Technical Report Documentation Page

1. Report No.	2. Government Accession No.	3. Recipient's Catalog No.
SWUTC/09/476660-00063-1		

4. Title and Subtitle	5. Report Date
Vehicle and Driver Scheduling for Public Transit	August 2009
	6. Performing Organization Code

7. Author(s)	8. Performing Organization Report No.
Kristen Torrance, Ashley R. Haire and Randy B. Machemehl	476660-00063-1

9. Performing Organization Name and Address	10. Work Unit No. (TRAIS)
Center for Transportation Research University of Texas at Austin 3208 Red River, Suite 200 Austin, Texas 78705-2650	11. Contract or Grant No. DTRT07-G-0006

12. Sponsoring Agency Name and Address	13. Type of Report and Period Covered
Southwest Region University Transportation Center Texas Transportation Institute Texas A&M University System College Station, Texas 77843-3135	14. Sponsoring Agency Code

15. Supplementary Notes

Supported by a grant from the U.S. Department of Transportation, University Transportation Centers Program

16. Abstract The problem of driver scheduling involves the construction of a legal set of shifts, including allowance of overtime, which cover the blocks in a particular vehicle schedule. A shift is the work scheduled to be performed by a driver in one day, while a block is a sequence of trips made by a single bus. Blocks can be divided between different drivers if they begin or end at relief points, providing the opportunity for change. The goal is to make the schedule as efficient as possible, therefore minimizing the amount of changes that need to be made.

In determining the ideal schedule, local and national labor rules must be considered. These involve restrictions specified by the user, including, but not limited to, total time worked per day; total time worked per week; the length of time that may be worked without a meal break; the total spreadover, which is the duration between beginning and ending a shift; and/or the number of days off per week (Wren and Rousseau, 1995).

The problem of vehicle scheduling involves determining the optimal route structure, span of service, and service frequencies for the transit agency and assigning vehicles to the routes. This involves considering cycle times, number of vehicles, timed transfers, layover time and locations, recovery time, and any difference in weekday and weekend services (Wren and Rousseau, 1995). Knowing the blocks of work that must be satisfied, vehicles must be assigned to each of the blocks and given departure and arrival times with a goal of optimizing the number of hours the vehicle is transporting passengers, known as platform hours.

The research contained herein describes work undertaken by the research team, including literature review and a survey of existing methodologies. Preliminary programming code was created to act as a benchmark for future endeavors in this area and elaboration of the complicating factors led the team to conclude that in order to fully complete a comprehensive working scheduling technique, substantially greater resources would be needed.

17. Key Words	18. Distribution Statement
Driver Scheduling, Vehicle Scheduling, Optimal Route Structure, Transit Platform Hours	No restrictions. This document is available to the public through NTIS: National Technical Information Service 5285 Port Royal Road Springfield, Virginia 22161

19. Security Classif.(of this report)	20. Security Classif.(of this page)	21. No. of Pages	22. Price
Unclassified	Unclassified	44	

Form DOT F 1700.7 (8-72) **Reproduction of completed page authorized**

Vehicle and Driver Scheduling for Public Transit

by

Kristen Torrance
Ashley R. Haire
Randy B. Machemehl

Research Report SWUTC/09/476660-00063-1

Southwest Region University Transportation Center
Center for Transportation Research
University of Texas at Austin
Austin, Texas

August 2009

ACKNOWLEDGEMENTS

The authors recognize that support for this research was provided by a grant from the U.S. Department of Transportation, University Transportation Centers Program to the Southwest Region University Transportation Center.

DISCLAIMER

The contents of this report reflect the views of the authors, who are responsible for the facts and the accuracy of the information presented herein. This document is disseminated under the sponsorship of the Department of Transportation, University Transportation Centers Program, in the interest of information exchange. The U.S. Government assumes no liability for the contents or use thereof.

ABSTRACT

The problem of driver scheduling involves the construction of a legal set of shifts, including allowance of overtime, which cover the blocks in a particular vehicle schedule. A shift is the work scheduled to be performed by a driver in one day, while a block is a sequence of trips made by a single bus. Blocks can be divided between different drivers if they begin or end at relief points, providing the opportunity for change. The goal is to make the schedule as efficient as possible, therefore minimizing the amount of changes that need to be made.

In determining the ideal schedule, local and national labor rules must be considered. These involve restrictions specified by the user, including, but not limited to, total time worked per day; total time worked per week; the length of time that may be worked without a meal break; the total spreadover, which is the duration between beginning and ending a shift; and/or the number of days off per week (Wren and Rousseau, 1995).

The problem of vehicle scheduling involves determining the optimal route structure, span of service, and service frequencies for the transit agency and assigning vehicles to the routes. This involves considering cycle times, number of vehicles, timed transfers, layover time and locations, recovery time, and any difference in weekday and weekend services (Wren and Rousseau, 1995). Knowing the blocks of work that must be satisfied, vehicles must be assigned to each of the blocks and given departure and arrival times with a goal of optimizing the number of hours the vehicle is transporting passengers, known as platform hours.

The research contained herein describes work undertaken by the research team, including literature review and a survey of existing methodologies. Preliminary programming code was created to act as a benchmark for future endeavors in this area and elaboration of the complicating factors led the team to conclude that in order to fully complete a comprehensive working scheduling technique, substantially greater resources would be needed.

EXECUTIVE SUMMARY

Vehicle and driver scheduling in public transportation systems are complex, interconnected problems requiring extensive knowledge of transit operations and legal labor agreements. The problems should be solved iteratively or simultaneously, because finding independent solutions to each does not necessarily guarantee a cohesive solution for the system as a whole.

The primary software programs in use today, and their associated algorithms, have been developed in the private domain and are proprietary in nature. The research here sought to develop a public-domain algorithm to find solutions to the vehicle and driver scheduling problems.

In the course of this research, the team conducted a background review of relevant literature surrounding the public transportation scheduling problems, and surveyed the 20 largest transit agencies to determine the scheduling methodologies in use at each. The team then investigated the most widely-used scheduling software programs through additional literature reviews and personal interviews with algorithm and software developers. A benchmark code was developed for small systems that may be expanded upon through future research in this area.

The vehicle and driver scheduling problems are made more complex through many constraints placed on both the definition of the mathematical problem and issues arising from logistical and labor ramifications. The most pervasive of these constraints affecting the scheduling problems, particularly the driver scheduling problem, are generated from the contractual labor agreements between the unionized driver work force and the transit operator. While similarities exist in these agreements from one agency to the next, the details of the stipulations can vary substantially. Ideally, a newly developed scheduling algorithm would be capable of flexibly accommodating any and all of these constraints. However, conducting a comprehensive survey of public transportation agencies to ensure inclusion of all potential labor rules would require substantially greater resources than those to which the research team has access. Furthermore, it is not clear that a demand exists for a public-domain scheduling product.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1: INTRODUCTION

In designing an effective and efficient public transportation system, virtually every transit authority in the world must optimize their available resources such that costs are minimized and numerous other criteria are met. Primary tasks in planning for transit systems include route design, vehicle scheduling, and driver scheduling. Aside from minimization of costs, the criteria involved in conducting these tasks may include operational time, the number of vehicles and/or drivers required, and a variety of driver-satisfaction aspects, including union rules. As such, the tasks become multi-objective processes.

Substantial research, primarily in the operations research field, has been conducted in the past by researchers seeking to develop new methodologies for solving the driver scheduling problem. This research has typically focused on solving the route design problem firstly, followed by the vehicle and then driver scheduling problems. While this sequential series of problem solving produces effective results, it is believed by this research team that an iterative process may better suit the procedure in order to develop a more efficient solution. Additionally, many of the developed methods researched previously lack an ability to be practically applied in transit planning practice, due to simplifying assumptions and loosely-defined constraints that prove unrealistic for practitioners.

Significant research efforts have focused on solving the problems of transit scheduling, both for vehicles and for drivers. Smaller transit authorities may employ relatively simplistic methods, while larger agencies obviously require more complex tools.

One of the basic references for transit scheduling can be found in *TCRP (Transit Cooperative Research Program) Report 30*, entitled "Transit Scheduling: Basic and Advanced Manuals". This publication of the Transportation Research Board presents a manual for self-learning about transit scheduling, breaking the material into step-by-step procedures for scheduling transit services. Each chapter is preceded with study objectives and concluded with practice problems covering the key points in the chapter. The steps outlined in the chapters include (a) service policies and schedule development, (b) trip generation, (c) blocking, (d) runcutting, and (e) rostering. These five items are presented for the basic scheduling problem and for the advanced scheduling problem, which includes the dynamic process of responsive scheduling (TCRP Report 30). The benefits of having a dynamic, automated system are that such a system would provide more efficient schedules, can reduce the staff time for scheduling processes, can reduce costs of both vehicles and operators, and it provides more flexibility in the scheduling process.

Most research efforts in the field of transit scheduling optimization have occurred in the realm of operations research. Using various solution techniques, these efforts have sought to solve the transit scheduling problem as one of three problems: the vehicle scheduling problem, the driver scheduling problem, or the solving of both problems simultaneously.

A study by Banihashemi and Haghani solved an optimization model for large-scale bus transit scheduling, which they called the multiple-depot vehicle scheduling problem with route time constraints. While the procedure this team developed was applied to test problems and to the Baltimore, Maryland transit system, the researchers acknowledge that application of the technique to real-world large-scale systems requires reducing the size of the basic multi-depot scheduling problem to a manageable and solvable size. However, once this reduction was accomplished, the researchers found that their model improved upon the then-existing Baltimore

schedule in the number of vehicles, operational time, and total cost required (Banihashemi and Hagani, 2000).

In solving the multi-objective metaheuristic problem of bus driver scheduling, a research team based in Spain and Portugal proposed in 2001 that the tabu search and genetic algorithms provided good solutions while presenting aspects that specifically suited the structure of the driver scheduling problem. This research broke the transit planning process into four steps, (1) timetabling, (2) vehicle scheduling, (3) crew scheduling, and (4) duty rostering, and focused its attentions on the bus driver scheduling problem. The procedures used in this research were incorporated into the GIST Planning Transportation Systems, a tool in use by several transit agencies, which produces timetables, vehicle schedules, driver duties, and duty rosters for individual drivers (Lourenço et al., 2001).

Research in the Netherlands sought to reinvent the crew scheduling procedure for the Netherlands Railways in 2001, following strikes conducted by the drivers who were unsatisfied with their schedules. Application of a new solution method and model led to greater scheduling diversity, lessened personnel costs, and improved the railway's efficiency. They named this new method "Sharing-Sweet-and-Sour" because it sought to allocate the desirable and less desirable shifts in a more equitable manner (Abbink et al., 2005).

Many studies, as mentioned, seek to combine the vehicle and driver scheduling problems, or to treat them as similar problems. In a 2001 study in Italy, a multi-depot scheduling problem was proposed in which the constraints included the "spread time" (the time between the beginning and end of a driver's duty) and the "working time" (the total shift time of the driver). In this model, a linear programming formulation was used as were exact and heuristic solution procedures. A branch-and-cut algorithm was used with heuristic procedures to solve both the vehicle scheduling and driver scheduling problems individually, and then were evaluated based on both artificial and actual test problems (Fischetti et al., 2001)

Another 2001 study from Germany and Canada solved the vehicle and crew scheduling problems simultaneously. The research presents an exact approach; however, it is applicable only to a system with a single depot and a homogenous vehicle fleet. Set-partitioning was used, as were side constraints for the bus itineraries, which guarantee an optimal vehicle assignment. Column generation and a branch-and-bound scheme were used to develop a model that provided improved performance on artificial scenarios (Haase, 2001).

Based on American Public Transportation Agency (APTA) data, the 20 largest transit companies in the United States were identified based on 2004 data of annual passenger trips (APTA, 2003). Currently, among these transit agencies, only two programs, both which are maintained in the private sector, are used for transit scheduling. These programs, TRAPEZE developed by the Trapeze Group and HASTUS developed by Giro, Inc. are optimization software programs that attempt to minimize the cost associated with both the drivers and the vehicles required to operate a transit service. In evaluating the current state of the industry and the studies that have been conducted in this field, it is evident that the applicability of the research to actual transit planning practice is limited. This is due to the assumptions and constraints that are currently used to simplify the problem. Therefore, by evaluating different solution techniques, a more realistic and applicable solution can be developed.

CHAPTER 2: DRIVER SCHEDULING METHODOLOGIES IN USE

Overview of the Driver and Vehicle Scheduling Problem

The driver and vehicle scheduling processes involve four steps, including (1) trip building, (2) blocking, (3) runcutting, and (4) rostering. Trip building involves determining time tables for scheduling vehicles to arrive at specified locations at specified times, in order to come up with a master schedule. Once the coverage area is known, blocking is performed to create assignments describing the daily activity for a single vehicle. Next, runcutting assigns drivers to vehicles, determining the number of drivers that the transit agency will need in order to operate in compliance with the master schedule. The final step, rostering, involves grouping daily operator runs into a weekly run package and finding individual drivers to fulfill each role (Florida DOT, 2005).

The problem of driver scheduling involves the construction a legal set of shifts, including allowance of overtime, to cover the blocks in a particular vehicle schedule. A shift is the work scheduled to be performed by one driver in one day, while a block is a sequence of trips made by a single bus. Blocks can be divided between different drivers if they begin or end at a relief point, providing the opportunity for change. The goal is to make the schedule as efficient as possible, therefore minimizing the amount of changes that need to be made.

In determining the ideal schedule, local and national labor rules must be considered. These involve restrictions specified by the user, including, but not limited to, total time worked per day; total time worked per week; the length of time that may be worked without a meal break; the total spreadover, which is the duration between beginning and end of a shift; and/or number of days off per week (Wren and Rousseau, 1995).

The problem of vehicle scheduling involves determining the optimal route structure, span of service, and service frequencies for the transit agency and assigning vehicles to the routes. This process involves considering cycle times, number of vehicles, timed transfers, layover time and locations, recovery time, and any differences in weekday and weekend services (Wren and Rousseau, 1995). Knowing the blocks of work that must be satisfied, vehicles must be assigned to each of the blocks and given departure and arrival times with a goal of optimizing the number of hours the vehicle is transporting passengers, known as platform hours.

Developers of transit scheduling software have options when it comes to choosing how to perform the scheduling tasks. Many choose to schedule vehicles and drivers simultaneously, building blocks which satisfy the rules for shifts. Some algorithms have been designed to build shifts directly from units of bus work. This task is generally accomplished by first creating an efficient bus schedule, forming a driver schedule to cover all the bus work, and then re-writing the vehicle schedule so that the vehicles follow any restrictions caused by the drivers.

Solution Methods

During the 1960's almost all scheduling was still being done manually, but as the average age of schedulers increased and few people desired to enter the transit scheduling career field, the Urban Mass Transportation Authority (UMTA), now the Federal Transit Administration (FTA), directed research toward automating the scheduling process (USDOT, 1985). The earliest research evaluating the driver scheduling problem used heuristics since integer linear programming models were not available to solve models of realistic size. These methods attempted to simulate the work of manual schedulers. This research resulted in the RUCUS

Software developed by the Mitre Corporation and placed into operational use in 1975 by the Central New York Regional Transportation Authority (USDOT, 1985). The benefits of the automated system are the provision of more efficient schedules, reduction of staff requirements for conducting scheduling processes, the reduction of costs for both vehicles and operators, and provision of enhanced flexibility in the scheduling process (Florida DOT, 2005).

By 1980, it was determined that heuristics alone were not suitable for general use in arriving at an optimal solution that created an efficient use of resources and provided the best quality service for the customers. In response, research was directed at producing methods that combined heuristics and mathematical programming. Since then, a variety of programs and solution methods have been explored by different companies around the world.

Each of these companies attempts to solve the same problem with slight variations in methodology. Given a set of *m* pieces of work, each requiring a driver, together with a set of *n* possible driver shifts, the goal is finding a sub-set of the *n* shifts which together cover all the pieces of work and minimize either total cost or number of shifts (Wren and Rousseau, 1995). Since *m* and *n* are so large in real-life applications, the full problem must be restricted so a solution can be obtained. Different methods have been used to restrict the full problem, including (Wren and Rousseau, 1995):

- Allowing several shifts to cover any piece of work in the solution, relying on the optimization to keep the amount of over-cover to a minimum;
- Relaxing the problem to obtain a set of idealized shifts which ensure that there are enough shifts in the solution to cover the number of buses running in any period of time; and
- Forming drivers' shifts first by partitioning the day approximately into halves and then solving each part as a set-covering problem.

Current State of the Practice

This sub-section describes investigatory survey work performed by the research team to discern current trends in driver scheduling practice at the 20 largest public transportation agencies in the US.

Although scheduling of vehicles and drivers in the United States began in the public sector with the use of RUCUS software, this approach has become very primitive compared to the newer software developed in the private sector. The Federal Transit Authority (FTA) explained that they did not believe that FTA currently endorsed any specific program for vehicle and crew scheduling.

Based on 2004 counts of annual passenger trips from the American Public Transportation Association (APTA), the 20 largest transit operators in the United States were identified (APTA, 2006). Each transit agency was contacted by e-mail or telephone to determine which method they currently employed in performing vehicle and crew scheduling for routine transit service. Results from this survey are given in Table 1. Of the 15 agencies that responded, seven use TRAPEZE, six use HASTUS, and two use other methods.

Table 1. Scheduling Means Used by 20 Largest Transit Agencies in United States

Rank	Transit Agency	Means of Scheduling
1	New York City Transit (MTA NYCT)	HASTUS
2	Chicago Transit Authority (CTA)	HASTUS
3	Washington Metropolitan Area Transit Authority	TRAPEZE
4	Los Angeles County Metropolitan Transportation Authority	HASTUS
5	Massachusetts Bay Transportation Authority (MBTA)	HASTUS
6	Southeastern Pennsylvania Transportation Authority (SEPTA)	TRAPEZE
7	New Jersey Transit Corporation	CASS; converting to HASTUS
8	San Francisco Railway	N/A*
9	Metropolitan Atlanta Rapid Transit Authority (MARTA)	TRAPEZE
10	Maryland Transit Administration (MTA)	VISTA
11	King County DOT (Seattle)	TRAPEZE
12	Miami-Dade Transit (MDT)	TRAPEZE
13	Tri-County Metropolitan Transportation District of Oregon	HASTUS
14	San Francisco Bay Area Rapid Transit (BART)	Berkeley Simulation used for dispatching trains
15	MTA Long Island Railway	N/A*
16	Metropolitan Transit Authority of Harris County, Texas	N/A*
17	Denver Regional Transportation District (RTD)	TRAPEZE
18	Dallas Area Rapid Transit (DART)	TRAPEZE
19	Metro North- NYC	Taken over by MTA NYCT
20	GTJC	Taken over by MTA NYCT

*N/A signifies that the transit agency did not respond

Outside the United States, another major software program used to perform vehicle and crew scheduling is TRACS. This system has been implemented in bus companies throughout most of the United Kingdom.

The following sections briefly describe the philosophies and programming methodologies comprising the driver scheduling software systems discovered from the survey to be in use in the largest US and UK agencies today: TRAPEZE, HASTUS, and TRACS. This information was obtained through materials prepared for the software packages and from personal interviews conducted with leaders at the software development companies.

TRAPEZE (Fulton, 2006)

The TRAPEZE software for public transportation was developed by the Trapeze Group in order to develop better, more efficient services for transit providers. Trapeze acquired AUSTRICS in 2004, a company based in Australia, providing access to software packages AUSTRICS had developed for network planning, and vehicle and crew optimization. The TRAPEZE software uses an optimization approach to handle the vehicle and crew scheduling problem, and is capable of solving the vehicle and crew scheduling problems together or separately, depending on the needs of the user.

According to Dave Fulton, Director of Fixed Route Products at Trapeze, in solving for the optimal schedule, the objective function for both vehicles and drivers is to minimize costs. Driver costs are a function of work content, rate of pay, and any variable and fixed overhead costs. Extra expenses that result from scheduling include preparing the bus and performing safety checks before leaving, the time for the driver to travel to relief points to switch off to another driver, clear time to bring in the bus and empty the fare box. In regards to the individual drivers, it is important to consider if they are guaranteed a minimum number of hours of work, how much they are paid in overtime for long days, and how much break or meal time they receive. Vehicle costs are based on mileage, time, and fuel cost, so an efficient vehicle schedule maximizes the revenue hours, which is the time actually transporting passengers, while minimizing deadheads and layover time within the constraints of geography and operational policy.

Before solving for an optimal solution, the user is allowed to specify both hard and soft constraints. Hard constraints detail the labor union rules of operation, management controls, and vehicle and operator limitations, while soft constraints control quality control aspects. In terms of drivers, they can set the maximum or minimum number of drivers they wish to have on staff and can choose if they want their solution to decrease shifts, decrease pay, decrease work time, or decrease overtime pay. In this step, they can implement labor union agreements and define part-time vs. full-time employees. Additionally, in terms of vehicles, the transit agency can determine the maximum or minimum number of buses available to perform the services and the maximum or minimum buses stored at a particular depot.

To solve the driver and vehicle scheduling problem, there are three stages that are implemented. The first stage involves subdividing the large problems into homogeneous sets, modeling the constraints using linear programming, and using successive decomposition to determine the most appropriate relief points. The second stage involves combining the sub-problems, using linear programming to extract shifts, and when the problem is small enough, use of integer programming to complete the solution. The final stage is to map out the solution to determine the number of consecutive pieces of work that can be performed by the same bus, and use of assignment algorithms to shift the segments to improve the solution.

HASTUS

The HASTUS scheduling software was developed by GIRO, Inc., aided by research at the Université de Montréal, and is used by over 250 bus and rail companies in the world (HASTUS, 2006). The initial software was developed 25 years ago as a collection of optimization algorithms. This optimization tool has evolved to adapt to the needs of individual transit companies and is now capable of solving vehicle and crew scheduling problems either separately or combined (Fleurent, 2006). It begins by cutting the blocks into pieces of work, then matches pieces of work to form drivers' schedules, then assigns actual operators, and finally improves the solution heuristically (Wren and Rousseau, 1995). The first three steps are

performed by three subsystems, HASTUS-Bus, HASTUS-Macro, and HASTUS-Micro, respectively (Rousseau and Blais, 1985). Although the subsystems are fully integrated, HASTUS-Macro can be run alone without involving other parts of the system. It is considered an "intelligent-interactive" system since it allows interplay between mathematical optimization tools and human guidance input. The interaction of the subsystems can be seen below in Figure 1.

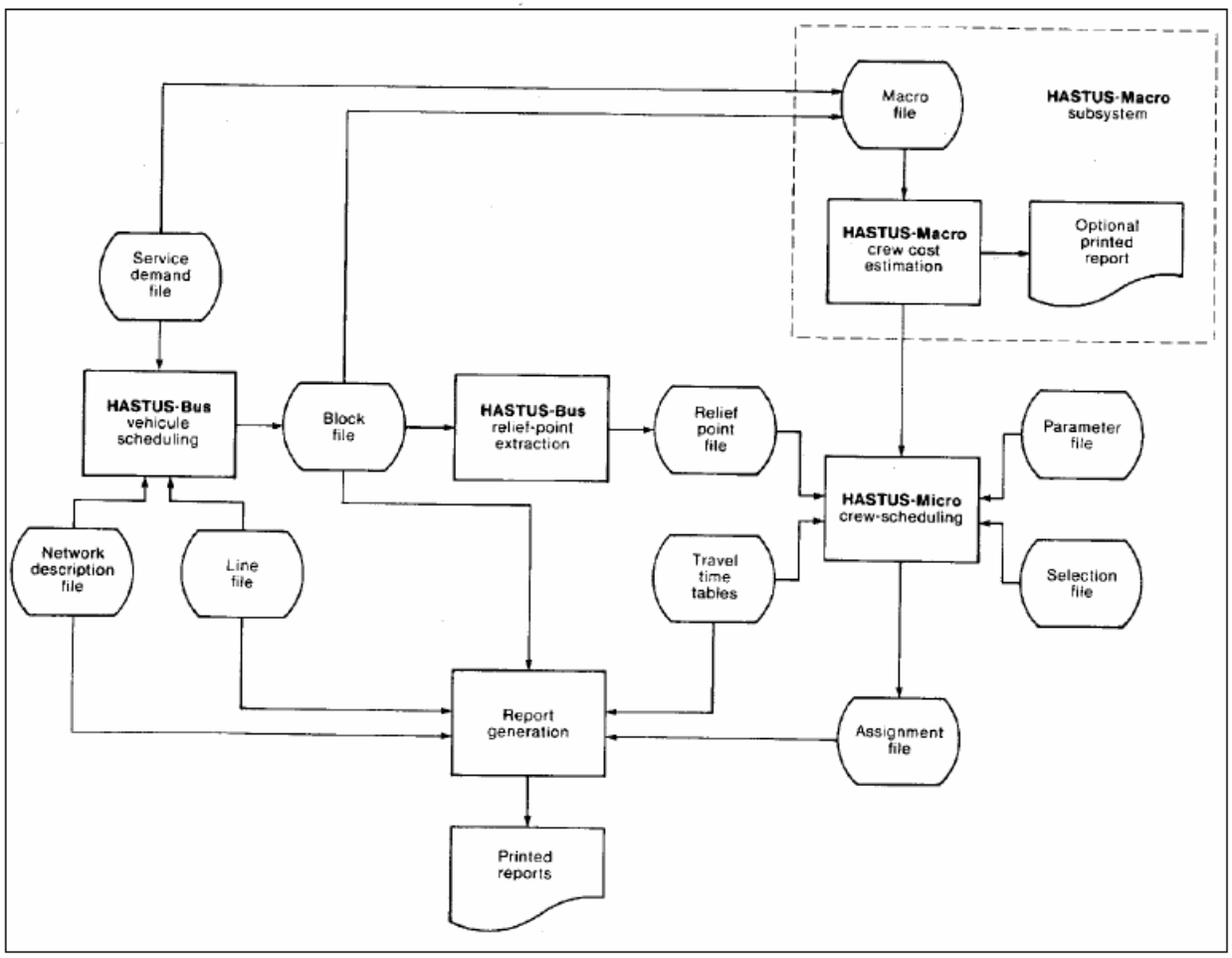

Figure 1. Interaction of HASTUS Subsystems (Rousseau and Blais, 1985)

According to Charles Fleurent, Manager of Optimization Algorithms at GIRO, the objective function for both vehicles and crews is to minimize costs. In terms of driver scheduling, the total time that has to be paid to drivers, including fringe benefits and special bonuses defined in the union agreement are considered (Fleurent, 2006). The driver must be paid not only for the actual time transporting riders, but also for travel to/from a garage or to a different terminus. For vehicle scheduling, the main cost comes from the number of vehicles that are necessary to cover the set of timetabled trips offered by the transit company (Fleurent, 2006). Penalties can be assessed for encouraging certain types of unfavorable duties such as early start time, late end times, or long spreads (Rousseau and Blais, 1985).

The constraints imposed on the system come from union agreements and company specific rules that must be enforced. This includes maximum spread or working time duration

for duties, maximum capacities of garages for vehicles, and maximum number of split duties for a crew (Fleurent, 2006).

To solve the scheduling problem, several heuristic and exact optimization approaches are used, most based on mathematical programming methods. Linear programming is used in the HASTUS-Macro to determine an approximate cost of covering a given vehicle schedule with manpower under the terms of the operators' contract. It searches for the set of shifts of minimum cost that will provide the number of drivers required during each period and satisfy all constraints required by the user. Based on this analysis, it outputs the number of drivers required, number of hours assigned, number of hours worked, number of trippers, number of hours paid in overtime, number of hours paid in break, total cost per assigned hour, and total cost (Rousseau and Blais, 1985).

The HASTUS-Bus problem requires input of the overall system structure and the structure of the transit lines which can be used to build all of the trips. This procedure results in an output containing the different blocks that a vehicle is assigned to based on the frequencies that the user requests (Rousseau and Blais, 1985).

The final step involves HASTUS-Micro which produces the detailed crew schedule and computes its exact cost. The relief time from vehicle scheduling, the time allocated and time paid to move operators between relief points, the union contract specifics, the specification of any duties, and the types of duties that should be penalized are all input into the system (Rousseau and Blais, 1985). This step results in an assignment file, listing all the duties for each of the operators.

TRACS (Fores and Proll, 1998)

The TRACS scheduling system was developed by the Transport Scheduling Unit at the University of Leeds and is currently used by many bus operators in the United Kingdom. The system prefers to first build vehicle schedules to cover sets of predetermined journeys, which gives the user flexibility of making changes to the vehicle schedules. Driver shifts are then assigned to the blocks of work created.

There are three stages to determining the optimal driver scheduling solution. First, a set of shifts which are valid according to labor agreement rules is generated. A subset of shifts is chosen heuristically due to the many million potential shifts. If necessary, the size of the generated shift set is reduced using integer linear programming. Once the set of shifts is determined, a subset is selected which covers all the vehicle work at a minimal cost.

The objective function of this system can vary but generally constitutes minimizing the number of drivers or minimizing the costs and penalties. Costs involve wages and overtime. Penalty costs are then added for (a) meal breaks at undesirable times, (b) shifts separated by long breaks, (c) shifts beginning or ending at undesirable times, (d) shifts that are longer or shorter than preferred, (e) shifts with a mixture of routes, (f) shifts that are undesirable in terms of spreadover or work content, (g) shifts starting or ending with meal breaks, (h) shifts involving a combination of vehicles, and (i) instances when a driver has to travel as a passenger (Kwan et al., 2004).

The constraints on the system are dictated by the user, but they include ensuring that every piece of work is covered by at least one driver. The user can also set an upper or lower limit on the number of shifts, hours worked, or the number of overtime hours allowed.

Once the objective function and the constraints have been determined, it is then possible to solve for the optimal solution. This task is generally accomplished using a dual steepest-edge

method for sets containing less than 30,000 shifts or with column generation for large shift sets. Column generation works by generating a shift subset, creating an initial solution and forming an initial shift subset to ensure each piece of work is covered, solving linear programming relaxation over the current shift subset, adding shifts that will improve the solution, and repeating this process until no more favorable shifts can be found. The integer solution is then determined using branch-and-bound by finding possible solutions and ensuring that they fit the cost criteria.

Summary of Current Methodologies

When a transit agency has reached a sufficiently large fleet and employment size, it becomes obvious that shifting from a manual to an automated system is necessary to perform driver and vehicle scheduling in way that will provide the most efficient and cost effective service to the passengers. There are three primary programs currently in use for driver and vehicle scheduling: TRAPEZE, HASTUS, and TRACS. All of these are optimization programs with similar objective functions. Each program's objective is to minimize cost, allowing users to enter any constraints particular to their agency or labor union agreements.

The automated systems allow for driver and vehicle scheduling to be done simultaneously. This process is completed by first determining the ideal routes, then using linear programming to fit shifts of work to the blocks ensuring that each block is assigned a vehicle and driver, and then finally iterating to come up with an ideal solution. All of these methodologies have allowed for significant cost savings for public transit agencies in terms of the amount of money spent on drivers, vehicles, and manpower to perform the scheduling task itself.

Although there are many benefits to the automated system, there is still room for improvement in these methodologies. Due to the large number of possible blocks and shifts, it is not possible for any of these programs to successfully solve the optimization problem without resorting to decomposition of the original problem. This decomposition results in reducing the original problem to a smaller sample by removing any shifts believed to be inefficient. By removing these shifts, the solution is not guaranteed to be optimal since all possible shifts have not been considered. The output of these software programs list the number of vehicles and drivers to satisfy their objective functions, but it does not appear that they consider the quality of service in terms of the extra (spare) drivers needed to cover contingencies in case of illness or drivers not reporting to work, or the number of extra buses the transit agency should have available in case of any maintenance issues. Implementation of these improvements could result in a methodology providing solutions that are more optimal for transit agencies.

CHAPTER 3: NEEDED EXTENSIONS AND GENERALIZATIONS

While the public transportation driver and vehicle scheduling algorithms and software in use today serve a vital role in managing transit system demands, some voids exist in their ability to adequately fulfill the needs of all systems. This chapter elaborates upon a few of these gaps, focusing primarily on the operational differences between transit agencies and provision of flexibility in managing these differences.

Vehicle Differentiation

Most transit agencies today offer multiple types of service to meet the needs of their communities. Even within a mode type, there are usually various types of service and the associated vehicles appropriated to the unique needs of each service type. For instance, although the "bus" mode brings to mind a particular type of service structure and vehicle type, defining bus service homogeneously as that served with city buses on standard fixed-route operation is generally inaccurate. Many bus fleets will consist of numerous vehicle types, including standard buses, large motorcoaches for commuter travel, and perhaps smaller vehicles serving special needs travelers or flexible route service. Even within the definition of "standard" buses, numerous variations exist in vehicle lengths and operational configurations.

The disparities between these vehicle types introduce complexities to the vehicle scheduling problem since the vehicles cannot be arbitrarily assigned to a particular type of service or route. Adding perhaps substantially more complexity, drivers are typically trained to operate a specific vehicle type and therefore must be assigned to vehicles for which they have been trained. These operational stipulations are set forth in driver contract specifications.

Contract Specifications

Any algorithm proposed to improve upon the current state of the practice must readily be capable of dealing with these and essentially any other set of contract specifications. Covering these specifications comprehensively introduces greater complexity regarding constraints to the problem, particularly in preparing the algorithm to handle the unique constraints of a particular agency. Ideally, a full review of agency contracts would provide the broadest knowledge of potential caveats to be covered by the scheduling solution; however, such a comprehensive contract review is somewhat impractical, between resource expenditure and issues regarding confidentiality. Instead, a more likely alternative may include surveying a sample of agencies to discern both standard and atypical specifications to be covered.

Related to these contract specification policies, a wide variety of operational conditions exist for public transportation agencies; these policies, while following similar structures, can vary substantially between operators. Policies such as those regarding driver relief, meal and break times, and provisions for inclusion and payment of spare drivers, among many other choices and constraints, must be captured and simulated within the algorithm, although the details thereof will differ from agency to agency. Again, a comprehensive survey of current policies would be ideal, albeit impractical. Rather, a surveyed sampling may provide the best alternative to a full review of all agencies, allowing for observation of trends that may be made adjustable within the algorithm to provide enhanced flexibility in application.

Summary of Needed Improvements

In creating an algorithm and scheduling system with universal applicability, it is essential to provide flexibility within the program allowing for the substantial differences between operational standards at transit agencies. The greatest source of variability in operational standards between transit agencies stems from conditions set forth in driver contract specifications. Adding to the complexity of the problem whether vehicle and driver scheduling are accomplished together, separately, or iteratively, differentiation between vehicle types and driver training feature prominently when developing schedules and work blocks.

Establishing a scheduling algorithm that captures these potentially infinite differences in a flexible and readily applied manner is a challenge. To improve upon the current state of the practice, undertaking a review of existing policies could aid in providing the most useful enhancements to software and algorithms in use today. Through completion of such a review, whether done comprehensively or selectively, trends may be uncovered and bounds ascertained on these trends that may then reduce the burden of creating an infinitely adjustable constraint set.

CHAPTER 4: CODE DEVELOPMENT

Current Code: A Benchmark for Future Work

The programming code developed in the course of this research provides a benchmark for future work in this area and serves as a reference point for building code for more extensive transit systems. This code is included in this report in Appendix A. Using the C++ programming language, this base code provides complete enumeration of the solution space for the driver scheduling problem for up to ten drivers and the total costs corresponding to these enumerations. Future work to expand upon these programming efforts could make use of this base code by following the same format for larger systems with more than four drivers, or by expanding the repetitive portions of the code ("loops") to allow for greater numbers of drivers.

As it stands now, the inputs utilized in the base code include the number of work hours to be covered through the design of driver shifts, and the maximum number of drivers to be considered in the complete enumeration of potential solutions. These basic inputs provide flexibility when considering the size of the public transportation system. The output of the existing code provides the least-cost combination of driver hours in terms of work blocks, and the total cost of that option. In future work, programming should be developed in which the work blocks are combined, when possible, into driver shifts.

The objective of the driver scheduling problem is, as with many problem formulations, to minimize the total value of the cost function, as stated in Equation 1.

$$\min c \tag{1}$$

where $c = f$(salary, overhead, penalties).

In order to avoid or minimize the occurrence of unwanted conditions, "penalty" costs may be added as required to a problem formulation so that allowing the undesirable conditions to develop creates, if not a prohibitively high total cost, one that places the problem solution achieving this inflated cost farther from a specified acceptable threshold. The weights of these penalties can be adjusted from those given in the existing code to reflect the relative undesirability associated with the conditions.

The conditions captured by these means in the base code currently include:

- Work blocks longer than ten hours; due to labor rules and the like, limits have been established concerning the limits on the maximum number of consecutive hours a driver may be assigned within a single piece of work. In the code provided in Appendix A, these overly long shifts are weighted as 100 (via the *Lshift* variable), which is subsequently added to the total cost function as defined in Equation 2.
- Work blocks of less than one hour; these fractional work pieces, while within the feasible solution space, are unreasonable as true optima. The penalty associated with these undesirably short shifts (*Sshift* variable) is 30 in the code included in Appendix A.
- Overhead costs associated with having multiple drivers; In the code provided, the penalties for these overhead hours are assigned dependent upon the number of drivers (three to ten) as first indicated by the user. These overhead costs (*Overhead* variable) represent the fixed overhead cost, separate from salary values, associated with

maintaining a system with multiple drivers; the values increase the total cost by 10 units over the case with one less driver in the system.

To determine the least-cost option, a cost function must be developed to assign a total cost to each option. The cost function used in the developed code is as a simple sum of the included factors.

$$Cost = Salary + Overhead + Lshift + Sshift \quad (2)$$

where
Salary represents the summation of work blocks multiplied by the hourly wage
Overhead indicates the fixed cost associated with maintaining additional drivers over a fewer-driver alternative
Lshift is the penalty assigned to over-long work blocks (more than 10 hours in duration)
Sshift is the penalty associated with work pieces of less than one hour in duration

The current code provides output illustrating (1) the full enumeration of work block combinations, (2) the costs associated with each of these combinations, and (3) the least-cost option. The enumerated combinations and their costs are stored in arrays, as are the least-cost alternatives, in increasing order.

Summary of Recommended Programming Constraints

While numerous constraints could be formulated and applied to this problem, those recommended for immediate future inclusion are highlighted here and should be augmented with those discussed in Chapter 3. In this section, the suggested constraints are separated when possible into mathematical constraints and contractual constraints.

Mathematical Constraints

Provisions should be made that prevent the user from entering erroneous or unrealistic data. For instance, knowing the limits on working hours from contract specifications, the user should be warned and queried for new input should the provided number of drivers be physically unable to cover the total work hours given; as an illustratively small example, if five drivers are available and each may work up to eight hours in a day, then no more than 40 hours may be covered by this combination of drivers.

Another unrealistic scenario that must be avoided in deriving solutions to the scheduling problem focuses on a driver's inability to take on more than one work block at a time. Care must be taken that the blocks of work organized in the scheduling algorithm are not only contiguous in time (to the extent possible), but also contiguous in space. As an example, if a driver is scheduled to end one route and begin another, the two routes must connect or be connected by a deadhead period of reasonable duration. This contingency may be most easily handled by combining vehicle scheduling with driver scheduling. Similarly, when combining work blocks to create shifts, the work blocks must be combined in such a way as to not overlap in time.

Contractual Constraints

In the absence of contractual constraints, the driver scheduling problem would simplify to a matter of paying drivers for their hours worked. Making the scheduling problem much more

complex, contractual constraints relating to labor rules and union provisions must, however, be considered, as highlighted in Chapter 3.

The constraints to be included in any transit agency's scheduling algorithm may be basic to all systems or distinct to a particular operator, and are comprised of those specifications devised as part of national and localized labor rules. To make the scheduling algorithm sufficiently general to apply to any agency's operations, it must be designed for flexibility in incorporating these contractual constraints and developing scheduling solutions that reflect inclusion of the provisions set forth in relevant labor contracts.

Basic constraints included in any transit system's labor contracts involve limitations on the total time a driver can work in a single day, the spreadover of work conducted in a single day (the duration between the start and end of the driver's work day), the number and extent of break periods taken during a work day (for meals and/or rest) and the work duration between these break periods, and the total time worked per week, to list only a few. Although these stipulations may be common to all agencies, the specifics of the constraints could vary—the algorithm must be flexible enough to accommodate all variations.

Many other issues arise when broader labor agreements are considered. Some drivers work full time (40-hour workweeks), while others may be part-time or drive split shifts. These work arrangements complicate matters because while in some contractual agreements, a driver may be paid only those hours in which he or she drives, drivers in other systems may accrue wages as long as they report to work. In this latter case, a driver with a split shift (working a few hours in the morning and, following a multi-hour break, a few hours in the afternoon) may be paid for the entire spreadover, regardless of whether he or she was driving a vehicle the entire time. The former case reduces costs for the transit operator, but labor stipulations do not always allow for such an arrangement. As such, the definition of the least-cost scheduling alternative may be impacted.

Logistical issues involving coordination between driver schedules and their necessary relationship to vehicle schedules heighten the intricacy of solving the driver scheduling problem and illustrate its interaction with the vehicle scheduling problem. These logistical issues may or may not be tied to contractual agreements, depending on the nature of the conflict. One such logistical issue concerns driver relief. In one case, a driver could end his or her shift at some point along a route where another driver takes over and begins his or her shift. Conversely, depending on the situation, it may be more logical for the driver to convey the vehicle back to the dispatch location and allow another driver to begin his or her shift from that point, rather than assigning the drivers to switch positions mid-route.

Inclusion of spare drivers introduces another layer of complexity. To cover work blocks left unmet by absentee or sick drivers, additional drivers (typically about ten percent of the required number of scheduled drivers) are required to report to and remain available at the dispatch base each day the transit system offers service. Depending on the number of absentee drivers on a given day, these spare drivers may not spend their work shifts driving but are still paid their full wages and benefits as though they had in fact driven.

Ultimately, creating a scheduling algorithm which generates solutions that account for all potential contractual constraints requires knowledge of typical labor contracts as well as those with more unusual stipulations. Building this knowledge and applying it to a scheduling algorithm necessitates a review of existing labor contracts, the discovery of which may be curtailed by privacy and proprietary issues, not to mention the resource expenditure required to pursue such a review.

Future Steps

As mentioned, the work done thus far represents a benchmark on which future work may build. The following section elaborates on future work that should be performed to enhance this research.

Driver scheduling and vehicle scheduling feed each other since vehicles cannot operate without drivers and drivers are charged with operating the transit vehicles—in this way, the problems cannot be independent and are rather functions of each other.

If the driver and vehicle scheduling tasks are undertaken out separately, then following the development of the driver schedule solution, the output must be carried over to the vehicle scheduling algorithm. It is likely that incongruencies would develop between the driver schedule and potential vehicle schedules, thus necessitating an iterative solution to the full scheduling problem, given the correlation between driver and vehicle scheduling.

Handled separately, the vehicle scheduling solution space is likely much smaller than the driver scheduling solution space. Due to logistical and mechanical issues with vehicle operation, the vehicle scheduling solution space can be minimized by eliminating unreasonable vehicle schedules. Examples of issues affecting the vehicle scheduling solution and affording the reduction of its solution space size include scheduled re-fueling, time required for engines to warm or cool, maintenance and cleaning, among other concerns.

Ideally, the driver and vehicle scheduling problems should be combined, solving the problems iteratively, if not simultaneously. In so doing, the product achieved by combining the problems generates an enormous pooled solution space, even after reducing the size of the vehicle scheduling solution space, but likely containing some number of overlapping solutions. The optimal driver scheduling solution is actually optimal only for the vehicle scheduling solution that has been specified for the driver scheduling optimization, or that developed in solving the problems simultaneously. Additionally, further investigation should delve into examining how the vehicle scheduling solution affects the driver scheduling solution.

Lastly, once algorithms have been developed to solve the driver and vehicle scheduling problems iteratively or simultaneously, heuristic methods should be applied to the problem to facilitate solution comparisons to ascertain the fastest and most reliable solution technique.

Summary of Work Performed

In the course of this project, the research team investigated background materials surrounding the driver scheduling problem and conducted a survey of existing methodologies in use today at the 20 largest transit agencies in the United State and United Kingdom.

Additionally, the team developed a benchmark code for driver scheduling which provides full enumeration of the solution space for as many as nine drivers. Future endeavors should build on this work to reach transit systems of much larger size and complexity.

The greatest difficulty in establishing a complete approach to driver scheduling involves the inclusion of labor contract provisions in accounting for scheduling constraints. The successful algorithm would be flexible enough to allow agency planners to generate schedules contingent on any set of contractual constraints. Provision of this flexibility hinges on comprehensive knowledge of an agency's labor contract stipulations; an algorithm with universal applicability would require similar knowledge of labor contracts organized at all potential using agencies.

Due to the perceived need for significant resources, time, and materials required to reach conclusive improvements on the algorithms established in the private sector (as highlighted in Chapter 2), the research team felt that, lacking a strong need to develop a next-generation scheduling algorithm in the public domain, further pursuit of the objective should cease. However, most work toward this end has, to this point, been conducted abroad, and other research teams should not be discouraged from pursuing this goal.

CHAPTER 5: CONCLUSIONS

Vehicle and driver scheduling in public transportation systems are complex, interconnected problems requiring extensive knowledge of transit operations and legal labor agreements. The problems should be solved iteratively or simultaneously, because finding independent solutions to each does not necessarily guarantee a cohesive solution for the system as a whole.

Currently, three primary software programs and their associated algorithms are in use at the largest transit agencies in the United States and United Kingdom. These have been developed in the private domain and are proprietary in nature. The research here sought to develop a public-domain algorithm to find solutions to the vehicle and driver scheduling problems.

In the course of this research, the team conducted a background review of relevant literature surrounding the public transportation scheduling problems, and surveyed the 20 largest transit agencies to determine the scheduling methodologies in use at each. The team then investigated the most widely-used scheduling software programs through additional literature reviews and personal interviews with algorithm and software developers. A benchmark code was developed for small systems that may be expanded upon through future research in this area.

The vehicle and driver scheduling problems are made more complex through many constraints placed on both the definition of the mathematical problem and issues arising from logistical and labor ramifications. The most pervasive of these constraints affecting the scheduling problems, particularly the driver scheduling problem, are generated from the contractual labor agreements between the unionized driver work force and the transit operator. While similarities exist in these agreements from one agency to the next, the details of the stipulations can vary substantially. Ideally, a newly developed scheduling algorithm would be capable of flexibly accommodating any and all of these constraints. However, conducting a comprehensive survey of public transportation agencies to ensure inclusion of all potential labor rules would require substantially greater resources than those to which the research team has access. Furthermore, it is not clear that a demand exists for a public-domain scheduling product.

APPENDIX

Developed Driver Scheduling Code

```
#include <iostream>
using namespace std;

int main ()
{
  int D=10;
  int H=10;
  int a, b, c, d, e, f, g, h, j, k;

//If there are H hours and D drivers, what are the combinations?

cout << "How many hours?" << endl;
cin >> H;
cout << "Enter number of drivers between 3 and 10" << endl;
cin >> D;

if (D<3){
  cout<<"number of drivers too low"<<endl;
  return 0;}

if (D==9){
j=0;}
  else if (D==8){
    j=0; h=0;}
    else if (D==7){
      j=0; h=0; g=0;}
      else if (D==6){
        j=0; h=0; g=0; f=0;}
        else if (D==5){
        j=0; h=0; g=0; f=0; e=0;}
          else if (D==4){
          j=0; h=0; g=0; f=0; e=0; d=0;}
            else if (D==3){
            j=0; h=0; g=0; f=0; e=0; d=0; c=0;}


int overhead; // Cost of
int salary; // Cost of Hourly Wage
int lshift; // Cost Penalty for shift greater than 10 hours
int sshift; // Cost Penarly for short shift (1 hour)
int cost;  // Sum of costs
```

```
int hourly; // Hourly Salary

cout << "What is the hourly driver salary? "<< endl;
cin>>hourly;

int LC_cost=0;//retain the least cost of the least cost condition
int LC_ak[10];//retain the variables in an array (as a to k) of least cost condition


if (D==3)
{
        for(a=0;a<=H;a++)
        {
          for(b=0;b<=H-a;b++)
          {
                k=H-a-b-c-d-e-f-g-h-j;
                if (a>0 && b>0 && k>0)
                  overhead = 30;
                if ((a>0 && b>0 && k==0)|| (a==0 && k>0 && b>0) || (a>0 && k>0 && b==0))
                  overhead = 20;
                if ((a>0 && b==0 && k==0)|| (a==0 && k>0 && b==0) || (a==0 && k>0 && b==0))
                  overhead = 10;
                if (a>10 || b>10 || k>10)
                  lshift = 100;
                else
                  lshift = 0;
                if (a==1 || b==1 || k==1)
                  sshift = 30;
                else
                  sshift = 0;

                salary = ((a+b+c+d+e+f+g+h+j+k)*hourly);

                cost = salary + overhead + lshift + sshift;

                                        cout<<"Driver hours are "<<a<<" " <<b<<" "<<c<<" "<<d<<" "<<e;
                                        cout << " "<<f<<" "<<g<<" "<<h<<" "<<j<<" "<<k<< " and total cost
is ";
                                        cout << cost << endl;
                if (a==0 && b==0)//give LC_cost the starting value as the same as the first cost value+1
(plus 1 is to avoid the situiation which
```

```
                {                   //the first condition is least cost
                        LC_cost=cost+1;
                }
                if (LC_cost>cost)//compare LC_cost to cost and retain the least cost condition to these
variables
                {
                        LC_cost=cost;
                        LC_ak[0]=a;
                        LC_ak[1]=b;
                        LC_ak[2]=c;
                        LC_ak[3]=d;
                        LC_ak[4]=e;
                        LC_ak[5]=f;
                        LC_ak[6]=g;
                        LC_ak[7]=h;
                        LC_ak[8]=j;
                        LC_ak[9]=k;
                }

          }
        }

        cout<<endl;
        cout<<"One of the least cost condition is:"<<endl;//print cout the least cost condition
        cout<<"Driver hours are "<<LC_ak[0]<<" " <<LC_ak[1]<<" "<<LC_ak[2]<<" "<<LC_ak[3]<<" "<<LC_ak[4];
                                        cout << " "<<LC_ak[5]<<" "<<LC_ak[6]<<" "<<LC_ak[7]<<"
"<<LC_ak[8]<<" "<<LC_ak[9]<< " and total cost is ";
                                        cout << LC_cost << endl;
}

if (D==4){
for(a=0;a<=H;a++){
  for(b=0;b<=H-a;b++){
    for(c=0;c<=H-a-b;c++){
      k=H-a-b-c-d-e-f-g-h-j;
                if (a>0 && b>0 && c>0 && k>0)
                  overhead = 40;
                if ((a>0 && b>0 && c>0 && k==0)|| (a==0 && k>0 && b>0 && c>0) || (a>0 && k>0 && b==0 && c>0)
|| (a>0 && b>0 && k>0 && c==0) )
                  overhead = 30;
```

```
                if ((a>0 && c>0 && b==0 && k==0)|| (a==0 && c>0 && k>0 && b==0) || (a==0 && c>0 && k>0 &&
b==0) || (a>0 && b>0 && c==0 && k==0) || (a==0 && c==0 && b>0 && k>0) || (a>0 && k>0 && b==0 && c==0))
                  overhead = 20;
                if ((a>0 && b==0 && c==0 && k==0) || (b>0 && a==0 && c==0 && k==0) || (c>0 && a==0 && b==0
&& k ==0) || (k>0 && a==0 && b==0 & c==0))
                   overhead = 10;
                if (a>10 || b>10 || c>10 || k>10)
                  lshift = 30;
                else
                  lshift = 0;
                if (a==1 || b==1 || c==1 || k==1)
                  sshift = 30;
                else
                  sshift = 0;

                salary = ((a+b+c+d+e+f+g+h+j+k)*hourly);

                cost = salary + overhead + lshift + sshift;

                                        cout<<"Driver hours are "<<a<<" " <<b<<" "<<c<<" "<<d<<" "<<e;
                                        cout << " "<<f<<" "<<g<<" "<<h<<" "<<j<<" "<<k<< " and total cost is
";
                                        cout << cost << endl;


    }}}
                   }

if (D==5){
for(a=0;a<=H;a++){
  for(b=0;b<=H-a;b++){
    for(c=0;c<=H-a-b;c++){
      for(d=0;d<=H-a-b-c;d++){
        k=H-a-b-c-d-e-f-g-h-j;

                cout<<"Driver hours are "<<a<<" " <<b<<" "<<c<<" "<<d<<" "<<e;
                cout << " "<<f<<" "<<g<<" "<<h<<" "<<j<<" "<<k<<endl;}
                         }}
                      }}

if (D==6){
```

```
for(a=0;a<=H;a++){
  for(b=0;b<=H-a;b++){
    for(c=0;c<=H-a-b;c++){
      for(d=0;d<=H-a-b-c;d++){
        for(e=0;e<=H-a-b-c-d;e++){
          k=H-a-b-c-d-e-f-g-h-j;

                cout<<"Driver hours are "<<a<<" " <<b<<" "<<c<<" "<<d<<" "<<e;
                cout << " "<<f<<" "<<g<<" "<<h<<" "<<j<<" "<<k<<endl;}
                       }}
                     }}
                   }

if (D==7){
for(a=0;a<=H;a++){
  for(b=0;b<=H-a;b++){
    for(c=0;c<=H-a-b;c++){
      for(d=0;d<=H-a-b-c;d++){
        for(e=0;e<=H-a-b-c-d;e++){
          for(f=0;f<=H-a-b-c-d-e;f++){
            k=H-a-b-c-d-e-f-g-h-j;

                cout<<"Driver hours are "<<a<<" " <<b<<" "<<c<<" "<<d<<" "<<e;
                cout << " "<<f<<" "<<g<<" "<<h<<" "<<j<<" "<<k<<endl;}
               }}
             }}
           }}

if (D==8){
for(a=0;a<=H;a++){
  for(b=0;b<=H-a;b++){
    for(c=0;c<=H-a-b;c++){
      for(d=0;d<=H-a-b-c;d++){
        for(e=0;e<=H-a-b-c-d;e++){
          for(f=0;f<=H-a-b-c-d-e;f++){
            for(g=0;g<=H-a-b-c-d-e-f;g++){
              k=H-a-b-c-d-e-f-g-h-j;

                cout<<"Driver hours are "<<a<<" " <<b<<" "<<c<<" "<<d<<" "<<e;
                cout << " "<<f<<" "<<g<<" "<<h<<" "<<j<<" "<<k<<endl;}
                             }}
```

```
                        }}
                      }}
                    }
```

```
if (D==9){
for(a=0;a<=H;a++){
  for(b=0;b<=H-a;b++){
    for(c=0;c<=H-a-b;c++){
      for(d=0;d<=H-a-b-c;d++){
        for(e=0;e<=H-a-b-c-d;e++){
          for(f=0;f<=H-a-b-c-d-e;f++){
            for(g=0;g<=H-a-b-c-d-e-f;g++){
              for(h=0;h<=H-a-b-c-d-e-f-g;h++){
                k=H-a-b-c-d-e-f-g-h-j;

                cout<<"Driver hours are "<<a<<" " <<b<<" "<<c<<" "<<d<<" "<<e;
                cout << " "<<f<<" "<<g<<" "<<h<<" "<<j<<" "<<k<<endl;}
                }}
             }}
           }}
         }}
```

```
if (D==10){
for(a=0;a<=H;a++){
  for(b=0;b<=H-a;b++){
    for(c=0;c<=H-a-b;c++){
      for(d=0;d<=H-a-b-c;d++){
        for(e=0;e<=H-a-b-c-d;e++){
          for(f=0;f<=H-a-b-c-d-e;f++){
            for(g=0;g<=H-a-b-c-d-e-f;g++){
              for(h=0;h<=H-a-b-c-d-e-f-g;h++){
                 for(j=0;j<=H-a-b-c-d-e-f-g-h;j++){
                  k=H-a-b-c-d-e-f-g-h-j;

                cout<<"Driver hours are "<<a<<" " <<b<<" "<<c<<" "<<d<<" "<<e;
                cout << " "<<f<<" "<<g<<" "<<h<<" "<<j<<" "<<k<<endl;}
                 }}
              }}
```

```
            }}
          }}
}



 return 0;
}
```

REFERENCES

Wren, A. and Rousseau, J.M. *Bus Driver Scheduling- An Overview*, University of Leeds Report 93.31, 1993, Computer-Aided Transit Scheduling, pp. 173-187, 1995.

"Transit Scheduling: Basic and Advanced Manuals". *TCRP Report 30*. Transit Cooperative Research Program. Transportation Research Board of the National Academies, Washington, DC. 1998.

Banihashemi, Mohamadreza and Ali Haghani. "Optimization Model for Large-Scale Bus Transit Scheduling Problems". *Transportation Research Record: Journal of the Transportation Research Board, No. 1733*. Transportation Research Board of the National Academies, Washington, DC. 2000.

Lourenço, Helena R., José P. Paixão, and Rita Portugal. "Multiobjective Metaheuristics for the Bus Driver Scheduling Problem". *Transportation Science*. The Institute for Operations Research and the Management Sciences (INFORMS). Volume 35, No. 3, August 2001.

Abbink, Erwin, et al. "Reinventing Crew Scheduling at Netherlands Railways". *Interfaces*. The Institute for Operations Research and the Management Sciences (INFORMS). Volume 35, No. 5. September-October 2005.

Fischetti, Matteo, et al. "A Polyhedral Approach to Simplified Crew Scheduling and Vehicle Scheduling Problems". *Management Science*. The Institute for Operations Research and the Management Sciences (INFORMS). Volume 47, No. 6, June 2001.

Haase, Knut et al. "Simultaneous Vehicle and Crew Scheduling in Urban Mass Transit Systems". *Transportation Science*. The Institute for Operations Research and the Management Sciences (INFORMS). Volume 35, No. 3, August 2001.

20 Largest Public Transit Agencies, 2003. American Public Transportation Association (APTA). Accessed 15 October 2006.
http://www.apta.com/research/stats/overview/20largest.cfm.

State of Florida Department of Transportation. *Fixed Route Transit Scheduling in Florida: The State of the Industry*. March 2005.

Fulton, Dave. Trapeze Group, E-mail Correspondence, 28 November 2006.

HASTUS for Transit Scheduling, GIRO, Inc. Accessed 4 October 2006.
http://www.giro.ca/English/HASTUS/transit_scheduling.htm

Fleurent, Charles. GIRO, Inc., E-mail Correspondence, 30 October 2006.

Rousseau, J.-M. and Blais, J.-Y. *HASTUS: An Interactive System for Buses and Crew Scheduling*, Computer Scheduling of Public Transport 2, J.-M. Rousseau (editor), North-Holland, pp. 45-60, 1985.

Fores, S. and Proll, L. *Driver Scheduling by Integer Linear Programming- The TRACS II Approach*, University of Leeds Report 98.01, 1998.

Kwan, A.S.K, Parker, M.E., Kwan, R.S.K, Fores, S., Proll, L., Wren, A. *Recent Advances in TRACS*, Proceedings from the 2004 Conference on Advanced Systems for Public Transport (CASPT).