Notes in Economics and Mathematical Systems*, 421: 81-102.

Kwan, R. and A. Wren. (1996). “Hybrid Genetic Algorithms for Bus Driver Scheduling.” *Advanced Methods in Transportation Analysis*, 609-619.

Lavoie, S., M. Minoux, and E. Odier. (1988). “A New Approach for Crew Pairing Problems by Column Generation with an Application to Air Transportation.” *European Journal of Operational Research* 35(1), 45–58.

Levine, D. (1996). “Application of a Hybrid Genetic Algorithm to Airline Crew Scheduling.” Computers and Operations Research 23(6), 547–558.

Lucic, P. and D. Teodorovic. (1999). “Simulated Annealing for the Multi-Objective Aircrew Rostering Problem.” *Transportation Research, Part A: Policy and Practice* 33(1), 19–45.

Makri A. and D. Klabjan (2004). “A new pricing scheme for airline crew scheduling” *INFORMS Journal on Computing* 16(1), 56–67.

Marsten R. (1974). “An algorithm for large set partitioning problems”. *Management Science* 20(5), 774–87.

Marsten, R. and F. Shepardson. (1981). “Exact Solution of Crew Scheduling Problems Using the Set Partitioning Model: Recent Successful Applications.” *Networks* 11, 165–177.

Mellouli, T. (2001). “A Network Flow Approach to Crew Scheduling Based on an

Analogy to a Vehicle Maintenance Routing Problem." *Computer-Aided Scheduling of Public Transport, Lecture Notes in Economics and Mathematical Systems*, 505, 91–120.

Nissen, R. and K. Haase. (2006). "Duty-period-based network model for crew rescheduling in European airlines." *Journal of Scheduling* 9(3), 255-278.

Rubin, J. (1973). "A Technique for the Solution of Massive Set Covering Problem, with Application to Airline Crew Scheduling." *Transportation Science* 7(1), 34–48.

Ryan, D. and B. Foster. (1981). "An Integer Programming Approach to Scheduling." *Computer Scheduling of Public Transport, Urban Passenger Vehicle and Crew Scheduling*, 269–280.

Sellman, M., K. Zervoudakis, P. Stamaopoulos, T. Fahle (2004). "Crew Assignment via Constraint Programming: Integrating Column Generation and Heuristic Tree Search". *Annals of Operations Research* 115(1-4), 207-225.

Shen, Y. and R. Kwan. (2001). "Tabu Search for Driver Scheduling". *Computer-Aided Scheduling of Public Transport, Lecture Notes in Economics and Mathematical Systems*, 505, 121–136.

Smith, B. and A. Wren. (1988). "A Bus Crew Scheduling System Using a Set Covering Formulation." *Transportation Research, Part A: General* 22A(2), 97–108.

Vaidyanathan, B. and K. Jha, R. Ahuja (2007). "Multicommodity network flow approach to the railroad crew-scheduling problem". *IBM Journal of Research and Development* 51(3/4), 325-344.

Willers, W. (1995). "Improved Algorithms for Bus Crew Scheduling." Doctoral dissertation, University of Leeds.

Wren, A. (1981). *Computer Scheduling of Public Transport Urban Passenger Vehicle and Crew Scheduling*. North-Holland, Amsterdam.

Wren, A. and D. Wren. (1995). "A Genetic Algorithm for Public Transport Driver Scheduling." *Computers and Operations Research* 22(1), 101–110.

Yan, S. and Y. Tu. (2002). "A Network Model for Airline Cabin Crew Scheduling." *European Journal of Operational Research* 140(3), 531–540.