AD-A281 990

NTATION PAGE

Form Approved
OMB No 0704-0188

RT DATE

3. REPORT TYPE AND DATES COVERED
~~THESIS~~/DISSERTATION

4. TITLE AND SUBTITLE
Constraints Cyclic Scheduling With Spacing

5. FUNDING NUMBERS

6. AUTHOR(S)
John Joseph Borsi, Major, USAF

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
AFIT Student Attending:
Georgia Institute of Technology

8. PERFORMING ORGANIZATION REPORT NUMBER
AFIT/CI/CIA-
94-015D

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
DEPARTMENT OF THE AIR FORCE
AFIT/CI
2950 P STREET
WRIGHT-PATTERSON AFB OH 45433-7765

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for Public Release IAW 190-1
Distribution Unlimited
MICHAEL M. BRICKER, SMSgt, USAF
Chief Administration

12b. DISTRIBUTION CODE

Accession For	
NTIS GRA&I	☑
DTIC TAB	☐
Unannounced	☐
Justification	
By	
Distribution/	
Availability Codes	
Dist	Avail and/or Special
A-1	

13. ABSTRACT (Maximum 200 words)

14. SUBJECT TERMS

15. NUMBER OF PAGES
155

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT

94-015D

Cyclic Scheduling With Spacing Constraints

John Joseph Borsi, Major, USAF

1994

155 Pages

Directed by Dr. Cynthia Barnhart and Dr. John Vande Vate

Degree Awarded: Doctor of Philosophy
Institution: Georgia Institute of Technology

We have studied a vehicle routing and scheduling problem in which a cyclic schedule of airlift missions is required to provide evenly spaced customer service. This thesis presents results in two areas relating to our solution of this problem.

First, we developed and tested a route-first/schedule-second heuristic. This approach is suitable for applications in which routing costs are the primary concern and the spacing requirement is not a hard constraint. The crucial element of this approach is the ability to schedule airlift missions so that the spacing requirement is satisfied. To accomplish this, we allow missions to be disrupted, either expediting or delaying mission stops compared to the standard time allowed for travel between consecutive mission stops. This scheduling problem is solved through a heuristic that requires the solution of a large number of assignment problems, which led to the second major area of our research.

We then developed efficient algorithms for special cases of the assignment problem related to our scheduling problem. We determined that cost matrices defined by a convex function of linear displacement between sorted locations on a line have the Strong Monge Property. Based on this finding, we developed assignment algorithms with worst case time complexities comparable to sorting a list.

CYCLIC SCHEDULING WITH SPACING CONSTRAINTS

A Thesis
Presented to
The Faculty of the Division of Graduate Studies

By

John Joseph Borsi

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy in Industrial and Systems Engineering

Georgia Institute of Technology
January, 1994

CYCLIC SCHEDULING WITH SPACING CONSTRAINTS

Approved:

Cynthia Barnhart

Cynthia Barnhart, Co-Advisor

John Vande Vate

John Vande Vate, Co-Advisor

John E. Jarvis

John J. Bartholdi

James E. Burns

Date Approved Jan 10 1994

ACKNOWLEDGMENTS

There are many people that contributed to this work and my education over the last eight years.

My advisors, Cindy Barnhart and John Vande Vate, were remarkable, showing amazing amounts of patience and endurance. If an education is measured primarily by changes in the way one thinks and perceives the world, then the vast majority of my education at Georgia Tech is a direct result of their kind guidance.

My friends, both at Georgia Tech and in the US Air Force, helped make this effort enjoyable.

Finally, I would like to thank my family. Their support kept me going, even though at times this process appeared endless. My mom lit many candles for me along the way, Jack helped remind me of the relative importance of this effort, and Jeri's support contributed more to this work than can be measured. While finishing this work is a milestone in my life, it cannot compare to the day I met Jeri or the day Jack arrived.

I dedicate this work to everyone that helped along the way, especially Jeri and Jack-man.

Table of Contents

LIST OF FIGURES

LIST OF TABLES

SUMMARY

We have studied a vehicle routing and scheduling problem in which a cyclic schedule of airlift missions is required to provide evenly spaced customer service. This thesis presents results in two areas relating to our solution of this problem.

First, we developed and tested a route-first/schedule-second heuristic. This approach is suitable for applications in which routing costs are the primary concern and the spacing requirement is not a hard constraint. The crucial element of this approach is the ability to schedule airlift missions so that the spacing requirement is satisfied. To accomplish this, we allow missions to be disrupted, either expediting or delaying mission stops compared to the standard time allowed for travel between consecutive mission stops. This scheduling problem is solved through a heuristic that requires the solution of a large number of assignment problems, which led to the second major area of our research.

We then developed efficient algorithms for special cases of the assignment problem related to our scheduling problem. We determined that cost matrices defined by a convex function of linear displacement between sorted locations on a line have the Strong Monge Property. Based on this finding, we developed assignment algorithms with worst case time

complexities comparable to sorting a list.

CHAPTER 1

PROBLEM DESCRIPTION

Air Mobility Command (AMC) of the United States Air Force is responsible for the air transportation of large quantities of material throughout the world for the United States Department of Defense and other government agencies. As recently reported in Defense News, "on average, Air Mobility Command ... flies more than 500 missions a day in more than 35 countries and the number is expected to increase in the future." A large part of this cargo is transported through the AMC Channel System, a transportation network of origin destination (O-D) pairs, known as channels, linked either directly or through transshipment points by regularly scheduled flights. As reported in Shepherd (1990), in Fiscal Year 1989 AMC served over 930 channels between 87 countries and moved nearly 320,000 tons of cargo at a cost of about 720 million dollars. Channel airlift is planned in a two phase approach, with a basic set of monthly airlift missions defined based on long term trends then altered on a monthly basis to meet actual customer requirements. The Force Structure Analysis Branch at Headquarters AMC develops the basic set of aircraft missions for the AMC Channel Network. They refer to

this problem as the Channel Network Design Problem.

1.1 The Channel Network Design Problem

An airlift **mission** is defined as the routing of an aircraft from its home base through a series of stops for onload/offload of cargo, refueling, and crew rest, returning to its home base. AMC develops a set of airlift missions and a schedule for a standard month based on projected demand for service between each O-D pair in the channel network. This monthly demand is forecast in terms of total tonnage and the frequency of service required. This forecast is consolidated from inputs by the various customers which use the channel network to transport material between O-D points.

In developing and updating the channel network, AMC seeks to satisfy the specified demand at minimum routing cost while providing evenly spaced repeated pickups for each channel. AMC planners wish to have the channel pickup times spaced evenly over a month to facilitate efficient planning and scheduling of activities and resources throughout the distribution network. In addition, this even spacing of service is seen as a way to increase customer satisfaction through timely movement of cargo after its arrival (through some other means of transportation) at an origin point. It can be shown (see Appendix A) that if cargo arrivals are completely random, then an even spacing of the pickups over

the planning horizon will minimize the average time a customer is forced to wait for its goods to be picked up at an origin point.

AMC's Channel Network Design Problem can be stated as: given forecast demands and frequency requirements for airlift service between channel origin and destination pairs, develop a set of airlift missions and a schedule which meets these demands. The objective is to minimize total routing cost with a secondary goal of providing evenly spaced repeated pickups for each channel. In designing and scheduling aircraft missions, AMC considers numerous constraints including: aircraft/airfield compatibility, the daily capacity of airfields, aircraft range and refueling requirements, airfield fuel supply, and crew duty day restrictions.

AMC currently solves this problem by first using a large linear programming model, known as the Strategic Transport Optimal Routing Model (STORM), to determine the number of times to fly different missions in order to satisfy demand requirements. These missions are then scheduled by a trial and error method, usually taking an analyst three to four days to develop. This mission plan is then evaluated through a large simulation model, checking for compliance with all required constraints. This is an iterative method, with results at any stage used to define additional routes for consideration in the LP model or to alter the mission

schedule. (Litko (1991)).

The focus of our research has been to develop a characterization of even spacing that meets AMC's needs and a methodology that efficiently incorporates the service spacing requirements into the mission design process. For this research we have considered an idealized version of the Channel Network Design Problem that we call the period pickup and delivery problem (PPDP).

1.2 The Period Pickup and Delivery Problem

The PPDP is a simplification of AMC's channel network design problem. The task in this problem is to develop a set of vehicle missions and schedule these missions over a given planning period of length H with the objective of minimizing routing costs. This set of missions and schedule are to be repeated indefinitely for successive periods.

Each customer requests several pickups of cargo for transportation between an O-D pair. The pickups and deliveries are accomplished by a capacitated fleet of vehicles and repeated pickups for the same customer, known as customer services, must be evenly spaced. A series of customer services over a period of time is **evenly spaced** if the times between successive services are equal to the length of the period divided by the number of services. For instance, if the period is 30 days long and a customer desires 5 pickups,

then an even spacing would have 6 days between successive service instances.

Routing problems typically require that missions start and end during the planning period. This will be treated as a special case of this problem and, in general, missions will not be constrained to be completed during the horizon. Since the mission schedule for this planning period is repeated indefinitely, each mission represents a set of equivalent missions which follow the same sequence of mission stops, with similar stops on consecutive equivalent missions separated by time intervals of length H. Therefore, if a mission is at some stop other than its home base at time H, i.e. the end of the planning horizon, then there is an equivalent mission at the same position in its route at the beginning of the horizon.

This dissertation details the results of our research on developing efficient solution methods for the PPDP. Before detailing our solution approach, we first present a survey of applicable results found in the literature.

1.3 Literature Survey

The period pickup and delivery problem is related to the period vehicle routing problem (PVRP) and the pickup and delivery problem (PDP).

1.3.1 The Period Vehicle Routing Problem The period vehicle routing problem is the problem of designing pickup or delivery routes for a given set of vehicles over a given H day period with constraints on the allowable spacing between visits to customers desiring repeated service (either a pickup or a delivery). The PVRP is also referred to as "The Assignment Routing Problem" in Russell and Igo (1979), the "Period Routing Problem" in Christofides and Beasley (1984), and the "Periodic Delivery Problem" in Bartholdi et al. (1987).

The surveyed papers present heuristics for the solution of the PVRP, but report no computational results for optimal approaches. The successful heuristics were based on decomposing the problem into an assignment of customer service to days within the horizon (usually one week), and then routing vehicles based on those assignments.

Beltrami and Bodin (1974) develop two approaches to a PVRP with customers specifying one of two possible frequencies for service. Their first approach was a route-first/schedule-second approach in which single day routes were developed with the restriction that no customer could be serviced on the same route more than once. Then the routes were assigned to days based on the frequency requirements. This approach was abandoned because they were unable to develop a method which guaranteed the scheduling requirements could be met with these generated routes. This is the only instance of the route-

first/schedule-second approach found in the literature. Their second approach was a schedule-first/route-second procedure. Customer services were randomly assigned to days of the week that satisfied spacing requirements. Routes were then developed based on this initial assignment.

Russell and Igo (1979) presented a more sophisticated approach to the assignment of service to days of the week. A customer's service spacing constraints were modeled by enforcing a minimum number of days between successive services. The assignment of a customer's services to specific days was based on measures of routing distance. Routes for each day were then developed using either an MTOUR approach presented in Russell (1977) or a modified Clarke and Wright (1964) savings algorithm.

Christofides and Beasley (1984), Tan and Beasley (1984) and Russell and Gribbin (1991) present heuristic algorithms based on choosing a delivery day combination for each customer from a set of acceptable combinations. Vehicle routes are then built based on this initial assignment, followed by local improvement procedures focusing on the effects of changing the assigned delivery day combinations for the customers.

Bartholdi, et al. (1987) presented two different heuristic approaches. Their first approach, the Marginal Cost Heuristic, assumed that each customer specifies a set of allowable delivery day combinations. The list of customers

was first sorted in increasing number of allowable service day combinations. Then the delivery days were assigned iteratively based on the marginal cost of each possible assignment. This marginal cost reflected the number of vehicles required in any time period of the planning cycle. In their second heuristic, the Busy Day Heuristic, they assumed that each customer only specifies the number of services desired over the horizon. Using a list of delivery days in a randomized order, a greedy heuristic assigned each customer service to days where a feasible routing was possible. In computational testing they found that their Marginal Cost Heuristic "reliably produced better solutions" than the heuristic of Christofides and Beasley (1984).

1.3.2 The Pickup and Delivery Problem The pickup and delivery problem (PDP) fits into the broad class of routing and scheduling problems characterized by task precedence and time window constraints. In the PDP customers specify amounts of material to be transported between origin and destination points subject to possible restrictions on allowable times for pickups and deliveries.

The objective functions for problems reported in the literature consist of combinations of two measures. The first measure is the cost of the routes needed to service all customers. The second measure reflects customer satisfaction

and is commonly the time a good is in transit in excess of the minimum time it would take for direct transit between its origin and destination.

Optimal solution techniques reported in the literature can only solve problems of very limited size. The largest problems solved optimally have between 40 and 50 O-D pairs (Desrosiers et al. (1986), Dumas (1985), and Dumas and Desrosiers (1986)).

Several Heuristic procedures have been reported in the literature (including those in: Stein (1978); Cullen, Jarvis and Ratliff (1981); Psaraftis (1983b); Bodin, Golden, Assad and Ball (1983), Desrosiers, Dumas and Soumis (1986b); Solomon and Desrosiers (1988); Jaw, Odoni, Psaraftis and Wilson (1986)). There has been no reported comparison of the quality of solutions among the various methods, possibly due to the lack of a standard set of test problems.

The most successful approach for large problems was reported in Jaw, et al. (1986) which presents an insertion heuristic for a multi-vehicle PDP with time windows on allowable service. This heuristic was used to generate a solution to the largest problem reported in the literature, with 2600 customers and 20 vehicles. Their algorithm uses a parallel insertion procedure to assign customers to vehicles and determine the best schedule for each vehicle. For each insertion the algorithm uses a cost function based on measures

of routing costs and customer inconvenience.

Insertion procedures of this type can handle large numbers of customers and a wide range of vehicle constraints and are compatible with the solution approach we now present in Chapter 2.

CHAPTER 2

SOLUTION APPROACH

2.1 Overall Approach

In this section we describe our route-first/schedule-second solution method for the PPDP. We first discuss reasons why a route-first/schedule-second approach is appropriate for application to the Channel Network Design Problem despite the feasibility problems noted in Beltrami and Bodin (1974).

A route-first/schedule-second approach emphasizes the use of low cost vehicle routes. This is compatible with the objective of the Channel Network Design Problem, which is to develop low cost missions with evenly spaced customer service a secondary goal. Approaches that consider the scheduling constraints as an integral part of the initial generation of vehicle routes inherently give precedence to the scheduling constraints over the routing objective. Also, without the spacing constraints the PPDP is a pickup and delivery problem. A route-first approach would be able to use the most powerful solution methods available for the generation of aircraft missions.

In addition, approaches that consider the scheduling constraints as an integral part of the initial generation of

vehicle routes do not explicitly consider the cost of enforcing spacing constraints. In situations in which good customer spacing is merely "nice to have", the routing penalty required to achieve a given level of service is an important piece of information. An approach that alters a set of vehicle missions to accommodate customer service spacing would provide estimates of the additional routing costs incurred in enforcing service constraints.

Therefore, a procedure that allows for the alteration of existing missions to accommodate spacing requirements is appropriate for application to the Channel Network Design Problem.

Based on these observations we have developed the following solution approach:

PPDP SOLUTION PROCEDURE

STEP 1: Initial Mission Generation Step: Develop a set of vehicle missions, ignoring spacing requirements, but enforcing all other constraints.

STEP 2: Spacing Step: Schedule these missions, enforcing spacing requirements and minimizing the routing penalties necessary to extend or compress the current missions to meet these spacing constraints.

STEP 3: Mission Alteration Step: Based on this schedule and estimate of the routing costs needed to enforce spacing requirements, alter the existing set of vehicle missions.

STEP 4: Iteration Step: Repeat STEP 2 and STEP 3 to find a good compromise between routing and spacing requirements.

The approach outlined above does not restrict the methodology used to generate the initial set of missions in STEP 1. Thus, this approach is compatible with AMC's STORM model. Otherwise, insertion heuristics, with their established success on similar problems (Jaw et al. (1986)), are appropriate.

In the Spacing Step, the missions are altered by increasing or decreasing the transit time between consecutive locations in order to achieve the desired customer service spacing. The goal of this step is to minimize the cost of altering the given set of missions. These costs are discussed in more detail in Section 2.2.2.

The literature contains many reports on local improvement interchange procedures and interactive route generation procedures, e.g. Bodin et al. (1983). These methods could be used to alter missions in STEP 3 based on the information generated in the Spacing Step. We have tested a simple heuristic for the alteration of missions in STEP 3 and report on this in Section 2.4.2.

The problem in STEP 2 of scheduling a given set of missions to satisfy service spacing requirements is referred to as the **Mission Disruption Problem** (MDP). The remainder of this dissertation presents the results of our investigation into efficient approaches for this problem.

2.2 The Mission Disruption Problem

2.2.1 Problem Description In the first step of the PPDP Solution Procedure, we generate a set of vehicle missions that collectively serve each customer the desired number of times. In the next step, we wish to schedule customer services evenly over the planning horizon.

In general, standard aircraft speeds and ground times (which include the time to load and unload cargo, service the aircraft, and allow for crew rest) are used when developing airlift schedules. Given a mission start time, each stop of a mission can be scheduled based on these standard times which are detailed in Air Force Pamphlet (AFP) 76-2. For example, consider a round trip C-141 mission which flies between McGuire Air Force Base (AFB), New Jersey and Torrejon, Spain with a stop at Lajes Field in the Azores for fuel. Under standard operating conditions, this C-141 mission would take 5.7 hours to fly to Lajes, spend 2.25 hours on the ground there, then take 3.1 hours to continue on to Torrejon. Before the return trip could start, another 2.25 hours of ground time plus the time required for crew rest would be required in Spain.

A schedule which deviates from these standard parameters may be feasible but can greatly increase the cost of a vehicle mission. In the previous example, it would be possible to use inflight refueling to avoid the refueling stop in the Azores.

Also, additional crew members could replace the original crew during the stop in Torrejon, and the entire mission could be accomplished in a single day. Both of these options require higher operating costs. **Mission disruption** is a measure of the deviation of a schedule from standard mission times. In our application, we allow mission disruptions in order to evenly space customer services. Formally, if a mission stop would normally be scheduled to occur at time t' and instead the mission stop is scheduled for time t, then the mission disruption associated with this stop is $t - t'$.

Therefore, the **mission disruption problem** may be expressed as follows:

Mission Disruption Problem: given a set of missions, minimize the cost of the mission disruption needed to provide evenly spaced service for each customer.

2.2.2 Mission Disruption Costs We consider two types of mission disruptions which we call positive and negative disruption. **Positive disruption** occurs when the scheduled time between two points visited consecutively by a mission exceeds the standard travel time between these points. In this case, a delay has been inserted into the schedule. This disruption is called "positive" disruption because the amount of disruption is non-negative. Theoretically, there is no limit on the time an aircraft can be on a mission although most missions last a week or less. The costs associated with

this parameter are the costs of supporting an aircraft and its crew away from the home base for longer than the standard amount of time required to accomplish its mission.

Negative disruption occurs when the scheduled time between two consecutively visited points is less than the standard allowed time. In this case, the mission must be expedited and service is provided sooner than it would under standard planning. Note that in this case the amount of disruption is negative. To accommodate negative disruptions the standard aircraft speed can be increased or aircraft ground processing time can be reduced. These options would result in increased fuel costs, extra aircraft maintenance, and the costs of supporting in-flight refueling. In addition, additional crew members can either augment or replace the aircraft crew, thus reducing ground time for crew rest. For example, aircrew restrictions dictate that the maximum time a crew can operate an aircraft is 16 hours. However, if this crew is augmented by adding backups for key positions, this maximum time can be extended to 24 hours. Finally, if the entire crew is replaced at the end of their duty day the aircraft can be operated almost continuously except for ground time for refueling and onload/offload of cargo. Pursuing these options to expedite aircraft operations requires that additional crew members be maintained away from their home bases for extended periods of time, which leads to increases

in personnel costs. As a final resort, a customer service could be accomplished by another vehicle. In this case the mission disruption costs would reflect the cost of altering another mission or detailing another vehicle to perform that customer service.

Let $MDC:R \rightarrow R^+$ be the function which defines the cost for a given level of mission disruption. We assume that MDC is normalized (i.e. $MDC(0) = 0$) with positive domain values representing positive disruption and negative domain values representing negative disruption. The cost of disrupting a mission which would normally arrive at a pickup point at time t to perform this service at another time s, denoted $c_t(s)$ and referred to as the service cost, depends on whether the planning period is a single independent period of time or is repeated indefinitely. Consider the former case, referred to as the "linear" case, in which the scheduling period is not repeated over time and all missions must start and end during this period. If $t = s$, then no disruption is required and the cost of mission disruption is zero. Note that, since MDC is normalized, $MDC(s-t) = MDC(0) = 0$. If $t < s$, then that mission stop must be delayed and the service cost would be $MDC(s-t)$. Otherwise, if $s < t$, then that mission must be expedited at cost $MDC(s-t)$. Therefore, in the linear case the service cost $c_t(s) = MDC(s-t)$. For example, consider the linear case with $MDC(x) = |x|$. The service cost function for

a mission stop that, using standard planning times, would normally occur at time t is illustrated in Figure 2.1 below

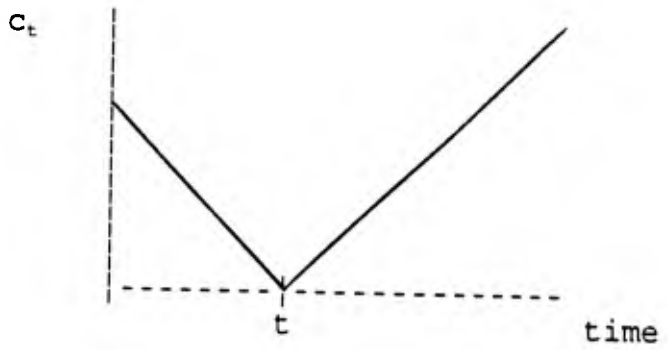

Figure 2.1. Service Cost for the Linear Case with MDC(x) = |x|.

On the other hand, consider a planning period of length H that repeats over time with missions that are not constrained to begin and finish during the horizon. We refer to this as the "circular" case. In this case, a mission that would normally serve a customer at time t represents a set of equivalent missions which arrive at that stop at times $t \pm (k*H)$ for $k = 0,1,\ldots,\infty$, as described in Chapter 1. As such, this mission or an equivalent mission could be either expedited or delayed to meet a scheduled service time s, regardless of the relative positions of t and s. For instance, if a mission would normally arrive for a customer service at day 5 of a 7 day period and the customer desired service on day 3, then that mission stop could either be expedited 2 days or an equivalent mission in the just previous

period could be delayed 5 days. In this case the service cost would be the minimum of the mission disruption costs to expedite and delay that service. Therefore, in the circular case, the service cost function $c_t: [0,H) \rightarrow R^+$ for a mission that would normally arrive for service at time t is defined as follows:

$$(2.1) \qquad c_t(s) = \begin{cases} \min\{MDC(s-t),\ MDC(H-t+s)\}, & \text{if } s \leq t \\ \min\{MDC(s-t-H),\ MDC(s-t)\}, & \text{if } t < s. \end{cases}$$

Figure 2.2 below depicts cost function (2.1) for a mission stop that would normally occur at time t in a planning period of length H with $MDC(x) = |x|$.

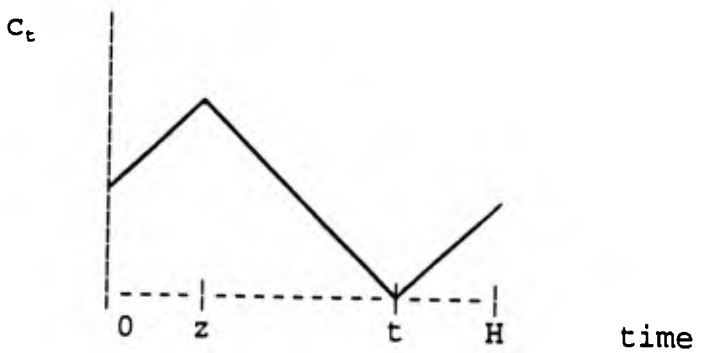

Figure 2.2. Service Cost for the Circular Case with $MDC(x) = |x|$.

Note that this cost function has a locally concave breakpoint at time z where the cost of expediting a mission to that time equals the cost of delaying an equivalent mission to the same time. This observation is examined in more detail in Chapter 4.

The form of the disruption cost function greatly influences the procedures used to develop evenly spaced schedules. In this chapter we present an approach that requires that the normalized disruption cost function $MDC: R \rightarrow R^+$ be piecewise linear. We do not consider this a restrictive assumption since a continuous function of one variable can be approximated to any desired degree of accuracy by a piecewise linear function.

2.2.3 Problem Formulation We now present the mathematical programming formulation of the circular case of the mission disruption problem. In this formulation we assume that a customer may be served at most once by each mission. This assumption could be relaxed by using an additional subscript to differentiate between different services of a given customer by the same mission.

Let i and m denote mission numbers and j and k denote customer numbers. The planning period's length is denoted by H. Customer j requires service $FREQ_j$ times over this period and M_j denotes the set of missions that perform these services. We let $PS_j = H/FREQ_j$ denote the perfect spacing interval for customer j.

When describing a mission i, A_{ij} denotes the customer service immediately following customer j on mission i for $i \in M_j$. A_{ij} is null if j is the last customer served on mission i.

The standard time used for planning purposes between the service of customer j by mission i and the start of the next customer service by mission i is denoted $StdTIME_{ij}$. This parameter, as discussed earlier, includes time for travel between vehicle stops, aircraft service, cargo handling and crew rest.

The decision variables for this formulation are denoted CS_{ij} for the time that mission i begins the service of customer j. Since we allow a mission to start in one period and finish in another, the true customer service time for a customer j by mission i is CS_{ij} (mod H). In order to avoid modular arithmetic in our formulation, we do not place an upper bound on the CS_{ij} values but require that all customer services of a given customer occur within H time units of the first scheduled service of that customer. This requirement is enforced by the constraints for even spacing.

To ensure that even spacing is enforced, we use variable IST_j to indicate the initial service time of that customer. All acceptable service times for customer j are then $IST_j + (k*PS_j)$ for $k = 0,\ldots,FREQ_j-1$.

Each value of IST_j for a customer j defines a set of possible service times. Since each service must be performed by exactly one mission, we use the variable T_{ij} to denote the order of missions which serve customer j. Therefore, T_{ij} is an integer on the range $[0,FREQ_j-1]$ and, given the values IST_j

and PS_j, mission i serves customer j at time $IST_j + (T_{ij}*PS_j)$.

Given a value for each CS_{ij}, MD_{ij} denotes the mission disruption immediately after the service of customer j on mission i. This parameter can be positive or negative.

Finally, the cost function for different levels of positive or negative mission disruption will be denoted as $COST(MD_{ij})$. This cost function is based on equation (2.1), modified to allow for arguments not on the range [0,H).

Using this notation, the mathematical formulation of the Mission Disruption Problem is:

$$\text{Minimize} \quad \sum_j \sum_{i \in M_j} COST(MD_{ij}) \tag{1}$$

subject to:

$$IST_j + T_{ij}*PS_j = CS_{ij} \qquad \forall j;\ \forall i \in M_j \tag{2}$$

$$T_{ij} - T_{mj} \geq 1 \quad \text{or} \quad T_{mj} - T_{ij} \geq 1 \qquad \forall j;\ \forall i,m \in M_j \tag{3}$$

$$CS_{ik} - CS_{ij} = StdTIME_{jk} + MD_{ij} \qquad \forall j;\ \forall i \in M_j;\ k = A_{ij} \tag{4}$$

$$IST_j \geq 0 \qquad \forall j \tag{5}$$

$$MD_{ij} \text{ urs} \qquad \forall j;\ \forall i \in M_j \tag{6}$$

$$0 \leq T_{ij} \leq FREQ_j - 1 \text{ and integer} \qquad \forall j;\ \forall i \in M_j \tag{7}$$

The service spacing constraints (2) define the times for evenly spaced service.

Constraints (3) and (7) define the service indexing variables: T_{ij}. These variables form an integer sequence from 0 to $FREQ_j - 1$ to ensure the spacing in constraints (2) is even.

The mission disruption constraints (4) allow deviations from the standard time between consecutive services by the given missions. The value of each mission disruption variable is based on the scheduled times for customer service, the input customer sequence for each mission, and the standard time between consecutive stops.

Given this formulation it can be shown (see Appendix B) that the mission disruption problem is a generalization of the directed optimal linear arrangement problem. This problem is NP-complete (Garey and Johnson (1979)); therefore, it is unlikely that an efficient method for generating optimal MDP solutions can be developed. Since our application leads to a large instance of the mission disruption problem, we have developed a heuristic solution approach which we now present in Section 2.3.

2.3 **MDP Solution Approach**

Our approach for the MDP is to solve a series of single customer problems. Considering the MDP from the standpoint of a single customer j, if the schedule for all other customers is fixed, then there are two questions which must be answered: "What sequence of missions will serve customer j over the planning period?", and "What are the times for evenly spaced service for customer j?" Note that, given a value for the variable IST_j, each service time is completely determined by

this value and the parameter PS_j. We label these times T_k for $k = 1,\ldots,FREQ_j$ with $T_k = IST_j+(k-1)*PS_j$. If all other customer service times are fixed, then standard mission planning factors specify the normal time each mission i in M_j arrive to serve customer j. Therefore, the disruption cost incurred to serve customer j at time T_k for $k = 1,\ldots,FREQ_j$ can be specified for each mission in M_j. We denote these costs by $T\text{-}COST_{ik}$, for the disruption cost incurred by using mission i to serve customer j at time T_k. Then, if we define variable x_{ik} as one if mission i serves customer j at time T_k and zero otherwise, the single customer problem is:

$$\text{Minimize} \quad \sum_{i\in M_j,\, k=1,\ldots,FREQ_j} T\text{-}COST_{ik}\, x_{ik} \qquad (8)$$

subject to:

$$\sum_{k=1,\ldots,FREQ_j} x_{ik} = 1 \qquad \forall i \in M_j \qquad (9)$$

$$\sum_{i\in M_j} x_{ik} = 1 \qquad k = 1,\ldots,FREQ_j \qquad (10)$$

$$x_{ik} \in \{0,1\} \qquad \forall i\in M_j,\ k=1,\ldots,FREQ_j \qquad (11)$$

In this formulation, constraint (9) specifies that each mission must be assigned to a service time, and constraint (10) ensures that a single mission serves customer j at time T_k. Thus, when IST_j is prespecified and all other customer service times are held constant, the single customer MDP can be formulated and solved as an assignment problem.

We now consider the set of possible values for IST_j. When customer service times are fixed and redefined using modular arithmetic, the first service of customer j during the planning period must be on the range $[0, PS_j)$. Therefore, the mission disruption problem for customer j becomes the problem of finding the value of IST_j on the range $[0, PS_j)$ that gives the minimum cost for the assignment problem (8)-(11).

Let $A_j: [0, PS_j) \rightarrow R^+$ denote the value of the assignment problem (8)-(11) for different values of IST_j. Our task is to find the minimum value of A_j for IST_j values on the range $[0, PS_j)$. Note that A_j is not differentiable. In addition, this function is not necessarily unimodal and in general, finding the minimum value of such a function can be quite difficult. In order to examine the properties of A_j we first examine the cost of assigning a mission to time T_k for $k \in \{1, \ldots, FREQ_j\}$ as a function of IST_j.

Assume, using standard planning factors and modular arithmetic, that mission i would normally arrive to serve customer j at time t. Consider the cost of using this mission to serve customer j at time $T_k = IST_j + (k-1)*PS_j$ and let $TC_{i,k}: [0, PS_j) \rightarrow R^+$ be the function of IST_j that describes the cost of assigning the service of customer j by mission i to T_k. Then, using equation (2.1), we have

$$TC_{i,k}(IST_j) = c_t(IST_j + (k-1)PS_j). \tag{2.2}$$

Therefore, $TC_{i,k}$ has the form of the service cost function $c_t: [0,H) \rightarrow R^+$. We have assumed that the mission disruption cost function is piecewise linear, implying that the service cost function is piecewise linear. Therefore, $TC_{i,k}$ is also piecewise linear. We say that a value of IST_j **corresponds to a breakpoint of** c_t if the function c_t has a breakpoint at domain value $IST_j + (k-1)*PS_j$ for some $k \in \{1,\ldots,FREQ_j\}$.

A feasible assignment for the problem (8)-(11) is the sum of the costs of each individual assignment of a mission to a service time. Let X denote the set of x_{ik} values that equal one for a given assignment. The cost of X for a given value of IST_j, denoted $C_x(IST_j)$, is as follows:

$$(2.3) \qquad C_x(IST_j) = \sum_{i \in M_j, x_{ik} \in X} TC_{ik}(IST_j + (k-1)PS_j).$$

Therefore, the cost of a given assignment is a function of IST_j and is piecewise linear with breakpoints at values of IST_j that correspond to breakpoints of the c_t functions for each mission serving customer j.

It is well known that the minimum of a piecewise linear function defined over a range will occur at an endpoint of the range, at a point of discontinuity, or at a point where the function's slope changes. Therefore, for a given assignment, the minimum value of (2.3) will occur for an IST_j value which is either equal to 0 or corresponds to a breakpoint of the service cost function c_t for some mission serving customer j.

Therefore, to find the minimum value of C_x we can specify all possible values of IST_j which meet these conditions and then evaluate the cost of X at each of these points. Therefore, we can find the minimum cost assignment and IST_j value by solving an assignment problem for each value of IST_j that corresponds to a breakpoint.

Note that the number of missions which serve customer j is $FREQ_j$ and let $|B|$ designate the number of breakpoints of a customer service cost function. When the services are evenly spaced, no more than one value of IST_j corresponds to a cost function breakpoint. Therefore, the minimum cost IST_j for customer j can be found by solving $O(FREQ_j*|B|)$ assignment problems.

To solve the MDP we have developed a local optimization procedure that iteratively solves a series of single customer MDPs. While addressing a single customer's sequencing and scheduling problem, the service times for all other customers are held constant. Noting that each single customer solution can change the mission schedule, our procedure steps through a circular list of the customers until the iterative decrease in disruption meets a stopping criteria. This iterative approach allows us to reconsider single customer solutions after subsequent schedule changes. We discuss stopping criteria in Section 2.4. We call our procedure the Single Customer Heuristic.

SINGLE CUSTOMER HEURISTIC

STEP 1: Initialization: Order the customers in a circular list. Generate an initial schedule for each mission.

STEP 2: Single Customer Problems: For each customer j in the customer list:

STEP 2a: List all IST_j values which correspond to mission disruption cost breakpoints.

STEP 2b: Solve the associated assignment problem for each IST_j value listed.

STEP 2c: Pick the minimum cost assignment solution from all solutions in STEP 2b, and reschedule the mission stops based on this solution.

STEP 3: Iterate: Repeat Step 2 until stopping conditions are met.

We now discuss our implementation and testing of this procedure.

2.4 Implementation and Computational Testing

2.4.1 Implementation Issues: We have examined methods to generate an initial schedule in STEP 1, sequence customers in STEP 2, and terminate the procedure in STEP 3.

We tested four methods for the generation of an initial schedule by designating a start time for each mission with the schedule of subsequent mission stops determined using the given standard times between stops. In the first method, we arbitrarily scheduled the first customer service of each mission at time zero. Alternately, we attempted to generate initial start times for each mission based on the linear

programming relaxation of the Mission Disruption formulation. As a third approach we implemented an algorithm, known as MISCHED, used by analysts at AMC for mission planning. The code for this program is contained in Appendix C. In the final method tested, random start times were generated for each mission. Using this approach, we ran the Single Customer Heuristic on several sets of randomly generated mission start times and output the best solution. Note that the initial schedules developed through these methods were not constrained to give even spacing of customer services. The results from testing these four methods are discussed in Section 2.4.2.

The sequence of customers for STEP 2 was generated by ordering the customers by number of services required over the horizon from high to low. The idea behind this approach is that those customers with a requirement for a large number of services are highly constrained and would affect the most missions. Likewise, the customers that require fewer services affect fewer missions and therefore, have a smaller impact on the total mission disruption cost than those customers which require many services.

We examined two possible stopping rules for STEP 3: stopping after a fixed number of iterations, and stopping when the change in total mission disruption from one iteration to the next fell below a given level.

2.4.2 Results of Computational Testing: We implemented the Single Customer Heuristic in Pascal on a Zenith 248 personal computer. We then tested this approach on problems ranging in size from a few customers and missions to a large problem of 247 customer services of 30 customers by 50 missions over a 30 day planning horizon. Tables II.1 and II.2 summarize the characteristics of the data files used for different tests of the algorithm.

Table 2.1 Data File Parameters

FILE	# CUSTOMERS	# MISSIONS	SERVICE FREQUENCY RANGE	TOTAL # SERVICES
MTEST.1	4	4	[4,4]	16
MTEST.1c	4	7	[4,6]	19
MTEST.2	4	4	[4,4]	16
MTEST.3	4	4	[3,4]	15
MTEST.4	4	4	[4,4]	16
MTEST.5	9	10	[2,6]	37
MTEST.5B	9	10	[2,6]	37
MTEST.5C	9	10	[2,6]	37
BTEST.1A	30	30	[2,10]	128
BTEST.1B	30	50	[3,16]	247
AMC.1	15	22	[2,17]	97
AMC.2	23	17	[4,9]	150
AMC.3	12	41	[2,26]	136

Table 2.2 Data File Characteristics

FILE	Characteristics
MTEST.1	Identical missions, customers located at corners of a square, optimal solution known.
MTEST.1c, MTEST.2, MTEST.3, MTEST.4	Small problems with slightly different sets of data: missions with different sequences and customers with different frequencies of service. Same customer locations as for MTEST.1
MTEST.5	Mission sequences randomly generated. Customer locations at intersections of a 3x3 grid. Customer service frequencies vary from 2 to 6.
MTEST.5b, MTEST.5c	Same customer locations, frequencies and number of missions as MTEST.5. Mission sequencing changed based on results of previous tests.
BTEST.1a	Customer locations randomly generated. Mission sequences were generated so that the optimal solution to the MDP was known.
BTEST.1b	Randomly generated missions and customer locations. Maximum number of customers per mission was 10.
AMC.1, AMC.2, AMC.3	Representative sets of AMC Channel Cargo Missions.

Computational testing was conducted to assess the quality of the final solutions, the convergence properties of this approach, the results of different initial mission start times, and the feasibility of using the results of this algorithm to generate new missions that could be scheduled with less mission disruption. In the following discussion, we categorize the size of the problem as **small** for data files with fewer than 20 customer service events. A **large** problem

has at least 100 customer service events, and any other problem will be termed a **medium** size problem.

To test solution quality, we generated data sets for which we either knew an optimal solution or an upper bound on the minimum disruption. This was accomplished for large and small problems. The algorithm successfully generated an optimal solution in all cases where we knew the optimal solution value. In the problems where we only knew an upper bound on an optimal solution value the algorithm generated solutions that were at least as good as these bounds. Several data sets were modified to test the sensitivity of the final results to minor changes in mission sequencing, number of missions, or number of customer services required. In all of these cases the output solutions did not vary widely in value for similar inputs. Therefore, while convergence to an optimal solution cannot be guaranteed, in our testing the approach generated reasonable solutions.

To examine the convergence properties of this approach, we calculated the level of mission disruption after each cycle through the customer list. In the simpler cases with known optimal solutions, the optimal solution was found in the first pass through the customer list. In the bigger test cases, convergence to a final solution usually occurred within 5 to 10 passes through the customer list in STEP 2 of the algorithm. The solution improvement between iterations of

STEP 2 did not occur at a monotonically decreasing rate. In several cases, a small change in the mission disruption was followed by a larger decrease in mission disruption. Therefore, a stopping rule that depends solely on percentage change between iterations may terminate prematurely. Overall, from the initial perfect spacing feasible solution to the final solution, the iterative approach generated decreases in mission disruption between 15% and 87%. In our computational testing, the solutions tended to converge to a local optima within a few iterations (usually less than 10), taking about 10 minutes for the large problems on a Zenith 248 PC. Therefore, we recommend a combined approach for terminating the algorithm, stopping either when the iterative change in the objective function falls below a given level for several consecutive iterations or after a maximum number of iterations have been performed.

We found that the choice of initial mission start times did affect the quality of the final solutions. Attempts to use a linear programming relaxation of the mission disruption formulation to generate initial mission start times were unsuccessful. A large number of fractional values for the integer variables which control service spacing resulted in solutions with no useful mission scheduling information. We implemented the other three approaches for generating initial mission start times. The first version arbitrarily

initialized all mission start times to zero, the second method used MISCHED, and the third randomly generated initial mission start times. In general, this final approach yielded better solutions than the other approaches. This testing did show that initial solutions with the least amount of mission disruption often did not lead to the best final solutions. In the absence of any firm results detailing how to generate an initial schedule that leads to a good final solution, we recommend using both MISCHED and random generation of mission start times to initialize the schedule.

Our testing also established the feasibility of altering the mission sequences to generate a lower cost solution to the mission disruption problem. We used an insertion/interchange approach to delete customers from missions where disruption costs reflected a need to expedite the mission, and reinsert these services into other missions where a disruption cost reflected a need to delay a mission. In this manner, we generated a new set of missions which satisfied the original customer service requirements but which did not increase the number of missions flown. For our testing we altered the missions based solely on the mission disruption costs (and did not consider the routing costs) and were able to reduce the total mission disruption penalty by as much as 71% for a medium sized problem. For the solution of the PPDP this procedure can be implemented as an insertion/interchange

approach that considers both routing and mission disruption costs.

2.5 Conclusions

In this chapter, we have presented our route-first/schedule-second procedure for the PPDP. From this approach we formulated the mission disruption problem and discussed a heuristic solution approach and related computational results for this problem. Overall, the computational testing of the Single Customer Heuristic showed that this method generates reasonable solutions. The choice of initial mission start times did make a difference on the final solution, but in the tested problems, good initial solutions did not guarantee having a good final solution. Therefore, the recommended approach is to generate an initial mission schedule using AMC's MISCHED program, and then, as computational resources allow, randomly generate initial schedules.

The most computationally expensive part of the Single Customer Heuristic is the solution of large numbers of assignment problems. Using an approach that randomly generates initial schedules will require a large amount of computer time for large problems. Increasing the computational efficiency of the solution of assignment problems would allow more initial schedules to be generated

and tested using the same amount of computer resources. As part of our research we have developed an efficient assignment problem solution methodology that takes advantage of cost function structure. In Chapters 3 and 4 we present efficient assignment problem algorithms for cases in which the assignment costs are generated by convex functions.

CHAPTER 3

THE ASSIGNMENT PROBLEM ON THE LINE

3.1 Introduction

In Chapter 2 we presented the Single Customer Heuristic to minimize the mission disruption required to schedule evenly spaced customer service. This procedure improves a schedule by reducing the disruptions associated with a single customer, holding the schedule for other customers fixed. For a given customer, the procedure solves a series of assignment problems to minimize the costs of expediting or delaying missions to achieve the desired service spacing. The disruption problem for a single customer may be solved several times to take advantage of schedule changes as other customers are rescheduled. Thus, the time required to find a good schedule depends heavily on how quickly we can solve each assignment problem. In this chapter we explore the structure of a practical special case of our problem and discuss efficient assignment procedures.

Currently, the most efficient general purpose assignment procedure is the auction algorithm (Ahuja et al. (1988)). This algorithm finds a schedule for a customer requiring n services in $O(n^{2.5}*\log(n))$ time. We do not know of a more

efficient solution method for the general case in which each mission may have a distinct mission disruption cost function for each customer served. During advanced mission planning it may be impractical or impossible to estimate mission disruption costs at this level of detail. A practical approach would be to develop generic mission disruption cost functions, estimating the disruption costs for broad classes of mission categories based on aircraft utilized, expected crew characteristics, typical cargo load and mission length. Since customers are usually served by very similar missions, in many cases a single disruption cost function could be applied to every mission serving that customer. When this happens it is possible to develop more efficient assignment procedures than the auction algorithm.

When the missions serving a customer are essentially the same, the primary concern becomes the difference between the scheduled mission arrivals and desired customer service times. Common sense suggests that missions should provide service in the order of their arrival. For example, consider a customer that desires service on Monday and Friday with similar missions arriving on Tuesday and Wednesday for these services. If the costs of mission disruption are non-decreasing for increasing amounts of disruption, then it would be unreasonable to expedite Wednesday's mission to Monday and delay the Tuesday mission long enough to provide the Friday

service. This observation leads to an assignment of missions to customer services formalized via non-crossing assignments, which we define and discuss in Section 3.2.

Not all mission disruption cost functions admit an optimal assignment that is non-crossing. In Section 3.2 we discuss conditions under which there is a non-crossing optimal assignment. In particular, we show in Section 3.3 that a non-crossing assignment is optimal when costs can be approximated by a convex function of the amount of mission disruption needed to reschedule a mission to service a customer. Based on this fundamental result, we develop and present an approach that can solve assignment problems in O(n*log n) computational time when the cost function is a piecewise linear convex function of the disruption incurred in scheduling a mission to service a customer.

We use the following notation and terminology in our discussion of the assignment problem. Consider two disjoint finite sets S and T of points in the interval [0,H). We refer to $S = \{s_i: i = 1,2,\ldots,m\}$ as the sources and $T = \{t_j: j = 1,2,\ldots,n\}$ as the destinations. Assume that $m \geq n$, and that these sets are indexed so that

$$s_1 \leq s_2 \leq \ldots \leq s_m \text{ and } t_1 \leq t_2 \leq \ldots \leq t_n.$$

Note that we allow sources (destinations) to have the same location as other sources (destinations). To avoid confusion,

we use the standard relational notation "$<$, $\leq$, $=$, $\geq$, $>$" to compare points in the interval $[0,H)$ and we use "$\prec$, $\preceq$, $\equiv$, $\succ$, $\succeq$" to compare the indexed order of sources or destinations.

An assignment of sources to destinations is a one-to-one mapping μ from $S \bigcup T$ to itself of order two (that is $\mu(\mu(x)) \equiv x$) such that

- For each $s \in S$, either $\mu(s) \in T$ or $\mu(s) \equiv s$ in which case we say that s is unmatched.
- For each $t \in T$, either $\mu(t) \in S$ or $\mu(t) \equiv t$ in which case we say that t is unmatched.

Since μ is of order two, if a source (destination) s is assigned to a destination (source) t, i.e. $\mu(s) \equiv t$, then $\mu(t) \equiv s$. In this case, we say s and t are matched and we refer to s and t as mates. For a subset X of $S \bigcup T$, we define $\mu(X) = \{\mu(x) : x \in X\}$.

In general, we are concerned with problems that require an assignment of T into S. Unless otherwise specified, an assignment matches each destination in T with a distinct source in S.

The assignment problems we consider require a minimum cost assignment. In this chapter we consider problems with the destinations and sources located on a line. On a line, the cost of matching a source s with a destination t is a function f of the linear displacement between them, i.e., the cost of matching s and t, denoted $c_L(s,t)$, equals $f(s-t)$. The cost of an assignment μ, denoted $\text{Cost}(\mu)$, is computed via an

mxn real-valued cost matrix $C_L(S,T,f) = (c_L(s,t): s \in S$, and $t \in T)$ and equals $\sum_{t \in T} c_L(\mu(t),t)$.

3.2 **Non-crossing Assignments**

In Section 3.1 we discussed the common sense observation that missions should provide service in the order of their arrival. This observation leads us to consider non-crossing assignments in which the source and destination locations correspond to mission arrival and customer service times.

Formally, two pairs (s_i, t_j) and (s_k, t_m) in S x T are said to <u>cross</u> if

$$i < k \text{ and } j > m.$$

An assignment is called (linearly) <u>non-crossing</u> if no two of its matched pairs cross. The name "non-crossing" is motivated by the following geometric characterization for a case with distinct source and destination locations (first developed in Gilmore and Gomory (1964)). Place the sources on one line segment and the destinations on a parallel line segment. An assignment μ is non-crossing exactly when the line segments connecting the sources to their mates do not cross.

While studying the construction of military fortifications, Gaspard Monge (1746-1818) observed that the north-west corner rule provides an optimal solution to the transportation problem when the associated n x n cost matrix

$C = (c_{ij})$ has the following property, called the Monge Property (Derigs et al (1986)):

$$c_{ii} + c_{jk} \leq c_{ik} + c_{ji}, \text{for } i=1,2,\ldots,n \text{ and all } j,k > i.$$

This result was presented in Hoffman (1961) and used to explain why certain warehousing problems were amenable to extremely efficient solution procedures. Barnes and Hoffman (1982 and 1985) have used these results to develop efficient procedures for a partitioning problem associated with logic chip design.

Monge's result is applicable to the assignment problem, which can be modeled as a transportation problem in which each source has one unit of supply and each destination has one unit of demand. Applied to this model, the north-west corner rule assigns the first source to the first destination, the second source to the second destination, etc. So, if the sources and destinations are ordered by their positions, the north-west corner rule generates a non-crossing assignment and, if the number of sources and destinations are equal and the cost matrix has the Monge Property, this assignment is optimal.

Therefore, an instance of the assignment problem with an equal number of sources and destinations is easily solved if the cost matrix has the Monge Property (possibly after rearranging rows and columns). However, the problem of

recognizing whether the rows and columns of a matrix can be rearranged to give a matrix with the Monge Property remains open. Deineko and Filonenko (1979) have shown how to determine in $O(n^2)$ time whether the rows and columns of a matrix can be rearranged to give a matrix with the following, more restrictive version of the Monge Property. A square matrix $C = (c_{ij})$ has the Strong Monge Property if the following relation holds:

$$c_{ik} + c_{jm} \leq c_{im} + c_{jk}$$

for all $1 \leq i < j$ and $1 \leq k < m$. An assignment problem instance with a cost matrix having the Strong Monge Property also has a non-crossing optimal solution.

Burkard (1985) lists several constructions that produce cost matrices having the Strong Monge Property. For example, if we associate non-negative weights $(U_s: s \in S)$ and $(V_t: t \in T)$ with the sources and destinations, the cost matrix $C = (c_{ij})$ with

$$c_{ij} = |U_i - V_j|^p \tag{3.1}$$

has the Strong Monge Property for each $p \geq 1$ (Burdyuk and Trofimov (1976)).

While working on the problem of relocating desks along a circular corridor, Karp and Li (1975) considered special cases of the assignment problem in which the cost of assigning a

source s to a destination t is defined by the displacement between s and t either on a line or on a circle. On a line, the displacement between two points s and t, $d(s,t) = |s-t|$, is the special case of the Burdyuk and Trofimov function (3.1) with $p = 1$ and the weights $(U_s: s \in S)$ and $(V_t: t \in T)$ corresponding to the positions of the points. On a circle of circumference H, the displacement between two points s and t is the minimum of the clockwise and counterclockwise displacements:

$$d(s,t) = \min \{r(s,t), r(t,s)\},$$

where $r(s,t)$ is the clockwise displacement from s to t:

$$r(s,t) := \begin{cases} t-s & \text{if } s \le t \\ t + H - s & \text{if } s > t \end{cases}$$

Karp and Li showed that there is a non-crossing optimal solution for these costs and developed an efficient procedure for finding a non-crossing optimal assignment. When the sources and destinations are distinct, their procedure requires $O(|S|*\log|S|)$ time. Otherwise, their method requires $O(|S|^2)$ time.

In this chapter we consider a generalization of the problem of Karp and Li in which cost is a convex function of displacement.

3.3 The Assignment Problem on the Line

In this section, we consider those assignment problems, called **assignment problems on the line**, in which we seek a minimum cost assignment of T into S with the cost of assigning source s to destination t a function f of the (linear) displacement $s-t$. We denote this problem by (S,T,f). In particular, we consider those assignment problems on the line (S,T,f) in which $f:R\rightarrow R$ is a convex function.

Convex functions have been used extensively to model production change costs incurred when changing away from a naturally efficient operational level, reflecting situations where these changes lead to the use of increasingly less efficient or more costly processes (Hax and Candea (1984), Johnson and Montgomery (1974), and Holt, Modigliani, Muth and Simon (1960)). In our application to aircraft scheduling the assignment costs reflect the penalties incurred for expediting or delaying aircraft operations to meet a customer service schedule. The primary method AMC uses to expedite missions is through the use of additional aircrew members (Borsi (1993)). We now show that these additional aircrew costs can be modeled as a convex function of mission disruption.

An airlift mission that uses a single crew spends almost two-thirds of its time on the ground, with about one half of this used for crew rest. Additional crew members can augment or replace the original crew and reduce the ground time.

Replacing an entire crew can eliminate about one half of all ground time, in effect doubling the distance that can be flown in a day, thereby expediting that mission by one day. In addition, each day a mission is delayed results in keeping its crew away from their home base for an additional day. Each day a crew member is away from his home base costs AMC $50 in temporary duty pay. The typical C-141 crew consists of 5 individuals. Therefore, each day a C-141 mission is delayed or expedited directly costs AMC $250 in temporary duty pay. This does not include the costs of transporting extra crews throughout the transportation network or penalties for idle time incurred while a crew, away from their home base, waits for an assigned mission to arrive. When small changes are made to the length of a mission, these additional costs are kept relatively small by using the most convenient and least expensive methods to augment the crew. For instance, when a single crew change occurs, it may be possible to replace that crew at a major hub of operations where the cost of prepositioning that crew is small and it is easy to return the used crew members to their home base or use them to replace other crews. However, as larger changes are made to a mission, less convenient and more costly methods are typically required to preposition the relief crews and return crew members to their home base. When several crew changes must occur, some of these changes would be required at locations

which are not convenient for the prepositioning or reuse of crews. Therefore, the total cost of crew replacement is an increasing function of the number of crews required and can be approximated as a convex function of mission disruption.

Since other options to expedite missions are used infrequently compared to the use of extra crew members (Borsi (1993)), disruption costs are heavily influenced by these crew costs. Therefore, we model disruption costs as a convex function of the total number of days that a mission is expedited or delayed.

Convex functions have the following special properties, which are useful in developing a solution methodology for the assignment problem on the line.

Lemma 3.1 Let $f:R \rightarrow R$ be convex and consider two real numbers x and y with $f(x) < f(y)$. If $x < y$ $(y < x)$ then $f(z) < f(y)$ for each z such that $x \leq z < y$ $(y < z \leq x)$.

Proof: If $x \leq z < y$ then there is a real number α, such that $0 < \alpha \leq 1$ and $z = \alpha x + (1-\alpha)y$. By the convexity of f, $f(\alpha x + (1-\alpha)y) \leq \alpha f(x) + (1-\alpha)f(y)$. Since $f(x) < f(y)$ and $\alpha \neq 0$, $\alpha f(x) + (1-\alpha)f(y) < f(y)$. Therefore, when $x \leq z < y$, $f(z) < f(y)$. A similar argument holds if $y < z \leq x$. □

Lemma 3.2 Given real numbers a, b, and Δ with $b \le a$ and $0 \le \Delta \le a-b$, if $f:[b,a]\to R$ is convex, then $f(a-\Delta) + f(b+\Delta) \le f(a) + f(b)$.

Proof: Since $b \le a$ and $\Delta \le a-b$, λ defined by

$$\lambda = \frac{(a-b)-\Delta}{a-b}$$

satisfies $0 \le \lambda \le 1$. Note that

$$a - \Delta = (1 - \lambda)b + \lambda a, \text{ and}$$
$$b + \Delta = (1 - \lambda)a + \lambda b.$$

Since f is convex, it follows that

$$f(a-\Delta) \le \lambda f(a) + (1-\lambda) f(b)$$
$$f(b+\Delta) \le (1-\lambda) f(a) + \lambda f(b)$$

Therefore, $f(a-\Delta) + f(b+\Delta) \le f(a) + f(b)$ as desired. □

Lemma 3.3 generalizes the results of Burdyuk and Trofimov (1976) by showing that $C_L(S,T,f)$ has the Strong Monge Property whenever $|S| = |T|$ and f is convex.

Lemma 3.3 Let $f:R\to R$ be convex. Then the cost matrix $C_L(S,T,f)$ has the Strong Monge Property for all finite sets S and T of points in R with $|S| = |T|$.

Proof: Given (S,T,f) with $f:R\to R$ convex, let $s \prec s'$ and $t \prec t'$ be members of S and T respectively. Let $a = s'-t$, $b = s-t'$ and $\Delta = t'-t$. Then $b \le a$ and $0 \le \Delta \le a-b$. By Lemma 3.2

we have

$$f(a-\Delta) + f(b+\Delta) \leq f(a) + f(b)$$

which implies that

$$f(s'-t') + f(s-t) \leq f(s'-t) + f(s-t').$$

Therefore, $C_L(s',t') + C_L(s,t) \leq C_L(s',t) + C_L(s,t')$. Since this result holds for arbitrary s, s', t, and t' with $s \prec s'$ and $t \prec t'$, $C_L(S,T,f)$ has the Strong Monge Property. □

In fact, the following lemma shows that convex functions are the only continuous functions with this property.

Lemma 3.4 Let $f:R \to R$ be continuous. If the cost matrix $C_L(S,T,f)$ has the Strong Monge Property for all finite sets S and T of points in R with $|S| = |T|$, then f is convex.

Proof: Assume to the contrary that f is not convex. Then there exists an interval $[a,b]$ and a value $\lambda \in (0,1)$ so that

$$\lambda f(a) + (1-\lambda) f(b) < f(\lambda a + (1-\lambda) b). \tag{3.2}$$

Let $z = \lambda a + (1-\lambda)b$. Consider the line segment connecting $f(a)$ and $f(b)$. By (3.2) we know that $f(z)$ is above this line and, since f is continuous, z is on a range of values (a',b') with functional values above this line segment. Formally, we

define this range as follows. Let λ' be the smallest value on $(\lambda,1]$ for which

$$f(\lambda' a + (1-\lambda')b) = \lambda' f(a) + (1-\lambda') f(b). \tag{3.3}$$

Since f is continuous, we know that (λ,λ') is not empty. Also, since λ' is the smallest value on the range $(\lambda,1]$ which satisfies (3.3) we know that for each $\gamma \in (\lambda,\lambda')$

$$f(\gamma a + (1-\gamma)b) > \gamma f(a) + (1-\gamma) f(b).$$

Similarly, let $\lambda"$ be the largest value on $[0,\lambda)$ for which

$$f(\lambda" a + (1-\lambda")b) = \lambda" f(a) + (1-\lambda") f(b). \tag{3.4}$$

Let $a' = \lambda' a + (1-\lambda')b$ and $b' = \lambda" a + (1-\lambda")b$ and note that $a' < b'$.

By construction, for each $\alpha \in (0,1)$

$$\alpha f(a') + (1-\alpha) f(b') < f(\alpha a' + (1-\alpha) b'). \tag{3.5}$$

Choosing $\delta = (b'-a')/4$, let $\alpha' = 0.75$. By equation (3.5):

$$f(a'+\delta) > \alpha' f(a') + (1-\alpha') f(b'), \text{ and} \tag{3.6}$$

$$f(b'-\delta) > (1-\alpha') f(a') + \alpha' f(b'). \tag{3.7}$$

Summing (3.6) and (3.7) yields

$$f(a'+\delta) + f(b'-\delta) > f(a') + f(b'). \tag{3.8}$$

We now define sets S and T for which $C_L(S,T,f)$ does not have the Strong Monge Property. Let $S = \{s,s'\}$ and $T = \{t,t'\}$ with $t = 0$, $t' = \delta$, $s = a'+\delta$, and $s' = b'$. Note that $t < t'$ and $s < s'$. Then by (3.8), $c_L(s,t) + c_L(s',t') > c_L(s,t') + c_L(s',t)$ and $C_L(S,T,f)$ does not have the Strong Monge Property. Therefore, we have our desired contradiction and f must be convex. □

One immediate consequence of Lemma 3.3 is:

Theorem 3.5 If $|S| = |T|$, then the assignment problem (S,T,f) has a non-crossing optimal assignment whenever $f:R\rightarrow R$ is convex.

Therefore, when $|S| = |T|$ the unique non-crossing assignment of sources onto destinations is optimal. This non-crossing assignment can be constructed in the $O(|S|\log|S|)$ time required to sort the sources and destinations. In fact, if we know that $C_L(S,T,f)$ has the (Strong) Monge Property, we do not even need to compute the cost matrix in order to construct an optimal solution.

In the remainder of this section, we concentrate on the special case of the problem (S,T,f) in which $f:R\rightarrow R$ is convex and $|S| > |T|$. The following corollary shows that this problem also admits a non-crossing optimal solution.

Corollary 3.6 If $f:R \rightarrow R$ is convex, the problem (S,T,f) has a non-crossing optimal assignment.

Proof: Consider an optimal assignment μ. By Theorem 3.5 the problem $(\mu(T),T,f)$ admits a non-crossing optimal assignment μ'. Since μ assigns T into $\mu(T)$, and μ' is an optimal assignment of T into $\mu(T)$, $Cost(\mu') \leq Cost(\mu)$; proving that there is a non-crossing optimal assignment μ' of T into S. □

When $|S| > |T|$ there are many non-crossing assignments of the destinations into the sources, but not all are necessarily optimal. Given a problem (S,T,f) we develop an optimal solution by considering a series of **partial assignments** μ in which an identified subset of destinations is matched into S and every other destination is unmatched. The following definitions apply to both assignments and partial assignments.

When $|S| > |T|$ we adopt the approach of identifying contiguous subsets $T' \subseteq T$ and $S' \subseteq S$ with $|S'| = |T'|$ such that there exists an optimal non-crossing assignment μ for (S,T',f) with $\mu(T') = S'$. Two sources s_i and s_k with $i \leq k$ define the **contiguous set**: $[s_i,s_k] = \{s_j \in S: i \leq j \leq k\}$ of sources. Analogously, two destinations t_i and t_k define the contiguous set $[t_i,t_k]$ of destinations. Given a contiguous set $X = [a,b]$ of sources or destinations, we refer to a as the left endpoint, denoted $\ell(X)$, and to b as the right endpoint,

denoted $r(X)$.

A contiguous set $[t_i, t_k]$ of destinations is a **block** of the assignment μ if μ maps $[t_i, t_k]$ onto a contiguous set of sources. Note that each destination t is a block of every assignment μ. A block $[t_i, t_k]$ of an assignment μ that is not properly contained in any other block of μ is called a maximal block of μ.

Let $[a,b]$ be a block of the non-crossing assignment μ and suppose $\mu(a) \equiv s_i$ and $\mu(b) \equiv s_j$. If $i > 1$ and the source s_{i-1} is not matched under μ, we define the **left shift of μ with respect to** $[a,b]$ to be the non-crossing assignment μ' with $\mu'(a) \equiv s_{i-1}$, $\mu'(s_p) \equiv \mu(s_{p+1})$ for $s_p \in [s_i, s_{j-1}]$, and $\mu'(t) \equiv \mu(t)$ for $t \in T \setminus [a,b]$. Analogously, if $j < |S|$ and the source s_{j+1} is unmatched under μ, we define the **right shift of μ with respect to** $[a,b]$ as the non-crossing assignment μ' with $\mu'(b) \equiv s_{j+1}$, $\mu'(s_p) \equiv \mu(s_{p-1})$ for $s_p \in [s_{i+1}, s_j]$, and $\mu'(t) \equiv \mu(t)$ for $t \in T \setminus [a,b]$. Therefore, a series of k left (right) shifts of μ with respect to the block $[a,b]$ is defined if $i \geq k+1$ ($j \leq |S|-k$) and no source in $[s_{i-k}, s_{i-1}]$ ($[s_{j+1}, s_{j+k}]$) is matched under μ. If μ' is the assignment resulting from a shift of μ the **change in cost** due to this shift is $\text{Cost}(\mu') - \text{Cost}(\mu)$. Note that a maximal block $[a,b]$ with respect to a non-crossing assignment μ may not be a maximal block after a shift of μ.

We associate with each destination t a set $S^*(t)$, called the **min-cost set** of t, defined by $S^*(t) = \{s \in S: c_L(s,t) \leq$

$c_L(s',t)$ for each $s' \in S\}$. The following two lemmas present useful properties related to min-cost sets.

Lemma 3.7 Consider the problem (S,T,f) with $f:R\to R$ convex. For each $t \in T$, $S^*(t)$ is a contiguous set of sources.

Proof: Consider an arbitrary destination t. Clearly $|S^*(t)| \geq 1$. If $|S^*(t)| = 1$, $S^*(t)$ is contiguous. If $|S^*(t)| > 1$, assume sources s_i and s_k are members of $S^*(t)$ and consider a source s_j with $i < j < k$. Then $s_j \in [s_i, s_k]$ and there exists $\lambda \in [0,1]$ such that $\lambda s_i + (1-\lambda)s_k = s_j$. Since $f:R\to R$ is convex and $f(s_i) = f(s_k)$, it follows that

$$f(s_j) \leq \lambda f(s_i) + (1-\lambda)f(s_k) = f(s_i).$$

Since $s_i \in S^*(t)$, $f(s_i) \leq f(s_j)$ and, hence, $f(s_j) = f(s_i)$ and $s_j \in S^*(t)$.

We have shown that if two sources s_i and s_j are in $S^*(t)$, then so is each s_k with $i < j < k$. Therefore, $S^*(t)$ is a contiguous set. □

Lemma 3.8 Consider the problem (S,T,f) with $f:R\to R$ convex. Given sources s' and s'' and destination t, if either $s' \prec s'' \preceq r(S^*(t))$ or $\ell(S^*(t)) \preceq s'' \prec s'$, then $c_L(s'',t) \leq c_L(s',t)$.

Proof: If $s' = s''$ the result is immediate. Otherwise consider the case in which $s' \neq s''$ and $s' \prec s'' \preceq r(S^*(t))$. If $s' \in S^*(t)$, then, by Lemma 3.7, $s'' \in S^*(t)$ and $c_L(s',t) =$

$c_L(s'',t)$. If $s' \notin S^*(t)$, then $f(s'-t) > f(s-t)$ for any $s \in S^*(t)$ and by Lemma 3.1, $c_L(s'',t) < c_L(s',t)$. The argument for the case in which $s' \neq s''$ and $\ell(S^*(t)) \preceq s'' \prec s'$ is similar. □

Finally, note that since the cost function f is the same for each destination, two destinations t and t' with $t \prec t'$ must have $\ell(S^*(t)) \preceq \ell(S^*(t'))$. Using these properties, we develop conditions sufficient to guarantee an assignment is optimal.

We say that a maximal block $[t,t']$ of an assignment μ is **valid** if $\mu(t) \preceq r(S^*(t))$ and $\ell(S^*(t)) \preceq \mu(t')$. An assignment μ is valid if and only if $\mu(a) \preceq r(S^*(a))$ and $\ell(S^*(b)) \preceq \mu(b)$ for each maximal block $[a,b]$ of μ. Lemma 3.9 shows that we may restrict our attention to valid non-crossing assignments.

Lemma 3.9 Consider the problem (S,T,f). If $f:R \to R$ is convex, there is a valid non-crossing optimal assignment of T into S.

Proof: If there are no valid non-crossing optimal assignments of T into S, then in the set of all non-crossing optimal assignments there must be a non-crossing optimal assignment μ with the fewest number of non-valid maximal blocks. We now develop another non-crossing optimal assignment with fewer non-valid maximal blocks, showing a contradiction and proving the lemma.

Suppose $[a,b]$ is a non-valid maximal block of μ and assume that $r(S^*(a)) \prec s_i \equiv \mu(a)$. Since $[a,b]$ is a maximal block of μ, there is a maximal contiguous set of unmatched sources $[s_j, s_{i-1}]$. Consider the assignment μ' defined by

$$\mu'(a) \equiv \begin{cases} r(S^*(a)), \text{ if } r(S^*(a)) \in [s_j, s_{i-1}] \\ s_j, \text{ otherwise,} \end{cases}$$

and $\mu'(t) \equiv \mu(t)$ for $t \in T \backslash a$. Note that μ' is non-crossing, and by Lemma 3.8, $\text{Cost}(\mu') \leq \text{Cost}(\mu)$. Also, note that a becomes the right endpoint of a maximal block $[z,a]$ of μ' and $\ell(S^*(a)) \preceq \mu'(a)$. Thus, this reassignment of a does not increase the number of non-valid maximal blocks. A similar argument applies if $\mu(b) \prec \ell(S^*(b))$, except in this case the reassignment of b corresponds to a series of right shifts of μ with respect to b.

Since there are a finite number of destinations in $[a,b]$, applying the above reassignment procedure a finite number of times will result in an optimal non-crossing assignment with strictly fewer non-valid maximal blocks. Therefore, we have our desired contradiction and there must exist a valid non-crossing optimal assignment of T into S. □

A non-crossing assignment μ is **locally optimal** if it is a valid assignment and no shift with respect to a block of μ leads to a lower cost assignment. Given a block $[a,b]$ of an

assignment μ, we say that μ is locally optimal for $[a,b]$ if the partial assignment μ' with $\mu'(t) \equiv \mu(t)$ for each $t \in [a,b]$ is locally optimal. The following lemmas establish a property of locally optimal assignments, which we use in Theorem 3.13 to show that locally optimal non-crossing assignments are optimal.

Lemma 3.10 Let $f:R \to R$ be convex and consider real numbers t, t', s, and s'. If $t \leq t'$ and $s \leq s'$ then $f(s-t) - f(s'-t) \leq f(s-t') - f(s'-t')$.

Proof: By Lemma 3.3, the cost matrix $C_L(\{s,s'\},\{t,t'\},f)$ has the Strong Monge Property. Therefore,

$$f(s-t) + f(s'-t') \leq f(s'-t) + f(s-t')$$

which implies that

$$f(s-t) - f(s'-t) \leq f(s-t') - f(s'-t'),$$

and the proof is complete. □

Lemma 3.11 Let μ be a locally optimal non-crossing assignment μ for the problem (S,T,f) with $f:R \to R$ convex, and let $[t_i, t_k]$ be a maximal block of μ with destination $t_j \in [t_i, t_k]$. Consider a series of left (right) shifts of μ with respect to $[t_i, t_j]$ ($[t_j, t_k]$) resulting in an assignment μ' with $\ell(S^*(t_j)) \preceq \mu'(t_j)$ ($\mu'(t_j) \preceq r(S^*(t_j))$), and $\mu'(t) \equiv \mu(t)$ for $t \in$

$T\backslash[t_i, t_j]$ ($t \in T\backslash[t_j, t_k]$). The change in cost due to each of the shifts in this series is non-negative.

Proof: Consider the series of left shifts of μ with respect to $[t_i, t_j]$. Let Δ_z denote the change in cost due to the z^{th} shift in this series, and let $\Delta_{[p,z]}$ designate the change in assignment cost of destination $t_p \in [t_i, t_j]$ due to the z^{th} left shift. Note that since μ is a valid assignment, $\mu(t_i) \leq r(S^*(t_i))$. Therefore, by Lemma 3.8, $\Delta_{[i,z]} \geq 0$ for $z \geq 1$. Therefore, if i = j, the result is proven. Now, consider the case in which i < j.

We are given that $\Delta_1 \geq 0$ and as our induction hypothesis assume that $\Delta_{z-1} \geq 0$. In a series of at least z left shifts of μ with respect to $[t_i, t_j]$

$$\Delta_{z-1} = \sum_{p=i,j-1} \Delta_{[p,z-1]} + \Delta_{[j,z-1]}, \text{ and}$$

$$\Delta_z = \Delta_{[i,z]} + \sum_{p=i+1,j} \Delta_{[p,z]}.$$

Examining Δ_{z-1} and Δ_z we see that:

(i) $\Delta_{[i,z]} \geq 0$. Since $z \geq 1$ and μ is valid, this follows from Lemma 3.8.

(ii) $\Delta_{[p+1,z]} \geq \Delta_{[p,z-1]}$ for $i \leq p \leq j-1$. Since we are considering non-crossing assignments, if shift z-1 reassigns destination t_p from s_q to s_{q-1} then shift z reassigns destination t_{p+1} from s_q to s_{q-1} and this inequality follows from Lemma 3.10.

(iii) $\Delta_{[j,z-1]} \leq 0$. Since, by assumption, the z^{th} left shift resulted in an assignment μ' with $\ell(S^*(t_j)) \leq \mu'(t_j)$, then Lemma 3.8 applies and $\Delta_{[j,z-1]} \leq 0$.

Therefore, $\Delta_z - \Delta_{z-1} \geq 0$. Since $\Delta_{z-1} \geq 0$ by the induction hypothesis, Δ_z must also be non-negative. The argument for a series of right shifts of μ with respect to $[t_j, t_k]$ is similar. □

The following lemma extends the results of Lemma 3.11.

Lemma 3.12 Consider a locally optimal non-crossing assignment μ for the problem (S,T,f) with $f:R\rightarrow R$ convex. Let $[t_i, t_k]$ be a maximal block of μ with $t_j \in [t_i, t_k]$. The change in assignment costs due to each shift in a series of left (right) shifts of μ with respect to $[t_i, t_j]$ ($[t_j, t_k]$) is non-negative.

Proof: As in Lemma 3.11 let Δ_z denote the change in cost due to the z^{th} left shift of μ and let $\Delta_{[q,z]}$ denote the change in assignment costs of destination t_q due to the z^{th} left shift of μ.

Consider a series of left shifts of μ with respect to $[t_i, t_j]$. Since μ is locally optimal we know that $\Delta_1 \geq 0$. Assume that $\Delta_{z-1} \geq 0$ and let μ' be the assignment after z-1 shifts. If $\ell(S^*(t_j)) \leq \mu'(t_j)$, then, by Lemma 3.11, $\Delta_z \geq 0$. Otherwise, let t_p be the highest indexed destination in $[t_i, t_j]$ for which $\ell(S^*(t_p)) \leq \mu'(t_p)$. If t_p does not exist, then $\mu'(t_q)$

$< \ell(S^*(t_q))$ for each $t_q \in [t_i, t_j]$ and, by Lemma 3.8, $\Delta_{[q,z]} \geq 0$ which implies $\Delta_z \geq 0$.

When t_p exists the change in assignment costs due to the z^{th} shift is:

$$\sum_{x=i,p} \Delta_{[x,z]} + \sum_{x=p+1,j} \Delta_{[x,z]}.$$

By Lemma 3.11 the first term is non-negative and by Lemma 3.8 the second term is non-negative. Therefore, $\Delta_z \geq 0$.

The proof for a series of right shifts is similar. □

Based on these results we now define a sufficient condition for the optimality of an assignment.

Theorem 3.13 If $f:R \to R$ is convex, then a locally optimal non-crossing assignment for the problem (S,T,f) is optimal.

Proof: Consider a locally optimal non-crossing assignment μ and let $[a,b]$ be a maximal block of μ. We now show that the partial assignment μ' defined by $\mu'(t) \equiv \mu(t)$ for each $t \in [a,b]$ is an optimal assignment for the problem $(S,[a,b],f)$.

Let μ" be an assignment for $(S,[a,b],f)$. The following procedure transforms μ" into a new assignment μ^* for which $\mu^*(t) \equiv \mu'(t)$ for each $t \in [a,b]$. Throughout this procedure we use μ^* to designate the "current" assignment which includes all changes made after the initialization of μ^* in STEP 1.

STEP 1: Initialize μ^* as the unique non-crossing assignment of $[a,b]$ onto $\mu''([a,b])$.

STEP 2: Partition the destinations into three contiguous sets L, E, and R, defined as follows:

$$\begin{aligned} L &:= \{t \in [a,b]: \ \mu^*(t) \prec \mu'(t)\} \\ E &:= \{t \in [a,b]: \ \mu^*(t) \equiv \mu'(t)\} \\ R &:= \{t \in [a,b]: \ \mu^*(t) \succ \mu'(t)\}. \end{aligned}$$

STEP 3: Identify $[a,t]$, the maximal block of μ^* containing destination a. If $a \in L$, perform a right shift of μ^* with respect to $[a,t]$. Update sets L and E as necessary. Repeat STEP 3 until μ^* is not changed.

STEP 4: Identify $[t,b]$, the maximal block of μ^* containing destination b. If $b \in R$, perform a left shift of μ^* with respect to $[t,b]$. Update sets R and E as necessary. Repeat STEP 4 until μ^* is not changed.

We now show that this procedure must halt after a finite number of steps. A shift of a non-crossing assignment results in a non-crossing assignment; therefore, in each step of this procedure μ^* is non-crossing. Since μ' is also non-crossing, L, E, and R remain contiguous throughout the above procedure. If L is not empty in STEP 2, then $a \in L$ and since only right shifts are accomplished with respect to members of L, a finite number of right shifts in STEP 3 results in $a \in E$. Likewise, a finite number of shifts in STEP 4 result in b being a member of E. Therefore, in a finite number of steps the above procedure terminates with $\mu^*(a) \equiv \mu'(a)$ and $\mu^*(b) \equiv \mu'(b)$. Since μ^* is non-crossing, at this time $\mu'(t) \equiv \mu^*(t)$ for each $t \in [a,b]$.

Now consider the cost of the assignment μ^*. As a result of Theorem 3.5, in STEP 1, $\text{Cost}(\mu^*) \leq \text{Cost}(\mu'')$. Any shifts

performed in STEPs 3 and 4 correspond to the reversal of a shift considered in Lemma 3.12; therefore, these shifts do not increase the cost of the assignment. As a result, at any stage of the given procedure $\sum_{t\in[a,b]} c_{\mu^*(t),t} \leq \sum_{t\in[a,b]} c_{\mu''(t),t}$. Since $\mu^*(t) \equiv \mu'(t)$ for each $t \in [a,b]$, there cannot exist an assignment of the destinations in block $[a,b]$ of strictly lower cost than their assignment under μ'.

Since this result holds for every maximal block of μ, μ must be an optimal assignment. □

Our approach to solving the problem (S,T,f) with f convex is to develop a series of non-crossing locally optimal partial assignments. Our solution approach starts with an optimal partial assignment for a single destination, then iteratively develops an optimal solution for larger sets of destinations until all destinations are matched. After each iteration of the procedure we have a partial assignment that is locally optimal for each of its maximal blocks. During the next iteration, a left shift of the current assignment with respect to a maximal block $[a,b]$ may result in an assignment for which that block is not maximal. When this happens we say that $[a,b]$ is adjacent to another block. Contiguous sets of destinations (sources) are **adjacent** if the sets are disjoint and their union forms a contiguous set. Given an assignment μ, two blocks $[t_i,t_j]$ and $[t_k,t_m]$ with $j < k$ are adjacent if

their respective sets of destinations and matched sources are adjacent and $\mu(t_j) \prec \mu(t_k)$. The following lemma shows that if an assignment μ is locally optimal for two adjacent blocks, then it is locally optimal for the larger block which is the union of these adjacent blocks.

Lemma 3.14 Consider a non-crossing assignment μ for the problem (S,T,f) with $f:R\to R$ convex. If μ is locally optimal for two adjacent blocks $[a,b]$ and $[c,d]$ with $b \prec c$, then it is locally optimal for the block $[a,d]$.

Proof: Define the partial assignment μ' by $\mu'(t) \equiv \mu(t)$ for each $t \in [a,d]$. Note that $[a,d]$ is a block of μ' since $[a,b]$ and $[c,d]$ are adjacent with $b \prec c$. Since μ is locally optimal for $[a,b]$ and $[c,d]$, $\mu'(a) \preceq r(S^*(a))$ and $\ell(S^*(d)) \preceq \mu'(d)$. Therefore, μ' is valid.

We now show that a left shift of μ' with respect to a subset of $[a,d]$ will not reduce assignment costs. Consider a left shift of μ' with respect to a contiguous subset $[a,z] \subseteq [a,d]$. Since μ is locally optimal for $[a,b]$, if $z \in [a,b]$, then a left shift of μ' with respect to $[a,z]$ will not reduce assignment costs. If $z \in [c,d]$, then a left shift of μ' with respect to $[a,z]$ can be decomposed into a left shift of μ' with respect to $[a,b]$ followed by a left shift of the resulting assignment with respect to $[c,z]$. Again, since μ is locally optimal for the blocks $[a,b]$ and $[c,d]$, these shifts

cannot reduce the total assignment cost of μ'.

The argument for right shifts of μ' with respect to subsets of $[a,d]$ is similar. Therefore, μ' is locally optimal and by definition μ is locally optimal for $[a,d]$. □

Note that a partial assignment of t_1 to a source in $S^*(t_1)$ is non-crossing and locally optimal. In order to ensure that at each iteration we have a non-crossing assignment our procedure assigns destinations based on their indexed order. Given a non-crossing locally optimal partial assignment μ of $[t_1, t_j]$ at the end of one iteration, the procedure develops an assignment for $[t_1, t_{j+1}]$ in the next iteration. The following lemma describes a situation in which it is easy to find a non-crossing assignment μ that is locally optimal for the set $[t_1, t_{j+1}]$.

Lemma 3.15 Consider a problem (S,T,f) with $f:R\rightarrow R$ convex. Let μ be a non-crossing locally optimal partial assignment of $[t_1, t_j]$. If $\mu(t_j) \prec \ell(S^*(t_{j+1}))$, then the partial assignment μ' with $\mu'(t) \equiv \mu(t)$ for each $t \in [t_1, t_j]$ and $\mu'(t_{j+1}) \equiv \ell(S^*(t_{j+1}))$ is non-crossing and locally optimal.

Proof: Since μ is non-crossing and $\mu'(t_j) \prec \mu'(t_{j+1})$, μ' is non-crossing. Now consider the maximal block $[t, t_{j+1}]$ of μ'. Since $\mu'(t_{j+1}) \in S^*(t_{j+1})$, μ' is locally optimal for t_{j+1}. Therefore, if $t \equiv t_{j+1}$, then μ' is locally optimal for this

block. Otherwise, Lemma 3.14 applies, and again μ' is locally optimal for this block.

Since μ is locally optimal for every other maximal block of μ', the assignment μ' is locally optimal and the proof is complete. □

When the conditions of Lemma 3.15 are not met we use the procedure SINGLE-MERGE below to generate an assignment for $[t_i, t_{j+1}]$. Consider a non-crossing partial assignment μ that is locally optimal for a maximal block $[t_i, t_j]$. If t_{j+1} is unmatched and $\ell(S^*(t_{j+1})) \leq \mu(t_j)$ then SINGLE-MERGE generates an assignment μ' for the block $[t_i, t_{j+1}]$.

SINGLE-MERGE

STEP 1: Let $s_p \equiv \mu(t_j)$. Initialize μ' as follows: If $p \neq |S|$, then $\mu'(t) \equiv \mu(t)$ for each $t \in [t_i, t_j]$ and $\mu'(t_{j+1}) \equiv s_{p+1}$; otherwise, let μ' be the assignment resulting from a left shift of μ with respect to $[t_i, t_j]$ and let $\mu'(t_{j+1}) \equiv s_p$.

STEP 2: If a left shift of μ with respect to $[t_i, t_j]$ was not performed in STEP 1 and a left shift of μ' with respect to $[t_i, t_{j+1}]$ would reduce assignment costs, perform this shift.

Lemma 3.16 establishes conditions under which SINGLE-MERGE generates a non-crossing locally optimal assignment.

Lemma 3.16 Consider the problem (S,T,f) with f:$R \to R$ convex. Let μ be a non-crossing locally optimal partial assignment of

$[t_1, t_j]$ with maximal block $[t_i, t_j]$. If $\ell(S^*(t_{j+1})) \preceq \mu(t_j)$, then SINGLE-MERGE generates a non-crossing locally optimal partial assignment μ' for $[t_1, t_{j+1}]$.

Proof: SINGLE-MERGE does not change the assignments of those assigned destinations not in $[t_i, t_j]$. Therefore, to show that μ' is non-crossing and locally optimal we only need consider the maximal block $[a, t_{j+1}]$ of μ'. By Lemma 3.14, we only need to show that μ' is non-crossing and locally optimal for $[t_i, t_{j+1}]$.

In STEP 1, $\mu'(t_j) \prec \mu'(t_{j+1})$. Since a shift maintains the relative order of assigned destinations and sources, μ' is non-crossing at the end of STEP 2.

We are given that μ is valid and $\ell(S^*(t_{j+1})) \preceq \mu(t_j)$. Since $t_j \prec t_{j+1}$, $\ell(S^*(t_j)) \preceq \ell(S^*(t_{j+1})) \preceq \mu(t_j)$. Since SINGLE-MERGE performs at most a single left shift, after STEP 2 $\ell(S^*(t_{j+1})) \preceq \mu'(t_{j+1})$ and $\mu'(t_i) \preceq \mu(t_i) \preceq r(S^*(t_i))$. Therefore, at the end of this procedure μ' is valid.

We now show that the partial assignment μ'' defined by $\mu''(t) \equiv \mu'(t)$ for each $t \in [t_i, t_{j+1}]$ is locally optimal. From the above argument we know that μ'' is non-crossing and valid. Therefore, we now consider the change in assignment cost caused by left or right shifts of μ'' with respect to subsets of $[t_i, t_{j+1}]$. Using the notation introduced in Lemma 3.11, $\Delta_{[x,0]}$ ($\Delta_{[x,-1]}$) denotes the change in assignment cost due to a left (right) shift of μ with respect to destination t_x. Since

μ is locally optimal we have

(3.9) $$\sum_{x=i,p}\Delta_{[x,0]} \geq 0, \text{ for } t_p \in [t_i, t_j]$$

(3.10) $$\sum_{x=p,j}\Delta_{[x,-1]} \geq 0, \text{ for } t_p \in [t_i, t_j]$$

If a left shift is not performed in STEP 1 or STEP 2, then a left shift of μ" with respect to $[t_i, t_{j+1}]$ will not reduce costs. In addition, $\mu(t) \equiv \mu"(t)$ for each $t \in [t_i, t_j]$. Therefore, by (3.9) a left shift of μ" with respect to $[t_i, t_p] \subset [t_i, t_{j+1}]$ will not reduce the assignment cost. Finally, since $\ell(S^*(t_{j+1})) \leq \mu"(t_{j+1})$, by Lemma 3.8 $\Delta_{[j+1,-1]} \geq 0$, and by (3.10), a right shift of μ" with respect to $[t_p, t_{j+1}] \subseteq [t_i, t_{j+1}]$ will not reduce assignment costs.

Now consider the case in which a left shift is made in either STEP 1 or STEP 2. First consider a left shift of μ", letting $\Delta_{[p,1]}$ denote the change in cost due to a left shift of μ" with respect to $t_p \in [t_i, t_{j+1}]$. Since μ" is valid, by Lemma 3.8 a left shift of μ" with respect to t_i will not reduce costs and $\Delta_{[i,1]} \geq 0$. The change in assignment cost due to a left shift of μ" with respect to a block $[t_i, t_p] \subseteq [t_i, t_{j+1}]$ with i < p is

(3.11) $$\Delta_{[i,1]} + \sum_{x=i+1,p}\Delta_{[x,1]}.$$

As mentioned earlier, $\Delta_{[i,1]} \geq 0$. By Lemma 3.10 and (3.9),

$$0 \leq \sum_{x=i,p-1}\Delta_{[x,0]} \leq \sum_{x=i+1,p}\Delta_{[q+1,1]}.$$

Therefore, a left shift of μ'' with respect to a subset of $[t_i, t_{j+1}]$ does not reduce assignment costs.

Now consider right shifts of μ'' after a left shift in STEP 1 or STEP 2. If $\mu''(t_{j+1}) \equiv s_{|S|}$ a right shift is not defined. Otherwise, the left shift was performed in STEP 2 and reduced assignment costs. Thus, a right shift of μ'' with respect to $[t_i, t_{j+1}]$ cannot reduce assignment costs. A right shift of μ'' with respect to $[t_p, t_{j+1}]$ for $t_p \in [t_{i+1}, t_{j+1}]$ can be decomposed into a right shift of μ'' with respect to $[t_i, t_{j+1}]$, which does not reduce assignment costs, followed by a left shift with respect to $[t_i, t_{p-1}]$. Just prior to this left shift $\mu''(t) \equiv \mu(t)$ for each $t \in [t_i, t_j]$ and, by (3.9), this left shift would not reduce assignment costs. Therefore, a right shift of μ'' with respect to a subset of $[t_i, t_{j+1}]$ does not reduce assignment costs.

To summarize we first showed that μ'' is non-crossing and valid. Then we showed that a shift of μ'' with respect to a block in $[t_i, t_{j+1}]$ would not reduce assignment costs. Therefore, μ'' is locally optimal and the proof is complete. □

The following assignment procedure is based on the results of Lemma 3.15 and Lemma 3.16.

LEFT-SHIFT

STEP 1: Initialization:

STEP 1a: Sort destinations and sources by location, assigning indices so that:

$$s_1 \le s_2 \le \ldots \le s_m \text{ and } t_1 \le t_2 \le \ldots \le t_n.$$

STEP 1b: $i \leftarrow 1$.

STEP 2: Block Assignment of t_i:

STEP 2a: If $\ell(S^*(t_i))$ is not assigned, match it with t_i. Otherwise, perform SINGLE-MERGE on t_i and the maximal block which has a destination currently matched with $\ell(S^*(t_i))$.

STEP 2b: If $i = n$, STOP. Otherwise, $i \leftarrow i + 1$, repeat STEP 2.

In LEFT-SHIFT the destinations are assigned in their indexed order to ensure that the resulting (partial) assignments are non-crossing. Note that in each iteration of STEP 2, no more than one left shift is performed. Also, since only left shifts are performed, once a block is formed, it is a block in each following partial assignment. The inductive argument below shows that LEFT-SHIFT generates an optimal solution to the assignment problem on the line when the assignment costs are determined by a convex function.

Theorem 3.17 If the function $f:R \to R$ is convex, LEFT-SHIFT generates an optimal assignment for the problem (S,T,f).

Proof: Since this procedure iterates through STEP 2 $|T|$ times and no more than one left shift is performed each iteration, this procedure is finite.

We now show that at the end of each cycle through STEP 2, the current partial assignment is locally optimal. Since the initial assignment of t_1 is made to a source in $S^*(t_1)$ this is true for $i = 1$. Assume this is true for the first j destinations processed, and let μ designate this non-crossing locally optimal partial assignment of destinations t_1 through t_j.

Consider the maximal block $[a, t_j]$ of μ. By our assumption $\ell(S^*(t_j)) \preceq \mu(t_j)$. Since only left shifts are performed, destination a was initially matched in STEP 2 with $\ell(S^*(a))$ and $\mu(a) \preceq \ell(S^*(a))$. Therefore, every source in the contiguous set $[\mu(a), \mu(t_j)]$, which contains $[\ell(S^*(a)), \ell(S^*(t_j))]$, is matched by μ. Since $\ell(S^*(t_j)) \preceq \ell(S^*(t_{j+1}))$, either $\mu(t_j) \prec \ell(S^*(t_{j+1}))$ or $S^*(t_{j+1}) \bigcap \mu([a, t_j]) \neq \emptyset$. By Lemma 3.15 and Lemma 3.16, the partial assignment of the set $[t_1, t_{j+1}]$ generated by STEP 2 is locally optimal. In particular, the solution after every destination has been matched is non-crossing and locally optimal, and, by Theorem 3.13, LEFT-SHIFT generates an optimal solution. □

In Section 3.4 we examine the computational complexity of LEFT-SHIFT. In Section 3.5 we present a more efficient

implementation for the case in which assignment costs are piecewise linear.

3.4 Computational Complexity of LEFT-SHIFT

In this section we show that LEFT-SHIFT solves the problem (S,T,f) with $f:R\rightarrow R$ convex in $O(|T|^2 + |S|\log|S|)$ computational steps.

LEFT-SHIFT, as presented in Section 3.3, consists of two steps: STEP 1: a sort of S and T, followed by STEP 2: $|T|$ iterations of an assignment step for each destination.

Based on our assumption that $|S| \geq |T|$, STEP 1 takes no more than $O(|S|\log|S|)$ computational steps.

The complexity of STEP 2 depends on how we store the assignment. Instead of identifying the mate of each destination, a non-crossing partial assignment μ can be defined by identifying each maximal block's assignment. Since μ is non-crossing, the assignment of a block $[t_i, t_j]$ can be stored by noting t_i, t_j, and the mate of t_j. This information is stored in a link list, which we call the **Block-list**. We now show that each iteration of STEP 2 requires no more than a constant number of steps to update this list.

The first iteration of STEP 2, matches t_1 with $\ell(S^*(t_1))$ and initializes the Block-list for the block $[t_1, t_1]$. This takes $O(1)$ steps. Now consider the processing required during iteration $i > 1$ of STEP 2 to match t_i and update the current

partial assignment. Note that at the beginning of this iteration t_{i-1} is the highest indexed destination of a maximal block of the current assignment. First, the procedure checks if $\ell(S^*(t_i))$ is matched by checking if $\ell(S^*(t_i)) \preceq \mu(t_{i-1})$. If $\mu(t_{i-1}) \prec \ell(S^*(t_i))$, then $\ell(S^*(t_i))$ is not matched and a new maximal block $[t_i, t_i]$ is created with $\mu(t_i) \equiv \ell(S^*(t_i))$. This takes O(1) computational steps to update the Block-list. Otherwise, if $\ell(S^*(t_i))$ is matched, a SINGLE-MERGE operation is performed.

The SINGLE-MERGE operation is presented below for the iteration in which we have a locally optimal partial assignment μ of destinations $[t_1, t_{i-1}]$ with maximal block $[t_a, t_{i-1}]$.

SINGLE-MERGE

STEP 1: Assume $s_p \equiv \mu(t_{i-1})$. Initialize μ' as follows: If $p \neq |S|$, then $\mu'(t) \equiv \mu(t)$ for each $t \in [t_1, t_{i-1}]$ and $\mu'(t_i) \equiv s_{p+1}$. Otherwise, let μ' be the assignment resulting from a left shift of μ with respect to $[t_a, t_{i-1}]$ and let $\mu'(t_i) \equiv s_p$.

STEP 2: If a left shift of μ with respect to $[t_a, t_{i-1}]$ was not performed in STEP 1 and a left shift of μ' with respect to $[t_a, t_i]$ would reduce assignment costs, perform this shift.

Consider the steps required to update the Block-list while performing SINGLE-MERGE. If a left shift is not performed, we only need to update the information on $[t_a, t_{i-1}]$, changing this block to $[t_a, t_i]$ and storing the mate of t_i.

Otherwise, assume a shift is performed, resulting in an assignment μ with $\mu(t_a) \equiv s_{a'}$. If a = 1, then we update that block's entry in the Block-list. If a > 1, consider the maximal block $[t', t_{a-1}]$ with $\mu(t_{a-1}) \equiv s_q$. If q < a'-1, then after the shift we simply update the assignment information on $[t_a, t_i]$. However, if q = a'-1, then $[t', t_i]$ is a maximal block after the shift. So we delete the information on block $[t_a, t_{i-1}]$ and update the maximal block entry for block $[t', t_{a-1}]$ to reflect the new maximal block $[t', t_i]$. Therefore, in a single iteration the number of computational steps needed to update block parameters is O(1) and the complexity of SINGLE-MERGE depends on how efficiently we can determine if a left shift reduces costs.

Assume we wish to determine the change in cost Δ_1 due to the left shift of a (partial) assignment μ with respect to block $[t_a, t_i]$. This value is calculated as follows:

$$\Delta_1 = \sum_{x=a,i} \Delta_{[x,1]},$$

where $\Delta_{[x,1]}$ denotes the change in cost for destination t_x. The number of terms in this summation equals the number of destinations in the block. Each term may be calculated as follows:

$$\text{if } \mu(t_x) \equiv s_k, \text{ then } \Delta_{[x,1]} = f(s_{k-1}-t_x) - f(s_k - t_x).$$

Assuming that each functional evaluation takes [illegible] than a

constant number of steps, since there are at most $|T|$ terms, this calculation takes $O(|T|)$ steps.

From the above discussion, the computational complexity of LEFT-SHIFT is a function of the time required to conduct the sort in STEP 1 plus the total number of steps required in $|T|$ iterations of STEP 2 to calculate the change in cost of a left shift. Therefore, the complexity of LEFT-SHIFT is $O(|S|\log|S| + |T|^2)$. We now consider a more efficient approach for the case when the function f is piecewise linear.

3.5 **Computational Complexity for Piecewise Linear Functions**

In Section 3.4, the computational complexity of Left-Shift was shown to be a function of the sort in STEP 1 and the calculation of shift costs in STEP 2. In this section we show that for problems (S,T,f) with $f:R\rightarrow R$ convex and piecewise linear the calculation of shift costs can be performed more efficiently by updating and storing these costs instead of recalculating them each iteration.

We first examine two consecutive shifts of the same block. Given an assignment μ for problem (S,T,f), the costs of two consecutive shifts of μ with respect to maximal block $[t_i, t_j]$ are:

(3.12) $$\Delta_1 = \sum_{x=i,j} \Delta_{[x,1]}, \text{ and}$$

(3.13) $$\Delta_2 = \sum_{x=i,j} \Delta_{[x,2]}.$$

Consider destinations t_k and t_{k+1} in $[t_i, t_j]$ with $\mu(t_k) \equiv s_q$. Note that the first shift reassigns t_k from s_q to s_{q-1} and the second shift reassigns t_{k+1} between the same sources (from s_q to s_{q-1}). In general, $\Delta_{[p,2]} - \Delta_{[p-1,1]}$ for $i < p \leq j$ is the difference between the changes in cost of destinations t_p and t_{p-1} over the interval $[s_{k-1}, s_k \equiv \mu(t_{p-1})]$. Letting $\delta = \sum_{x=i+1,j}(\Delta_{[x,2]} - \Delta_{[x-1,1]})$ represent the sum of these differences, Δ_2 can be calculated in terms of Δ_1 and δ as:

$$\Delta_2 = \Delta_1 + \Delta_{[i,2]} - \Delta_{[j,1]} + \delta. \tag{3.14}$$

Since the $\Delta_{[i,2]}$ and $\Delta_{[j,1]}$ terms can be calculated in O(1) steps, if we know Δ_1 and δ, we can calculate Δ_2 in O(1) steps.

Now consider the processing required to update the cost of a shift. Suppose, just before iteration i of STEP 2, we know the cost of a shift with respect to the maximal block $[t, t_{i-1}]$. If t_i becomes a member of the block $[t, t_{i-1}]$, then the shift cost with respect to $[t, t_i]$ can be calculated in O(1) steps by adding the change in cost of t_i due to a shift to the cost of a shift with respect to $[t, t_{i-1}]$. Then, if a shift is performed, equation (3.14) can be used to update the shift cost of this block in O(1) steps. Otherwise, when $[t_i, t_i]$ is a maximal block at the end of iteration i no shift is performed. Since this block contains only one member, the cost of a left shift with respect to $[t_i, t_i]$ can be calculated in O(1) steps and stored for the next iteration. Therefore,

if we know the δ terms, using this approach to update the cost of a left shift requires O(1) steps in each iteration of STEP 2. Thus, the $|T|$ iterations of STEP 2 can be accomplished in $O(|T|)$ steps plus the amount of time required to calculate and store the δ terms for each shift, which we now consider.

Let $\delta_{p,q}$ for $1 < p \leq |T|$ and $1 < q \leq |S|$ denote the difference between the changes in cost of destinations t_p and t_{p-1} over an interval (s_{q-1}, s_q). More precisely,

$$\delta_{p,q} = (f(t_p - s_{q-1}) - f(t_p - s_q)) - (f(t_{p-1} - s_{q-1}) - f(t_{p-1} - s_q)). \quad (3.15)$$

Therefore, for a given assignment μ, the δ term required by equation (3.14) to update the cost of a shift with respect to a block $[t_i, t_j]$ with $\mu(t_i) \equiv s_k$ is:

$$\delta = \sum_{x=1,(j-i)} \delta_{(i+x),(k+x-1)}.$$

Note that we only need to sum the non-zero $\delta_{p,q}$ terms for a given shift. Therefore, if the number of non-zero $\delta_{p,q}$ terms and the work required to calculate and store them for specific shifts is less than $O(|T|^2)$, then this is a more efficient way to calculate shift costs than the method presented in Section 3.4.

We now present notation needed to define necessary conditions for non-zero $\delta_{p,q}$ values.

Given a problem (S, T, f) with $f: R \rightarrow R$ convex and piecewise linear, let $B = \{b_i: i = 1, .., |B|\}$ be the finite set of

breakpoint values where the slope of f changes. We index these breakpoints so that i < j implies that $b_i < b_j$. Let $b_{[i,j]}$ denote the position on the line where $b_i = b_{[i,j]} - t_j$. Note that for a given destination t_j, $c_L(s, t_j)$ is a function of the value of s. We denote this function by $c_L(*, t_j)$ and refer to it as the **cost function of** t_j. Therefore, $b_{[i,j]}$ denotes a breakpoint for the cost function $c_L(*, t_j)$.

Now consider the term $\delta_{p,q}$. Since the function f is convex, it is also continuous. Therefore, if the slopes of the cost functions for t_p and t_{p-1} are equal at every point in (s_{q-1}, s_q) where a slope is defined, then $(c_L(s_{q-1}, t_p) - c_L(s_q, t_p)) = (c_L(s_{q-1}, t_{p-1}) - c_L(s_q, t_{p-1}))$, and hence, $\delta_{p,q} = 0$. Therefore, for $\delta_{p,q}$ to be non-zero the slopes of $c_L(*, t_p)$ and $c_L(*, t_{p-1})$ must be unequal at some point on the interval (s_{q-1}, s_q). Using this observation, Lemma 3.18 defines necessary conditions for non-zero $\delta_{p,q}$ terms.

Lemma 3.18 Consider the problem (S,T,f) with $f:R \to R$ convex and piecewise linear. $\delta_{p,q} \neq 0$, for $1 < p \leq |T|$ and $1 < q \leq |S|$, only if $t_p \neq t_{p-1}$ and one of the following two conditions are satisfied:

(1) there is a value $b_{[j,p-1]}$ or $b_{[k,p]}$ on the interval (s_{q-1}, s_q),
(2) there are values $b_{[z,p-1]}$ and $b_{[z,p]}$ with $b_{[z,p-1]} \leq s_{q-1} \leq s_q \leq b_{[z,p]}$.

Proof: If $t_p = t_{p-1}$, then $c_L(s, t_p) = c_L(s, t_{p-1})$ for each $s \in S$, and, by equation (3.15), $\delta_{p,q} = 0$ for $1 < q \leq |S|$. Therefore, whenever $\delta_{p,q} \neq 0$, $t_p \neq t_{p-1}$.

Now assume $t_p \neq t_{p-1}$ and neither condition (1) or (2) are satisfied. When condition (1) is not satisfied, the slopes of the functions $c_L(*, t_p)$ and $c_L(*, t_{p-1})$ are constant on the interval (s_{q-1}, s_q). When condition (2) is not satisfied, either $b_{[y,p-1]} \leq b_{[y,p]} \leq s_{q-1}$ or $s_q \leq b_{[y,p-1]} \leq b_{[y,p]}$ for each $y \in \{1, 2, \ldots, |B|\}$. Therefore, the slopes of $c_L(*, t_p)$ and $c_L(*, t_p)$ are defined and equal at every point of the interval (s_{q-1}, s_q). Therefore, since the function f is continuous, $c_L(s_{q-1}, t_p) - c_L(s_q, t_p) = c_L(s_{q-1}, t_{p-1}) - c_L(s_q, t_{p-1})$ and $\delta_{p,q} = 0$. This implies that $\delta_{p,q} \neq 0$ only when $t_p \neq t_{p-1}$ and at least one of conditions (1) and (2) are satisfied. □

Using Lemma 3.18, we now show that the number of non-zero $\delta_{p,q}$ terms cannot exceed $O(|S||B|)$.

Lemma 3.19 Consider the problem (S, T, f) with $f: R \rightarrow R$ convex and piecewise linear with $|B|$ breakpoints. There are $O(|S||B|)$ non-zero $\delta_{p,q}$ values for this problem.

Proof: By Lemma 3.18, a $\delta_{p,q}$ term can only be non-zero if conditions (1) or (2) of that Lemma are satisfied. Given a destination t_p, condition (1) is satisfied for each interval (s_{q-1}, s_q) which contains either a $b_{[i,p]}$ or $b_{[j,p-1]}$ value. There

are at most 2|B| of these intervals for the given destination. Therefore, considering all destinations, since |T| ≤ |S|, condition (1) is satisfied in O(|S||B|) intervals.

Now consider condition (2) of Lemma 3.18. Consider the set $\{b_{[i,j]}: \ j = 1,\ldots,|T|\}$ for $1 \le i \le |B|$. Since $b_{[i,j]} \le b_{[i,j+1]}$ for $j = 1,\ldots,|T|-1$, there are at most $(|S|-1)$ intervals $[s_{q-1}, s_q]$ with

$$b_{[i,j]} \le s_{q-1} \le s_q \le b_{[i,j+1]}$$

for at least one $j \in \{1,\ldots,|T|-1\}$. Since this is true for each i, there are at most $|B|(|S|-1)$ intervals (s_{q-1}, s_q) which satisfy condition (2) of Lemma 3.18.

Therefore, by Lemma 3.18, there are O(|B||S|) non-zero $\delta_{p,q}$ terms to calculate and store for the problem (S,T,f). □

We now consider the processing required to find, calculate, and store the non-zero $\delta_{p,q}$ values. We take the approach of calculating non-zero $\delta_{i,q}$ terms during iteration i of STEP 2, which requires the accomplishment of three tasks:

(1) identify the intervals (s_{q-1}, s_q) which satisfy the conditions of Lemma 3.18 for destination t_i.

(2) calculate the $\delta_{i,q}$ values for these intervals, and

(3) store these values.

To determine the intervals (s_{q-1}, s_q) that satisfy the conditions of Lemma 3.18 for destination t_i, we must know the locations of the breakpoints of $c_L(*, t_i)$ and $c_L(*, t_{i-1})$ with respect to the sources in S. This can be accomplished prior to STEP 2 by sorting a list containing the $b_{[i,k]}$ values of every $t_k \in T$, then comparing each of these values with the elements of a sorted list of source locations, generating a link list, called the bp_k-list for each destination t_k, with records ordered by breakpoint index. Record j of this list identifies either the interval (s_q, s_{q+1}) or source location that contains breakpoint $b_{[j,k]}$. These operations take $O(|T||B|\log|T||B|) + O(|S| + |T||B|) = O(|T||B|\log|T||B| + |S|)$ steps.

In iteration i of STEP 2, we use the bp_i-list and bp_{i-1}-list to identify those intervals which satisfy the conditions of Lemma 3.18. These intervals are identified by stepping through the bp_i-list and the bp_{i-1}-list, starting with the last entry in each list and noting those intervals that either contain the breakpoints or are between breakpoints $b_{[i-1,k]}$ and $b_{[i,k]}$. In addition to the time required to calculate the non-zero $\delta_{i,q}$ values, this requires $O(|B|)$ steps in each iteration of STEP 2.

Given destinations $t_i \neq t_{i-1}$ and an interval (s_{q-1}, s_q), the $\delta_{i,q}$ value for this interval can be determined using equation (3.15) in a constant number of computational steps.

Consider the storage of non-negative $\delta_{i,q}$ values after any required shift in iteration i > 1 of STEP 2. Let μ designate the partial assignment at the end of iteration i and assume $\mu(t_{i-1}) \equiv s_j$ and $\mu(t_i) \equiv s_k$. Assume that at the end of iteration i-1 of STEP 2 we had a list, called the δ-list, where the k^{th} entry in this list is the sum of the $\delta_{p,q}$ terms needed to update the shift cost after the k^{th} left shift of the maximal block $[\ell, t_{i-1}]$. We use δ_k to denote the k^{th} entry of this list. We now consider the cases in which j = k-1 and j < k-1.

When j = k-1, t_i is in the same block as t_{i-1} and in all subsequent shifts with respect to a maximal block containing t_i the $\delta_{i,q}$ terms for intervals (s_{q-1}, s_q) with $s_q \leq s_j$ must be added to the entries in the δ-list. Specifically, non-zero $\delta_{i,q}$ terms for $q \leq j$ are added to δ_{j-q+1}.

When j < k-1, t_i is not in the maximal block of μ containing t_{i-1}. It would take (k-j) left shifts with respect to the maximal block containing t_i before t_i and t_{i-1} are in the same maximal block. Any additional left shifts of the maximal block containing t_i would require δ_k terms that reflect the differences in assignment cost between t_{i-1} and t_i. Therefore, to update the δ-list, we add (j-k) records with zero values to the beginning of the δ-link list. Then, the $(j-k+1)^{st}$ entry of the δ-list and every entry thereafter is updated to include the non-zero $\delta_{i,q}$ terms for those shifts.

Therefore, the time required to store the non-zero $\delta_{p,q}$ terms (in addition to the time to calculate them) is a function of the number of these terms and the number of "zero-records" added to the front of the δ-list throughout all iterations of STEP 2. We have already noted that the number of non-zero $\delta_{p,q}$ terms equals $O(|S||B|)$, and now consider the number of zero-records.

Zero-records are only added to the front of the δ-list in an iteration i in which a new maximal block $[t_i, t_i]$ is created. In the current partial assignment μ, each of these zero-records corresponds to an interval (s_{q-1}, s_q) with $\mu(t_{i-1}) \preceq s_{q-1} \prec s_q \preceq \mu(t_i)$. Now consider a subsequent iteration j of STEP 2 and let μ' and μ'', respectively, denote the partial assignments just prior to and after iteration j. Since Left-Shift generates a non-crossing assignment and at most one left shift is performed in each iteration, $\mu'(t_{j-1}) \preceq \mu''(t_j)$. In particular, this observation holds for j = i+1, and therefore, $\mu(t_i)$ is matched for all iterations j > i of STEP 2. Thus, the intervals that corresponded to zero-records in iteration i will not correspond to zero-records in any subsequent iteration and there can be at most $|S|-1$ zero-records added to the beginning of the δ-list.

Therefore, we require $O(|S||B|)$ steps to store the non-zero $\delta_{p,q}$ terms.

In summary, in order to calculate the shift costs in STEP 2, we first require $O(|T||B|\log|T||B| + |S|)$ steps to define the bp_k-list for each destination t_k, followed by $O(|S||B|)$ steps to calculate the appropriate non-zero $\delta_{p,q}$ terms and add them to the δ-list. Given the values of the δ-list, it takes a total of $O(1)$ steps to update the cost of a shift in each iteration of STEP 2. Therefore, it takes $O(|T||B|\log|T||B| + |S| + |S||B| + |T|) = O(|T||B|\log|T||B| + |S||B|)$ steps to determine the cost of all shifts performed by Left-Shift.

Since we also require $O(|S|\log|S|)$ steps to sort the source and destination locations, when the function f is piecewise linear and convex with $|B|$ breakpoints, Left-Shift can be performed in $O(|S|\log|S| + |T||B|\log|T||B| + |S||B|)$ computational steps. If $|B|$ can be bounded by a constant, then Left-Shift is an $O(|S|\log|S|)$ procedure.

CHAPTER 4

The Assignment Problem on the Circle

4.1 Introduction

In this chapter we consider an assignment problem, known as the **assignment problem on the circle**, in which sources and destinations are located on a circle and costs are a function of clockwise and counterclockwise displacement. We denote this problem by (S,T,f,H) for source set S and destination set T, with assignment costs based on a function f of displacement around a circle of circumference H.

As discussed in Section 3.2, Karp and Li (1975) investigated this problem in their application of relocating desks around a circular corridor. In this application, desks were moved from one location to another with cost equal to the distance moved. On a circular corridor, the cost of moving a desk from one location to another is the minimum of the clockwise and counterclockwise distances between these locations.

In addition to modeling spatial differences, a circle can be used to model temporal differences between events that repeat over time. For instance, consider Figure 4.1, which depicts two events A and B that occur on Monday and Wednesday

every week. In this case, event B occurs 2 days after the last occurrence of event A, and event A occurs 5 days after the last occurrence of event B. Note that on a circle these differences coincide with the clockwise and counterclockwise distances between these events.

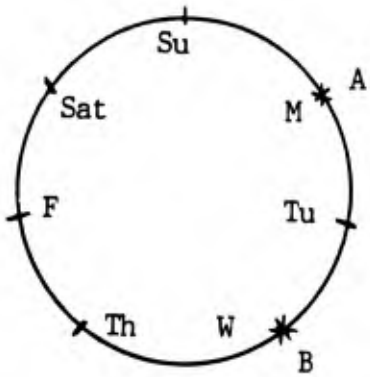

Figure 4.1 Circular Model of Repeating Events

In our application to mission scheduling, we consider customer services and mission stops that repeat each H days. We alter mission schedules, expediting or delaying mission stops, so that each mission stop coincides with a desired customer service time. If these events are represented on the circle, then expediting a mission stop moves this event backward in time and corresponds to a counterclockwise displacement on the circle. Likewise, delaying a mission stop moves this event forward in time, corresponding to a clockwise displacement on the circle. For example, consider a mission due to arrive on day four of a seven day cycle for a customer service on day two. This mission stop and service are

represented by events B and A of Figure 4.1 where Sunday is day 1 of the planning cycle. Either this stop could be expedited to occur 2 days earlier or the previous mission stop could be delayed 5 days to perform this service. Since a mission stop can be expedited or delayed to coincide with a customer service, the cost of altering a mission to perform a customer service is the minimum of the costs to expedite or delay that mission.

We say a schedule that repeats each H days has a cycle length of H days. The length H of the cycle influences both the events that are modeled and the quality of the solution. Since the schedule developed for a single cycle is repeated each H days, an event occurring less often than every H days cannot be accurately modeled. For instance, if an event occurs quarterly and we consider a cycle of only 30 days, then we must choose between scheduling this event every 30 days or not scheduling it at all.

The length of the cycle also influences the quality of the solution. In general, the longer the cycle, the more solution options that can be considered. For example, every solution for a two week cycle can be repeated, yielding solutions for a four week cycle. This implies that an optimal solution for the four week cycle cannot be worse than a solution for a two week cycle.

Therefore, increasing the length of a cycle increases both the number of events that are accurately modeled and the expected solution quality. In general then, the cycle length should be as long as is practical for a given application. For example, in our application, AMC conducts its advance planning using a cycle length of 30 days to coincide with their current planning requirements. They develop a standard monthly schedule based on forecast demands and every 30 days modify this schedule manually to accommodate predicted changes in cargo movement for the next month. These modifications are repeated every 30 days because customer forecasts are considered unreliable more than a month in advance. Having a standard schedule to alter each month makes this task easier. Other than this, there is no operational reason why advance planning could not consider a cycle length longer than 30 days. On the other hand, choosing a shorter cycle length would not be desirable since advance planning could not accurately model those customers requiring service once a month.

As in Section 3.3, we model the customer and mission events as sources and destinations with locations corresponding to their scheduled times. Since we model a single cycle and events occurring H time units apart are in different cycles, each source and destination has a value on the range [0,H). The sources and destinations are indexed so

that for each pair of sources s_i and s_j (destinations t_i and t_j) if $i < j$, then $s_i \leq s_j$ ($t_i \leq t_j$).

On a circle of circumference H the clockwise displacement from t to s, denoted $cwd(t,s)$, is defined as:

$$(4.1) \qquad cwd(t,s) = \begin{cases} s-t & \text{if } s \geq t \\ H-t+s & \text{if } s < t. \end{cases}$$

The counterclockwise displacement from t to s, denoted $ccwd(t,s)$, is $H-cwd(t,s)$. Note that $cwd(t,t) = 0$, $ccwd(t,t) = H$, and $cwd(t,s)$ and $ccwd(t,s)$ are nonnegative for all s and t.

A counterclockwise displacement from a destination represents expediting a mission stop, and a clockwise displacement represents delaying a mission stop. In Section 3.3, we used a function $f:R \rightarrow R$ to calculate assignment costs with negative arguments corresponding to expediting a mission and positive arguments corresponding to delaying a mission. Analogously, we calculate the cost of expediting a mission by x time units as $f(-x)$. Since we can choose between expediting or delaying a mission to perform a given service, the cost of serving a customer at time s by a mission arriving at time t is the minimum of the costs to expedite and delay that mission. Therefore, on a circle of circumference H, the cost $c_c(s,t)$ of matching destination t with source s is defined as:

(4.2) $c_c(s,t) = \min \{f(\text{cwd}(t,s)),\ f(-\text{ccwd}(t,s))\}$.

We denote the cost function of a destination t by $c_c(*,t)$ and the cost matrix of problem (S,T,f,H) by $C_c(S,T,f,H)$.

Since the assignment problem on the circle is similar to the assignment problem on the line, we would like to use the results of Chapter 3 to solve this problem. Unfortunately, the form of the cost function prevents a straightforward application of these results. For instance, consider the function $f:R\to R^+$ defined by $f(x) = |x|$. Figure 4.2 shows that although f is convex, $c_c(*,t)$ is not.

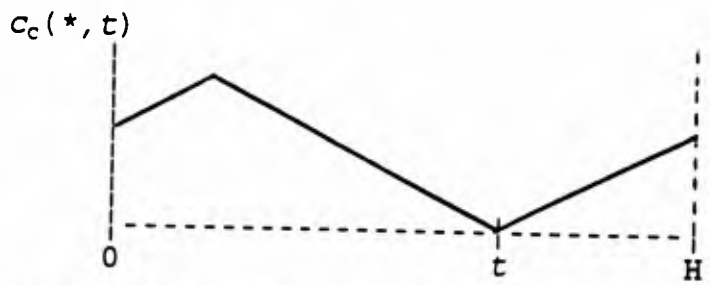

Figure 4.2: $c_c(*,t)$ for $f(x) = |x|$.

Nevertheless, the problem has sufficient structure in common with the problem on the line that we can apply many of the results developed in that context, though with some modifications.

We now present notation needed for our analysis of the assignment problem on the circle.

Given locations a and b on a circle, the arc $[a,b]$ is the closed arc of the circle that starts at location a and contains all points clockwise from a to b. An arc may be closed, denoted $[a,b]$; open, denoted (a,b); or half open, denoted $(a,b]$ or $[a,b)$. The arcs $[a,a)$ and $(a,a]$ contain every point of the circle, starting at point a or ending at point a, respectively.

We use "$<$, $\leq$, $>$, $\geq$" to compare the relative positions of points with respect to a given point. For example, we say $c \leq d$ with respect to reference point a if $\text{cwd}(a,c) \leq \text{cwd}(a,d)$, i.e., if we pass c when moving clockwise from a to d. If the reference point is not specified, then it is understood to be the point 0.

As stated earlier, the sources and destinations are indexed based on their location with respect to point 0. We use "$\prec$, $\preceq$, $\equiv$, $\succ$, $\succeq$" to compare the order, based on both position and index, of sources or destinations on an arc. Consider two sources s_i and s_j on arc $[a,b]$. We say $s_i \equiv s_j$ only if $i = j$. We say $s_i \prec s_j$ on arc $[a,b]$ when either $s_i < s_j$ with respect to a, or $s_i = s_j$ with $i < j$. Finally, $s_i \preceq s_j$ on arc $[a,b]$ when $s_i \equiv s_j$ or $s_i \prec s_j$ on arc $[a,b]$. We use the same notation to compare destinations.

We say that destinations t_i and t_j (sources s_i and s_j) are **distinct** when $i \neq j$ and are at **different locations** when $t_i \neq t_j$ ($s_i \neq s_j$).

We now examine assignment costs on the circle.

4.2 Convex Costs on a Circle

In this section we examine properties of convex functions of displacement on the circle and develop non-crossing results similar to those in Chapter 3. We first note that not all convex functions are suitable for our application and restrict the convex functions that we consider.

A large enough displacement on the circle brings us back to our starting point. In the application of moving desks around a circular corridor, a displacement of this amount would not change the arrangement of the desks. Therefore, in modeling these applications on a circle it is reasonable to consider only those displacements which are less than the circumference, and, in fact, this assumption is implicit in our definition of assignment costs in equation (4.2). Not every convex function is compatible with this assumption. In particular, a cost function would not be suitable for our application if the cost of moving a desk around a corridor and returning it to its original location is no worse than leaving it in its original position.

In addition, when moving desks or altering missions, some changes should be better than others. This is not the case if costs are constant for displacements on the circle. In this case, every possible move or change requires the same energy

or incurs the same cost.

Therefore, in the remainder of this chapter we only consider problems (S,T,f,H) with f convex and consistent, which we now define. We say $f:R\to R$ is **consistent** when the cost functions $c_c(*,t)$ are not constant, and $f(0) < \min\{f(H), f(-H)\}$, i.e., not moving is less expensive than going all the way around the circle in either direction.

Restricting attention to consistent functions of displacement does not limit our ability to solve the more general problem (S,T,f,H) with $f:R\to R$ convex and displacements restricted to be less than H. Given a problem (S,T,f,H), if each function $c_c(*,t)$ is constant, then all assignments have equal value and the solution $\mu(t_i) \equiv s_i$ for $i = 1,\ldots,|T|$, for example, is an optimal solution. If $f(0) \geq \min\{f(-H), f(H)\}$, then the problem (S,T,$f$,H) with displacements constrained to be less than H can be transformed to an equivalent problem (S,T,f'',H) with $f'':R\to R$ convex and consistent. We now demonstrate this transformation for the case in which $f(-H) \leq f(0)$ (the transformation for the case in which $f(H) \leq f(0)$ is similar).

Let $\gamma = \max\{ccwd(t,s): s \in S, t \in T\}$ and define $f':R\to R$ by

$$f'(x) = -K^*(x+\gamma) + f(-\gamma), \tag{4.3}$$

where

(4.4) $$K > (f(0) - f(-\gamma))/(H - \gamma).$$

Define $f'':R \to R$ by

(4.5) $$f''(x) = \text{maximum } \{f(x), f'(x)\}.$$

The following lemma shows that this transformation can be used to generate a function that is convex and consistent.

Lemma 4.1 Given a convex function $f:R \to R$ with $f(-H) \leq f(0)$, the function f'', defined in (4.3)-(4.5) is convex and $f''(-H) > f''(0)$.

Proof: Note that f' is linear. Since the function f'' is the maximum of convex functions, f'' is convex. We now show that $f''(-H) > f''(0)$.

We are given that $f(-H) \leq f(0)$, and, by (4.3) and (4.4), $f'(-H) > f(0)$. Thus, $f'(-H) > f(-H)$, implying

(4.6) $$f''(-H) = f'(-H) > f(0).$$

Observe that $f(-\gamma) = f'(-\gamma)$. Noting that $f'(-H) > f(-H)$, f is convex, and f' is linear, it follows that $f(0) > f'(0)$. Therefore, $f''(0) = f(0)$ and, by (4.6), $f''(-H) > f''(0)$. □

Lemma 4.2 shows that the cost matrices for (S,T,f,H) and (S,T,f'',H) are equal. Therefore, the problem (S,T,f,H) in which we only consider displacements less than H is equivalent

to (S,T,f'',H).

Lemma 4.2 Consider a problem (S,T,f,H) with $f:R\to R$ convex and $f(-H) \le f(0)$. Cost matrix $C_c(S,T,f,H) = C_c(S,T,f'',H)$ where f'' is the function defined by equations (4.3)-(4.5).

Proof: First we show that $f''(x) = f(x)$ for $x \ge -\gamma$. Note that $f(-\gamma) = f'(-\gamma)$, so consider $x > -\gamma$. Since $-H < -\gamma < x$ we can choose $\lambda \in (0,1)$ so that $\lambda(-H) + (1-\lambda)x = -\gamma$. Since f is convex and f' is linear, $\lambda f(-H) + (1-\lambda)f(x) \ge f(\lambda(-H) + (1-\lambda)x) = f(-\gamma) = f'(-\gamma) = \lambda f'(-H) + (1-\lambda)f'(x)$. Since $f(-H) < f'(-H)$, it follows that $f(x) > f'(x)$ and $f''(x) = f(x)$.

We now show that $\min\{f(\mathrm{cwd}(t,s)),\ f(-\mathrm{ccwd}(t,s))\} = \min\{f''(\mathrm{cwd}(t,s)),\ f''(-\mathrm{ccwd}(t,s))\}$ for each $s\in S$ and $t\in T$. Since $\mathrm{cwd}(t,s) \ge 0 \ge -\gamma$, $f(\mathrm{cwd}(t,s)) = f''(\mathrm{cwd}(t,s))$. Also, since $\gamma = \max\{\mathrm{ccwd}(t,s):\ s\in S \text{ and } t\in T\}$, $-\mathrm{ccwd}(t,s) \ge -\gamma$ and, $f(-\mathrm{ccwd}(t,s)) = f''(-\mathrm{ccwd}(t,s))$. Therefore, the cost matrices $C_c(S,T,f,H)$ and $C_c(S,T,f'',H)$ are identical. □

Considering only convex, consistent functions of displacement, we now demonstrate those properties of assignment costs that are useful in defining noncrossing assignments on the circle. In Figure 4.2 we demonstrated a case in which the displacement cost function $c_c(*,t)$ was not convex. This function was locally concave at a point x where $f(-\mathrm{ccwd}(t,x)) = f(\mathrm{cwd}(t,x))$. Since the counterclockwise and

clockwise costs are equal at this location, we refer to this point as the **equality point** for destination t, and denote it by $E(t)$. This location is important because it can be used to unambiguously define the cost of assigning a destination t to a source s, i.e. to determine whether $c_c(s,t) = f(-ccwd(t,s))$ or $c_c(s,t) = f(cwd(t,s))$. For instance, as we prove in Lemma 4.3 below, if a point x is on the arc $[E(t),t)$, then $f(-ccwd(t,x)) \leq f(cwd(t,x)$ and the cost of assigning destination t to x is $f(-ccwd(t,x))$.

Lemma 4.3 Consider a problem (S,T,f,H) with $f:R\rightarrow R$ convex and consistent. For each destination t, there is a point $E(t) \neq t$ in $[0,H)$ with $f(-ccwd(t,E(t))) = f(cwd(t,E(t)))$ and

$$c_c(s,t) = \begin{cases} f(-ccwd(t,s)) \text{ if } s \text{ is on arc } [E(t),t), \\ f(cwd(t,s) \text{ if } s \text{ is on arc } [t,E(t)]. \end{cases} \tag{4.7}$$

Proof: We first show that there is at least one value $E(t) \neq t$ with $f(-ccwd(t,E(t)) = f(cwd(t,E(t))$. Consider a point $s \neq t$ in $[0,H)$. If $f(cwd(t,s)) = f(-ccwd(t,s))$, then let $E(t) = s$. Otherwise, assume $f(cwd(t,s)) > f(-ccwd(t,s))$ (the argument is similar when $f(cwd(t,s)) < f(-ccwd(t,s))$). Consider the functional values $f(cwd(t,x))$ and $f(-ccwd(t,x))$ for x on the arc $[s,t]$. Note that $f(cwd(t,s) > f(-ccwd(t,s))$, and $f(cwd(t,t)) = f(0) < f(-H) = f(-ccwd(t,t))$. Since f is convex on R, it is continuous on (s,t) and there is a location

x' on the arc (s,t) with $f(\text{cwd}(t,x'))) = f(\text{ccwd}(t,x')))$.

Now consider equation (4.7). Note that $-\text{ccwd}(t,E(t)) < \text{cwd}(t,E(t))$, f is convex and $f(\text{cwd}(t,E(t))) = f(-\text{ccwd}(t,E(t)))$. Therefore, if $z' < -\text{ccwd}(t,E(t))$ or $z' > \text{cwd}(t,E(t))$, then $f(z') \geq f(z)$ for $z \in [-\text{ccwd}(t,E(t)), \text{cwd}(t,E(t))]$. If a location s is on the arc $(E(t),t]$, then $\text{cwd}(t,s) > \text{cwd}(t,E(t))$ and $f(\text{cwd}(t,s)) \geq f(-\text{ccwd}(t,s))$. Likewise, if s is on the arc $[t,E(t))$, then $\text{ccwd}(t,s) > \text{ccwd}(t,E(t))$ and $f(-\text{ccwd}(t,s)) \geq f(\text{cwd}(t,s))$. Therefore, (4.7) holds. □

We base our definition of noncrossing assignments on the locations of these equality points. Lemma 4.4 shows that these locations are unique, and hence our definition is not ambiguous.

Lemma 4.4 Consider a problem (S,T,f,H) with $f:R\to R$ convex and consistent. For each destination t, $c_c(x,t) < c_c(E(t),t)$ for each point $x \neq E(t)$ in $[0,H)$, and the point $E(t)$ is unique.

Proof: Lemma 4.3 established that for a destination t there is at least one point $E(t)$ with $f(\text{cwd}(t,E(t))) = f(-\text{ccwd}(t,E(t)))$. Given this point, consider another point $x \in [0,H)$. If $c_c(x,t) \geq \alpha = f(-\text{ccwd}(t,E(t)) = f(\text{cwd}(t,E(t)))$, then by the convexity of f, $\alpha = f(x)$ for each $x \in [-\text{ccwd}(t,E(t)), \text{cwd}(t,E(t))]$. Equation (4.7) then implies

$c_c(*,t)$ is constant, contradicting the assumption that f is consistent. Therefore, $c_c(x,t) < c_c(E(t),t)$.

By equation (4.7), if $x \in (E(t),t)$, then $c_c(x,t) = f(-ccwd(t,x))$. Since $cwd(t,x) > cwd(t,E(t))$ and f is convex,

$$f(-ccwd(t,x)) < f(cwd(t,E(t))) < f(cwd(t,x)).$$

Likewise, if x is on the arc $[t,E(t))$, then $f(cwd(t,x)) < f(-ccwd(t,x))$. Therefore, $E(t)$ is uniquely determined for each destination t. □

Note that for a given problem (S,T,f,H) with $f:R\to R$ convex and consistent, since each cost function $c_c(*,t)$ is defined using the same function f and value H, $cwd(E(t),t) = cwd(E(t'),t')$ for each pair of destinations t and t'. We denote this constant distance for problem (S,T,f,H) by d_E.

We now develop results for the assignment problem on the circle analogous to the non-crossing results of Chapter 3. We develop the intuition behind our definition of non-crossing assignments on the circle by considering Karp and Li's application of moving desks along a circular corridor. In particular, we examine two types of inefficient movements, which occur when desks pass each other going in opposite directions or the path of one desk completely contains the path of another desk.

We move a desk to another location along the least cost path. Therefore, by Lemma 4.3, we move a desk at t counterclockwise (clockwise) to a point s on the arc $[E(t),t)$ $((t,E(t)))$. These paths are illustrated in Figure 4.3 by the lines drawn from t to s' and s.

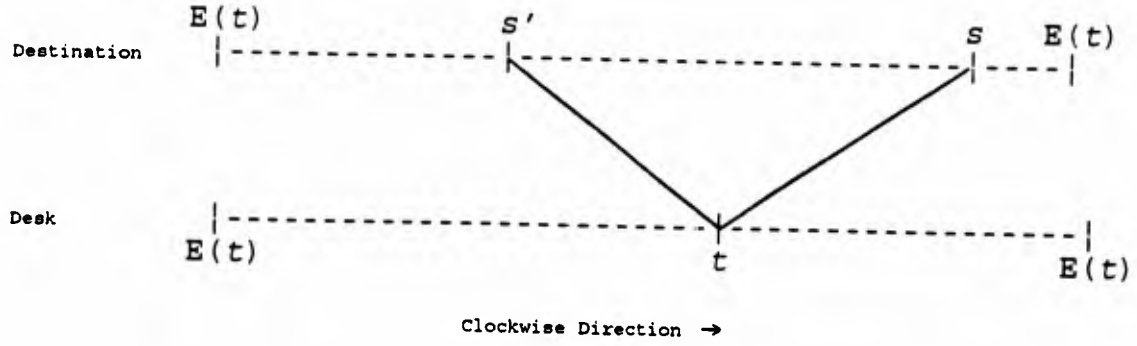

Figure 4.3 Paths taken between points on a circle.

Now consider the case in which desks pass each other going in opposite directions. This is illustrated below in Figure 4.4, which depicts two desks at locations t and t' assigned to points s and s', respectively.

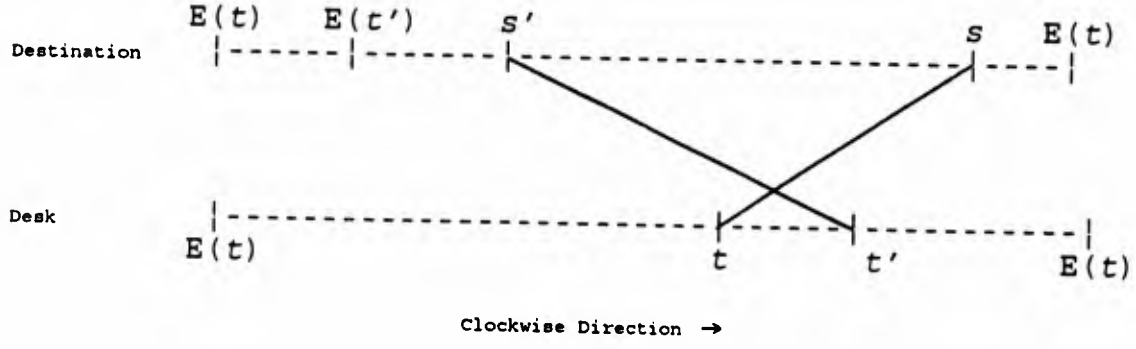

Figure 4.4 Paths that cross going in opposite directions.

The interval on which these paths overlap represents wasted effort since desk t could have been moved to s', a shorter distance than from t' to s', and desk t' could have been moved to s, a shorter distance than from t to s.

Similarly, moving two desks from different locations in the same direction wastes effort when one path completely contains the other. This case is illustrated in Figure 4.5, which illustrates the paths followed by desks located at points t and t' to their assigned locations at points s and s', respectively.

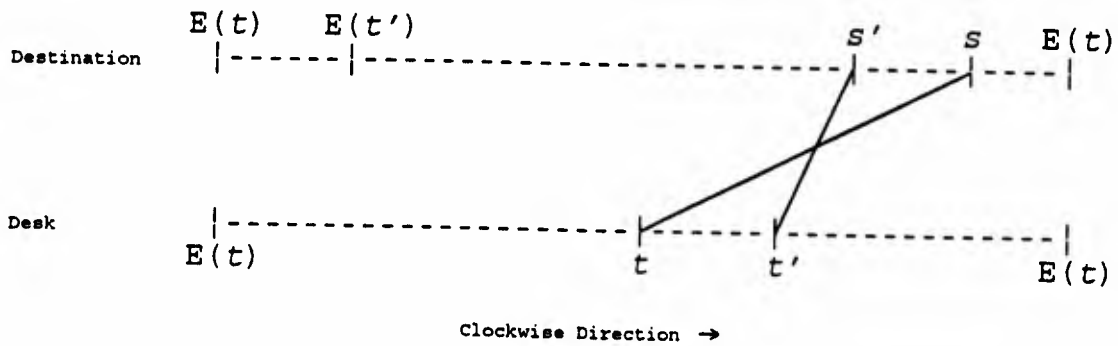

Figure 4.5 Paths that cross going in the same direction.

Since f is convex, moving costs increase at a nondecreasing rate and the additional cost from moving desk t' from s' to s cannot be higher than moving desk t on the same interval. Therefore, the savings realized by moving desk t' the relatively short distance to s' cannot make up for the higher premiums charged for moving desk t further than was necessary.

These observations also apply to our application of altering airlift missions. For instance, let t and t' represent mission stops that perform customer services at times s and s'. Figure 4.4 represents the case where we simultaneously expedite one mission and delay the other. This is not an efficient schedule since delaying mission stop t' to perform customer service s cannot cost more than delaying mission stop t to perform the same service, and expediting mission stop t to perform a customer service at time s' cannot be more expensive than expediting mission stop t' to perform this service. Similarly, Figure 4.5 represents the case where a mission, arriving at time t, sits and waits to perform its assigned customer service while another mission arrives and leaves.

Therefore, an assignment is inefficient if two paths cross going in opposite directions or if one path contains another. One of these cases occurs whenever desks at different locations t and t' are moved to locations s and s', respectively, and $s' \preceq s$ on the arc $[E(t'),E(t)]$ or $s \preceq s'$ on the arc $[E(t),E(t')]$. For example, consider the case with $t \prec t'$ with respect to $E(t)$ and $s \preceq s'$ on the arc $[E(t),E(t')]$ as shown in Figures 4.6 and 4.7 below.

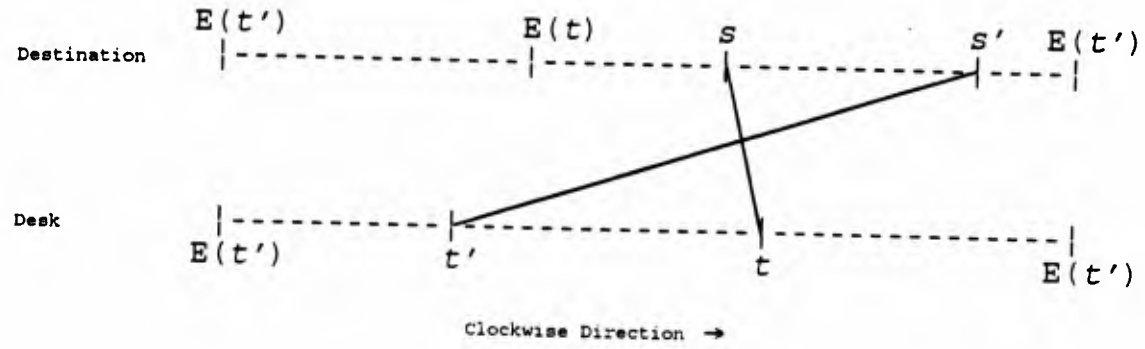

Figure 4.6 Crossing paths with s on the arc $[E(t), t)$.

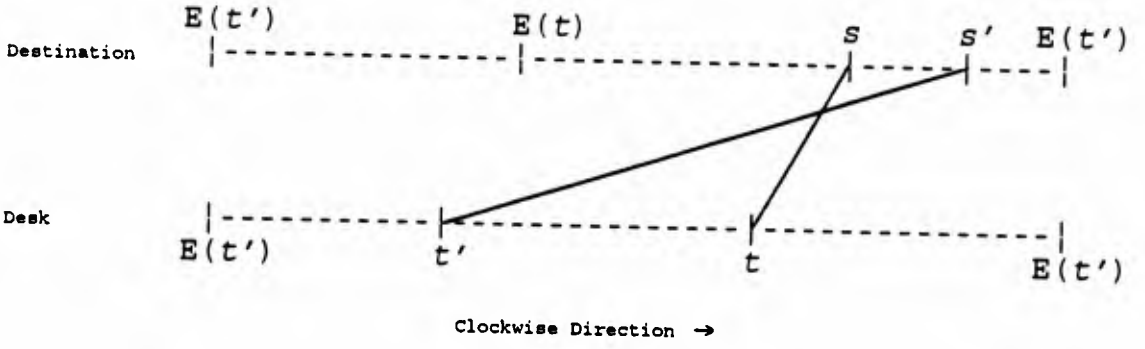

Figure 4.7 Crossing paths with s on the arc $[t, E(t))$.

Note that, if s is on the arc $[E(t), t)$, as in Figure 4.6, then the paths from t to s and t' to s' cross going in opposite directions. Otherwise, if s is on the arc $[t, E(t')]$, as in Figure 4.7, then the path from t' to s' contains the path from t to s. From these observations, we have the following definition of crossing assignments on the circle.

Let μ be an assignment for problem (S, T, f, H) with $f: R \to R$ convex and consistent. The pairs $(\mu(t), t)$ and $(\mu(t'), t')$ **cross** if t is at a different location than t' and either $\mu(t) \prec \mu(t')$ on arc $[E(t), E(t')]$ or $\mu(t') \prec \mu(t)$ on arc $[E(t'),$

$E(t)]$. Note that, by this definition, if $\mu(t) = E(t)$, then $(\mu(t), t)$ crosses each pair $(\mu(t'), t')$ with $t' \neq t$.

If (s,t) and (s',t') cross, then **uncrossing** these pairs results in a new assignment μ' with $s \equiv \mu'(t')$, $s' \equiv \mu'(t)$, and $\mu'(t") \equiv \mu(t")$ for each $t" \in T\backslash\{t,t'\}$. Note that either $\mu'(t) \prec \mu(t')$ on arc $[E(t'),E(t)]$ or $\mu(t') \prec \mu(t)$ on arc $[E(t),E(t')]$ and the pairs $(\mu'(t),t)$ and $(\mu'(t'),t')$ do not cross.

We say that an assignment μ for the problem (S,T,f,H) with $f:R\rightarrow R$ convex and consistent is **non-crossing** if no matched pairs of μ cross. In order to avoid needless detail when discussing non-crossing assignments, throughout this chapter we assume that in a non-crossing assignment μ the mates of collocated sources and destinations follow their indexed order. Therefore, if destinations t_i and t_j have the same location and $i < j$, then $\mu(t_i) \preceq \mu(t_j)$ with respect to $E(t_i)$. Likewise, if sources s_i and s_j have the same location, $i < j$, $\mu(s_i) \equiv t_p$, and $\mu(s_j) \equiv t_q$, then $p < q$. Finally, if $[s_i, s_{i+k}]$ is a set of collocated sources matched with $j < k+1$ destinations, then the sources in the set $[s_{i+j}, s_{i+k}]$ are unmatched.

To show that there is a non-crossing optimal assignment for problems (S,T,f,H) with $f:R\rightarrow R$ convex and consistent, Lemma 4.5 formalizes the preceding discussion by showing that uncrossing a crossing pair does not increase costs. This is then followed by Lemma 4.6, which shows that uncrossing a

crossing pair decreases the number of crossing pairs of an assignment.

Lemma 4.5 Consider an assignment μ for the problem (S,T,f,H) with $f:R\rightarrow R$ convex and consistent. If $(\mu(t),t)$ and $(\mu(t'),t')$ cross, then the assignment μ' obtained from uncrossing these pairs has $\text{Cost}(\mu') \leq \text{Cost}(\mu)$.

Proof: Let $s \equiv \mu(t)$ and $s' \equiv \mu(t')$. Without loss of generality assume $s \prec s'$ on arc $[E(t),E(t')]$. Since (s,t) and (s',t') cross, $t \neq t'$ and t' is either on arc $(t,E(t))$ or arc $[E(t),t)$.

Suppose t' is on arc $(t,E(t))$ (see Figure 4.8) and assume, for notational simplicity, that $t' = 0$.

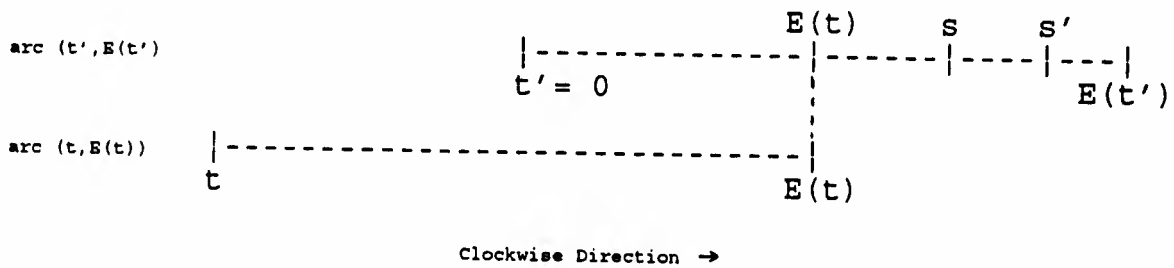

Figure 4.8 $t'= 0$ on arc $(t,E(t))$.

First, we consider the costs of matching destinations t and t' with sources s and s'. Since the assignment costs for t and t' are defined by the same function f, $\text{cwd}(t,E(t)) = \text{cwd}(t',E(t')) = d_E$. Therefore, $E(t)$ is on arc $[t',E(t'))$. Since arc $[E(t),E(t')]$ is contained in arc $[t',E(t')]$, both s

and s' are on the arc $[t',E(t')]$. Then, by equation (4.7), $c_c(s,t') = f(\text{cwd}(t,s))$ and $c_c(s',t') = f(\text{cwd}(t',s'))$. Since $t' = 0$, $t' \le s$ and $t' \le s'$. By equation (4.1),

$$c_c(s',t') = f(s'-t'), \text{ and}$$
$$c_c(s,t') = f(s-t').$$

Now consider the costs $c_c(s,t)$ and $c_c(s',t)$. Since $t' = 0$, $E(t) \le s \le s'$ and $E(t) \le t$. Therefore, if $s < t$, then s is on arc $[E(t),t)$. It then follows by (4.7) and (4.1) that $c_c(s,t) = f(-\text{ccwd}(t,s)) = f(s-t)$. Otherwise, if $t \le s$, then s is on arc $[t,E(t))$. By (4.7) and (4.1), $c_c(s,t) = f(\text{cwd}(t,s)) = f(s-t)$. Likewise, $c_c(s',t) = f(s'-t)$. Therefore, we have shown

$$c_c(s,t) = f(s-t), \text{ and}$$
$$c_c(s',t) = f(s'-t).$$

Since f is convex, we now prove our assertion using Lemma 3.2. Let $a = s'-t'$, $b = s-t$ and $\Delta = s'-s$. Since $s \le s'$ and $t' < t$, we have $b \le a$ and $\Delta \ge 0$. Also, $a-b = \Delta + (t-t')$. Since both Δ and $(t-t')$ are nonnegative, $0 \le \Delta \le a-b$. So, by Lemma 3.2, $f(a-\Delta) + f(b+\Delta) \le f(a) + f(b)$, implying $c_c(s',t) + c_c(s,t') \le c_c(s,t) + c_c(s',t')$. Therefore, $\text{Cost}(\mu') \le \text{Cost}(\mu)$.

A similar argument holds when t' is on arc $[E(t),t)$ by letting $E(t)=0$, $a = s'-t'$, $b = s-t$, and $\Delta = (s'-s)$. □

Lemma 4.6 Consider an assignment μ for problem (S,T,f,H) with $f:R\rightarrow R$ convex and consistent. Uncrossing a crossing pair of μ decreases the number of crossing pairs.

Proof: If $|S| = 2$, this result is immediate. Otherwise, let (s,t) and (s',t') be crossing pairs of μ. As noted previously, uncrossing (s,t) and (s',t') results in pairs (s',t) and (s,t') which do not cross. Assume without loss of generality that $s \prec s'$ on arc $[E(t),E(t')]$. Now consider another pair, (s'',t'') with $s'' = \mu(t'')$.

If (s'',t'') crosses both (s,t) and (s',t'), then the number of pairs that cross (s'',t'') cannot be increased by uncrossing (s,t) and (s',t'). Therefore, we must consider instances in which (s'',t'') crosses exactly one of (s,t) and (s',t'), or does not cross either (s,t) or (s',t').

Assume (s'',t'') crosses (s,t) but not (s',t') (the proof when (s'',t'') crosses (s',t') but not (s,t) is similar). Recall that we have assumed $s \prec s'$ on arc $[E(t),E(t')]$. Then one of the following cases holds:

Case I: $s'' \prec s$ on arc $[E(t),s]$,

Case II: $s \prec s'' \prec s'$ on arc $[s,s']$,

Case III: $s' \prec s''$ on arc $[s',E(t))$.

In Case I (Figure 4.9), since (s'',t'') does not cross (s',t'), $E(t'')$ is not on arc $(E(t'),s'']$, which is shaded in Figure 4.9.

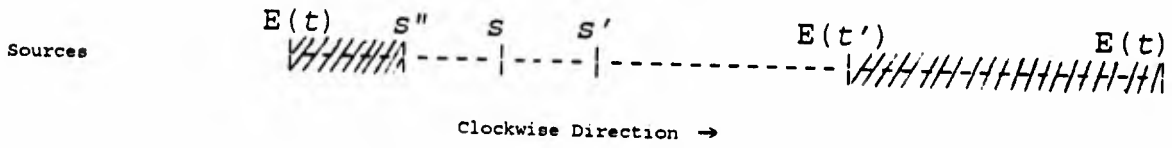

Figure 4.9 $s'' \prec s$ on arc $[E(t), s]$.

Therefore, $E(t'')$ is on arc $(s'', E(t'))$.

In order for (s'', t'') to cross with (s, t') either $s'' \prec s$ on arc $[E(t''), E(t')]$ or $s \prec s''$ on arc $[E(t'), E(t'')]$. But, this first condition cannot be met unless $E(t'')$ is on the arc $(E(t'), s'']$, which is the shaded region. The second condition cannot be met unless $E(t')$ is on the arc $[s'', s]$ which is not the case since $s'' \prec s \prec s'$ on arc $[E(t), E(t')]$. Therefore, neither condition is satisfied and (s'', t'') does not cross with (s, t'). Likewise, (s'', t'') does not cross with (s', t).

Similar arguments hold for Case II and Case III and for the instance in which (s'', t'') does not cross (s, t) or (s', t'). Therefore, we have shown that uncrossing (s, t) and (s', t') decreases the number of crossing pairs. □

Lemmas 4.5 and 4.6 show that uncrossing a pair strictly decreases the number of crossing pairs without increasing the cost of the assignment. Therefore, a crossing assignment (in particular an optimal assignment) can always be transformed into a non-crossing assignment by a series of uncrossings without increasing cost. Thus, we have proved the following.

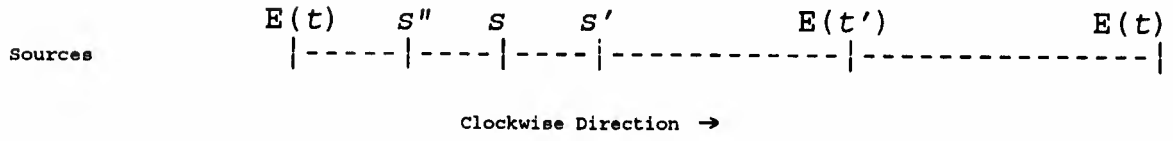

Figure 4.9 $s'' \prec s$ on arc $[E(t),s]$.

Therefore, $E(t'')$ is on arc $(s'',E(t')]$.

In order for (s'',t'') to cross with (s,t') either $s'' \prec s$ on arc $[E(t''),E(t')]$ or $s \prec s''$ on arc $[E(t'),E(t'')]$. But, this first condition cannot be met unless $E(t'')$ is on the arc $(E(t'),s'']$, which is the shaded region. The second condition cannot be met unless $E(t')$ is on the arc $[s'',s]$ which is not the case since $s'' \prec s \prec s'$ on arc $[E(t),E(t')]$. Therefore, neither condition is satisfied and (s'',t'') does not cross with (s,t'). Likewise, (s'',t'') does not cross with (s',t).

Similar arguments hold for Case II and Case III and for the instance in which (s'',t'') does not cross (s,t) or (s',t'). Therefore, we have shown that uncrossing (s,t) and (s',t') decreases the number of crossing pairs. □

Lemmas 4.5 and 4.6 show that uncrossing a pair strictly decreases the number of crossing pairs without increasing the cost of the assignment. Therefore, a crossing assignment (in particular an optimal assignment) can always be transformed into a non-crossing assignment by a series of uncrossings without increasing cost. Thus, we have proved the following.

Theorem 4.7 If $f:R\to R$ is convex and consistent, then there is a non-crossing optimal assignment for the problem (S,T,f,H).

Using this non-crossing result, in the next section we develop a generalized assignment problem on the line that is equivalent to an assignment problem on the circle.

4.3 An Equivalent Problem on the Line

We say two problems are **equivalent** when optimal solutions to each problem have equal values and there exists a mapping from an optimal solution of one problem to an optimal solution of the other.

To develop an assignment problem on the line equivalent to the problem (S,T,f,H) with $f:R\to R$ convex and consistent, we first consider the easier task of defining an equivalent problem on the line, denoted (S',t',f), for the problem (S,t,f,H) with a single destination. We wish to define S' and t' so that there is a one to one mapping from S onto S' with each source $s \in S$ mapped to a source $s' \in S'$ so that $c_C(s,t) = c_L(s',t')$. Recall that:

$$c_C(s,t) = \begin{cases} f(-\text{ccwd}(t,s)) & \text{if } s \text{ is on arc } [E(t),t) \\ f(\text{cwd}(t,s)) & \text{if } s \text{ is on arc } [t,E(t)). \end{cases}$$

Therefore, for each source s on the arc $[E(t),t)$ ($[t,E(t))$) there must be a corresponding source s' on the line with s'-t'

= -ccwd(t,s) (cwd(t,s)). We show in Lemma 4.8 that this occurs when we define t' and s' as

(4.8) $t' = d_E$

(4.9) $s' = cwd(E(t),s)$

This corresponds to cutting the circle at $E(t)$, making $E(t)$ the origin of the line, and locating s' and t' based on the positions of s and t with respect to $E(t)$.

Lemma 4.8 Consider the problem (S,t,f,H) where $f:R\rightarrow R$ is convex and consistent. For each source s and destination t,

$$c_c(s,t) = f(cwd(E(t),s) - d_E).$$

Proof: Source s is either on the arc $[E(t),t)$ or on the arc $[t,E(t))$. When s is on the arc $[E(t),t)$, by Lemma 4.3, $c_c(s,t) = f(-ccwd(t,s))$. In addition, $cwd(E(t),s) < cwd(E(t),t)$. This implies that $cwd(E(t),s) - d_E = cwd(E(t),s) - cwd(E(t),t) = -cwd(s,t) = -ccwd(t,s)$. A similar result follows when s is on arc $[t,E(t))$. Therefore, $c_c(s,t) = f(cwd(E(t),s) - d_E)$. □

By Lemma 4.8, the cost matrices $C_c(S,t,f,H)$ and $C_L(S',t',f)$ are equal when $t' = d_E$ and S' is defined as

(4.10) $S' = \{s'_i: s'_i = cwd(E(t),s_i) \text{ for } i = 1,\ldots,|S|\}$.

So we conclude that the problem (S,t,f,H) defined on the circle is equivalent to the problem (S',t',f) defined on the line.

To extend this transformation to problems (S,T,f,H) having more than one destination, we arbitrarily choose a "leftmost" destination $t_1 \in T$ and define the mapping $\alpha:S\rightarrow S'$ by

$$\alpha(s) = \mathrm{cwd}(E(t_1),s).$$

By Lemma 4.8, we know that $c_C(s,t_1) = c_L(\alpha(s),d_T)$ for each $s \in S$. The challenge remains to construct a mapping $\beta:T\rightarrow T'$ so that

$$c_C(s,t) = c_L(\alpha(s),\beta(t)) \text{ for each } s \in S \text{ and } t \in T.$$

Unfortunately, when there is more than one destination location, the transformation is not as straightforward as the previous case. Lemma 4.9 shows that if we define $\beta(t) = d_E + \mathrm{cwd}(t_1,t)$, then $c_C(s,t) = c_L(\alpha(s),\beta(t))$ whenever s is on arc $[E(t),E(t_1))$.

Lemma 4.9 Consider the problem (S,T,f,H) where $f:R\rightarrow R$ is convex and consistent and let $t_1 \in T$. Then

$$c_C(s,t) = c_L(\alpha(s),\beta(t))$$

whenever s is on the arc $[E(t),E(t_1))$, where $\alpha(s) = \mathrm{cwd}(E(t_1),s)$ and $\beta(t) = d_E + \mathrm{cwd}(t_1,t)$.

Proof: If $t = t_1$, then $\beta(t) = d_E$ and, by Lemma 4.8, the result is proven.

Now assume $t \neq t_1$, and note that $cwd(t_1,t) = cwd(E(t_1),E(t))$. Therefore, $\beta(t) = d_E + cwd(E(t_1),E(t))$. Also, since s is on the arc $[E(t),E(t_1))$, $\alpha(s) = cwd(E(t_1),s) = cwd(E(t_1),E(t)) + cwd(E(t),s)$. Therefore,

$$\alpha(s) - \beta(t) = cwd(E(t),s) - d_E,$$

and, by Lemma 4.8, $c_c(s,t) = c_L(\alpha(s),\beta(t))$. □

Unfortunately, when s is on arc $[E(t_1),E(t))$, this transformation does not work. In fact, in this case,

$$c_c(s,t) < c_L(\alpha(s),\beta(t)).$$

However, as we show in Lemma 4.10,

$$c_c(s,t) = c_L(\alpha(s)+H,\beta(t)).$$

Lemma 4.10 Consider the problem (S,T,f,H) where $f:R\to R$ is convex and consistent. For each $s \in S$ and $t \in T$

$$c_c(s,t) \leq c_L(\alpha(s),\beta(t)),$$

and, if $s \in [E(t_1),E(t))$,

$$c_c(s,t) = c_L(\alpha(s)+H,\beta(t)).$$

Proof: By Lemma 4.9, this is true for each pair ($s \in S, t \in T$) when s is on the arc $[E(t), E(t_1))$. Note that this is always the case when $t = t_1$. Now assume that $t \neq t_1$ and s is on the arc $[E(t_1), E(t))$. In this case,

$$\begin{aligned} \beta(t) &= d_E + cwd(t_1, t) \\ &= d_E + cwd(E(t_1), E(t)) \\ &= d_E + cwd(E(t_1), s) + cwd(s, E(t)). \end{aligned}$$

Therefore,

$$\begin{aligned} \alpha(s) - \beta(t) &= -cwd(E(t), t) - cwd(s, E(t)) \\ &= -(ccwd(t, E(t)) + cwd(s, E(t))) \\ &< -ccwd(t, E(t)). \end{aligned}$$

Hence, by Lemma 4.4 and the convexity of f,

$$c_C(s, t) \leq c_L(\alpha(s), \beta(t)).$$

However,

$$\begin{aligned} (\alpha(s)+H) - \beta(t) &= H - cwd(E(t), t) - cwd(s, E(t)) \\ &= H - cwd(E(t), t) - (H - cwd(E(t), s)) \\ &= cwd(E(t), s) - d_E. \end{aligned}$$

Therefore, by Lemma 4.8, $c_C(s, t) = c_L(\alpha(s)+H, \beta(t))$. □

Note that each arc $[E(t_1), E(t))$ for $t \neq t_1$ is included in the arc $[E(t_1), E(t_{|T|}))$. Thus, given a problem (S, T, f, H) on

the circle, we define the sets S" and T" of locations on the line as follows:

$$S'' = \{s''_i : s''_i = \alpha(s_i), \text{ for } i = 1,\ldots,|S|\} \cup \{s''_{i+|S|} : s''_{i+|S|} = \alpha(s_i)+H, \text{ for each } s_i \text{ on the arc } [E(t_1), E(t_{|T|})) \text{ when } t_1 \neq t_{|T|}\}$$

$$T'' = \{t''_i : t''_i = \beta(t_i), \text{ for } i = 1,\ldots,|T|\}.$$

Therefore, for each pair $(s \in S, t \in T)$ there is a pair $(s'' \in S'', t'' \in T'')$ with $c_C(s,t) = c_L(s',t')$, and, given an assignment for the problem on the circle, we can define an assignment on the line of equal cost. However, the converse is not necessarily true as an assignment on the line could match both images of a single source on the circle. We now consider a way to ensure that this does not occur.

Note that any contiguous set of no more than $|S|$ sources in S" cannot contain both images of any source in S. Therefore, to ensure that an assignment of T" into S" matches no more than one image of a source in S, we add the restriction that the matched sources of S" must be contained in a contiguous set of no more than $|S|$ sources. We refer to this restriction as the **range constraint** and denote a problem on the line with the constraint that matched sources must be contained in a contiguous set of no more than I sources by $(S,T,f,\leq I)$.

We call $(S'',T'',f,\leq|S|)$ the **corresponding problem on the line** for (S,T,f,H), and say that $(S'',T'',f,\leq|S|)$ corresponds to

(S,T,f,H) and vice versa. We now define transformations from an assignment for one problem to an assignment for its corresponding problem.

Let μ be a non-crossing assignment for (S,T,f,H). Define an assignment $\mu"$ for the corresponding problem on the line $(S",T",f,\leq|S|)$ as follows for each $t \in T$:

$$(4.11) \qquad \mu"(\beta(t)) = \begin{cases} \alpha(\mu(t)), & \text{if } \mu(t_1) \leq \mu(t) \text{ with respect to } E(t_1) \\ \alpha(\mu(t))+H, & \text{otherwise.} \end{cases}$$

Lemma 4.11 shows that $\mu"$ is an assignment satisfying the range constraint and has the same cost as μ.

Lemma 4.11 Consider the problem (S,T,f,H) with $f:R\to R$ convex and consistent. If μ is a non-crossing assignment for (S,T,f,H), then the assignment $\mu"$ defined by (4.11) for the corresponding problem on the line $(S",T",f,\leq|S|)$ satisfies the range constraint and $Cost(\mu") = Cost(\mu)$.

Proof: We first show $\mu"$ is an assignment, i.e. each destination is matched with a single source, and each source is matched with no more than one destination. Since μ is an assignment and $\beta:T\to T"$ is one-to-one and onto, equation (4.11) matches each destination of T" with a single source. In addition, each matched source of μ is matched with a single destination. Therefore, since equation (4.11) matches exactly one image of each matched source in S, $\mu"$ is an assignment.

We now consider the range constraint. Assume $s_j \equiv \mu(t_1)$ and $\mu(\beta(t_1)) \equiv s"_j$ and consider the contiguous set $[s"_j, s"_{j+|S|-1}]$ of S", which contains $|S|$ sources. Note that $s"_j$ is the lowest indexed source $s \in S"$ with $\alpha(s_j) \leq s$ and $s"_{j+|S|-1}$ is the highest indexed source $s \in S"$ with $s < \alpha(s_j)+H$. By (4.11), every matched source $s_k \geq s_j$ on the arc $[s_j, E(t_1))$ corresponds to a matched source $s"_k = \alpha(s_k) = \alpha(s_j) + cwd(s_j, s_k) < \alpha(s_j)+H$. Therefore, $s"_j \leq s"_k \leq s_{j+|S|-1}$ and $s"_k$ is in the set $[s"_j, s"_{j+|S|-1}]$. A similar result holds for the other matched sources of S" and $\mu"$ satisfies the range constraint.

We now show that the costs of assignments μ and $\mu"$ are equal. Note that equation (4.11) associates each pair $(s \equiv \mu(t), t)$ on the circle with a matched pair $(s" \equiv \mu"(t"), \beta(t) \equiv t")$ on the line. If $s \prec \mu(t_1)$ with respect to $E(t_1)$, then $t \neq t_1$ and $s" \equiv \alpha(s)+H$. Since $t \neq t_1$ and μ is non-crossing, $E(t)$ is on the arc $(s, E(t_1))$. Therefore, s is on the arc $[E(t_1), E(t))$ and by Lemma 4.10, $c_C(s,t) = c_L(s",t")$. A similar argument holds when $\mu(t_1) \preceq s$ on arc $[E(t_1), E(t_1))$. Therefore, $Cost(\mu") = Cost(\mu)$. □

We now define an assignment μ for the problem (S, T, f, H) based on an assignment $\mu"$ for its corresponding problem on the line. Define μ as follows for each $t \in T$:

(4.12) $\mu(t) \equiv s$, if $(\alpha(s), \beta(t))$ or $(\alpha(s)+H, \beta(t))$ are a matched pair of $\mu"$.

The following lemma shows μ is an assignment and the cost of μ does not exceed the cost of μ".

Lemma 4.12 Consider the problem (S,T,f,H) with f:$R\rightarrow R$ convex and consistent. If μ" is an assignment for the corresponding problem on the line, then μ, defined by (4.12) is an assignment for (S,T,f,H) with Cost(μ) $\leq$ Cost(μ").

Proof: We first show μ is an assignment. Since β:T$\rightarrow$T" is one-to-one and onto, and each destination in T" is matched with the image of a source in S, equation (4.12) matches each destination in T with a source in S. Also, since μ" satisfies the range constraint, each matched source of S" is the image of a distinct source in S. Therefore, equation (4.12) does not match a source in S with more than one destination in T, and μ is an assignment.

By Lemma 4.10, the cost of a matched pair (s",t") of μ", does not exceed the cost of the matched pair of μ generated by (4.12). Therefore, Cost(μ) $\leq$ Cost(μ"). □

Given these results, Theorem 4.13 shows that a problem on the circle and its corresponding problem on the line are equivalent.

Theorem 4.13 The problem (S,T,f,H) with f:$R\rightarrow R$ convex and consistent and the corresponding problem on the line

$(S",T",f,\leq|S|)$ are equivalent.

Proof: By Theorem 4.7 we know there exists a non-crossing optimal assignment μ for (S,T,f,H). By Lemma 4.11, equation (4.11) defines an assignment $\mu"$ for $(S",T",f,\leq|S|)$, satisfying the range constraint, with $Cost(\mu") = Cost(\mu)$. $\mu"$ must be optimal or by Lemma 4.12 we could generate an assignment μ' for (S,T,f,H) with $Cost(\mu') < Cost(\mu") = Cost(\mu)$, contradicting the optimality of μ.

Similarly, equation (4.12) transforms an optimal assignment $\mu"$ for $(S",T",f,\leq|S|)$ to an optimal assignment μ for (S,T,f,H) with $Cost(\mu") = Cost(\mu)$.

Therefore, given an optimal assignment for one problem, we can transform it to an optimal assignment for the other problem at the same cost and these problems are equivalent. □

Note that given an optimal solution to the corresponding problem on the line, we can use (4.12) to define an optimal solution for our original assignment problem on the circle. We now develop solution algorithms for the corresponding problem on the line.

4.4 Solving the Corresponding Problem on the Line

A problem $(S,T,f,\leq I)$, corresponding to a problem on a circle of circumference H, has three characteristics that

distinguish it from a general assignment problem on the line. First, an assignment must satisfy a range constraint, i.e. all matched sources must be contained in a contiguous set of sources of cardinality no more than I. Second, destinations t_1 and $t_{|T|}$ are located with $t_{|T|} - t_1 < H$. Third, sources are located so that, for each pair of sources s_i and s_j with $i < j$: if $j-i < I$, then $s_j - s_i \leq H$; if $j-i = I$, then $s_i - s_j = H$; and, if $j-i > I$, then $s_j - s_i \geq H$. We refer to these last two characteristics as the **circularity conditions**. Throughout this section, we assume that the problem $(S,T,f,\leq I)$ satisfies the circularity conditions for a circle with circumference H.

In Section 3.3 we developed LEFT-SHIFT, which solves the assignment problem on the line. Since the problem $(S,T,f,\leq I)$ is an assignment problem on the line with an added constraint, if the assignment μ, generated by LEFT-SHIFT for the problem (S,T,f) satisfies the range constraint, then μ is also optimal for the problem $(S,T,f,\leq I)$. This will always be the case when $I = |S|$; since, in this case, every assignment satisfies the range constraint. However, μ will not always satisfy the range constraint, so we must develop more general optimality conditions that account for this.

We first show that, even with the range constraint, when assignment costs are determined by a convex function we need only consider non-crossing assignments.

Lemma 4.14 There is a non-crossing optimal assignment for the problem $(S,T,f,\leq I)$ when $f:R\rightarrow R$ is convex.

Proof: Consider an optimal assignment μ for $(S,T,f,\leq I)$. Every assignment for the problem $(\mu(T),T,f,\leq I)$ is an assignment for $(\mu(T),T,f)$. By Theorem 3.5, there is a non-crossing optimal assignment μ' for $(\mu(T),T,f)$. Note that μ' satisfies the range constraint and $Cost(\mu') \leq Cost(\mu)$. Therefore, the non-crossing assignment μ' is optimal for $(S,T,f,\leq I)$. □

As in Chapter 3, there are special cases of the problem $(S,T,f,\leq I)$ that are relatively easy to solve because of this non-crossing result. In Chapter 3, this was true for problems on the line with $|S| = |T|$. A similar result holds for the problem $(S,T,f,\leq I)$ when $f:R\rightarrow R$ is convex and $|S| = |T|$. In this case there is only one non-crossing assignment μ, with $\mu(t_i) \equiv s_i$ for $i = 1,\ldots,|T|$, which by Lemma 4.14 must be optimal. In addition, the corresponding problem on the line is easily solved whenever $I = |T|$, which we now discuss.

When $I = |T|$, each non-crossing assignment μ consists of a single maximal block $[t_1, t_{|T|}]$. A simple solution procedure is to calculate the assignment cost for each possible non-crossing assignment and choose the least cost assignment, taking $O(|S||T|)$ steps. However, a more efficient procedure is possible, based on characteristics of the assignment costs

of this block.

The following property of the assignment costs results from the circularity conditions.

Lemma 4.15 Consider a problem $(S,T,f,\leq I)$ with $f:R\rightarrow R$ convex. If s_{i-1}, s_i, s_{I+i-1}, s_{I+i} are sources, then

$$f(s_{i-1}-t_1) - f(s_i-t_1) \geq f(s_{I+i-1}-t_{|T|}) - f(s_{I+i}-t_{|T|}).$$

Proof: Note that by the circularity conditions, $t_1 + H > t_{|T|}$; hence, by Lemma 3.10: $f(s_{I+i-1}-t_{|T|}) - f(s_{I+i}-t_{|T|}) \leq f(s_{I+i-1}-(t_1+H)) - f(s_{i+i}-(t_1+H))$. The desired result follows immediately from this equation by noting that $s_{I+i-1} = s_{i-1} + H$ and $s_{I+i} = s_i + H$. □

Using this result, Lemma 4.16 characterizes the changes in cost due to a series of left or right shifts with respect to a non-crossing solution.

Lemma 4.16 Consider the problem $(S,T,f,\leq I)$ with $f:R\rightarrow R$ convex and $I = |T|$. Let μ be a non-crossing assignment for $(S,T,f,\leq I)$ for which the cost of a left (right) shift with respect to the block $[t_1, t_{|T|}]$ is non-negative. The change in cost due to each subsequent left (right) shift in a series of left (right) shifts with respect to this block is also non-negative.

Proof: Consider a series of left shifts of μ with respect to $[t_1, t_{|T|}]$, the proof for a series of right shifts is similar. Let Δ_z denote the change in cost due to the z^{th} left shift of μ and let $\Delta_{[p,z]}$ denote the change in assignment cost for t_p due to shift number z. We are given that $\Delta_1 \geq 0$ and assume this is true for z-1 left shifts. Let μ" denote the assignment resulting from z left shifts of μ with respect to $[t_1, t_{|T|}]$. The changes in cost due to left shifts z-1 and z are:

$$\Delta_{z-1} = \sum_{x=1,|T|-1}\Delta_{[x,z-1]} + \Delta_{[|T|,z-1]}$$

$$\Delta_z = \Delta_{[1,z]} + \sum_{x=2,|T|}\Delta_{[x,z]}.$$

We now show that $\Delta_z \geq \Delta_{z-1}$.

By Lemma 3.10, $\sum_{x=1,|T|-1}\Delta_{[x,z-1]} \leq \sum_{x=2,|T|}\Delta_{[x,z]}$. Now consider $\Delta_{[1,z]}$ and $\Delta_{[|T|,z-1]}$.

Letting $s_{i-1} \equiv \mu"(t_1)$ and $s_q \equiv \mu"(t_{|T|})$, note that q = I+i-2. Then,

$$\Delta_{[|T|,z-1]} = f(s_{I+i-1}-t_{|T|}) - f(s_{I+i}-t_{|T|}), \text{ and}$$

$$\Delta_{[1,z]} = f(s_{i-1}-t_1) - f(s_i-t_1).$$

By Lemma 4.15, $\Delta_{[1,z]} \geq \Delta_{[|T|,z-1]}$. Therefore, $\Delta_z \geq \Delta_{z-1} \geq 0$.

□

It follows immediately from Lemma 4.16 that a non-crossing solution for which a right and a left shift of the

maximal block $[t_1, t_{|T|}]$ do not reduce costs is optimal. In addition, by Lemma 4.16, given a candidate non-crossing assignment μ, if a left (right) shift reduces costs, then the optimal non-crossing assignment μ^* must have $\mu^*(t_1) \prec \mu(t_1)$ ($\mu(t_1) \prec \mu^*(t_1)$). Therefore, the following BISECTION SEARCH procedure generates an optimal assignment, which is stated formally as Theorem 4.17. In this procedure $\lfloor x \rfloor$ denotes the largest integer not greater than a real number x.

BISECTION SEARCH

STEP 1: Initial Solution:

STEP 1a: Sort the destinations and sources by location, assigning indices so that:

$s_1 \le s_2 \le \ldots \le s_{|S|}$ and $t_1 \le t_2 \le \ldots \le t_{|T|}$.

STEP 1b: Let i = 1 and j = |S|-|T|+1. The set of possible mates for t_1 is the contiguous set $[s_i, s_j]$. Let $k = \lfloor (j+i)/2 \rfloor$.

STEP 1c: Define the assignment μ as: $\mu(t_p) \equiv s_{k+p-1}$ for each $t_p \in T$.

STEP 2: Bisection Search: If i = j, STOP. Otherwise:

STEP 2a: Calculate the changes in cost due to right and left shifts of μ with respect to $[t_1, t_{|T|}]$.

-- If both these changes in cost are non-negative, STOP.
-- If a left shift would reduce costs, redefine the set of possible t_1 mates, letting j := k-1 and $k := \lfloor (j+i)/2 \rfloor$.
-- Otherwise, let i := k+1 and $k := \lfloor (j+i)/2 \rfloor$.

STEP 2b: Redefine μ, letting $\mu(t_p) \equiv s_{k+p-1}$ for each $t_p \in T$. Return to STEP 2.

The sort in STEP 1 can be accomplished in $O(|S|\log|S|)$ computational steps. In STEP 2 there are no more than $O(\log|S|)$ candidate assignments for t_1 before $i = j$, with each of these steps taking no more than $O(|T|)$ steps to calculate shift costs. Therefore the procedure is finite, taking $O(|S|\log|S|)$ steps.

We have already shown that the solution generated by BISECTION SEARCH is optimal, which is stated below as Theorem 4.17.

Theorem 4.17 When the function $f:R\to R$ is convex and $I = |T|$, BISECTION SEARCH generates an optimal assignment μ for the problem $(S,T,f,\leq I)$.

We now consider problems $(S,T,f,\leq I)$ with $I > |T|$. We first define terms needed to describe an assignment that satisfies the range constraint.

Given a non-crossing assignment μ for $(S,T,f,\leq I)$ with $f:R\to R$ convex, the range constraint is **binding** when the contiguous set $[\mu(t_1),\mu(t_{|T|})]$ contains I sources. Now consider the blocks of μ. We say the block $[a,b]$ of μ is **constrained** when the range constraint is binding and $a \equiv t_1$ or

$b \equiv t_{|T|}$. Otherwise, $[a,b]$ is **unconstrained**.

An assignment shift, as defined in Chapter 3, with respect to a constrained block could result in an assignment that violates the range constraint. For instance, consider an assignment μ with constrained maximal blocks $[t_1, t_p]$ and $[t_q, t_{|T|}]$. A left shift of μ with respect to $[t_1, t_p]$ would violate the range constraint unless the assignment of $[t_q, t_{|T|}]$ is also changed. Therefore, we now define an assignment shift with respect to a constrained block.

Let μ be a non-crossing assignment for the problem $(S,T,f,\leq I)$ and let $[t_1, t_i]$ and $[t_j, t_{|T|}]$ be maximal constrained blocks of μ. When $s_1 \prec \mu(t_1)$, a **left shift of μ with respect to** $[t_1, t_p]$, with $p \leq i$, results in the assignment μ' defined as: for each destination t in $[t_1, t_p] \cup [t_j, t_{|T|}]$ with $\mu(t) \equiv s_j$: $\mu'(t) \equiv s_{j-1}$, and for every other destination t': $\mu'(t') \equiv \mu(t')$. For example, in Figure 4.10 we show an assignment with constrained maximal blocks $[t_1, t_4]$ and $[t_6, t_6]$. Figure 4.11 shows the assignment resulting from a left shift with respect to $[t_1, t_3]$.

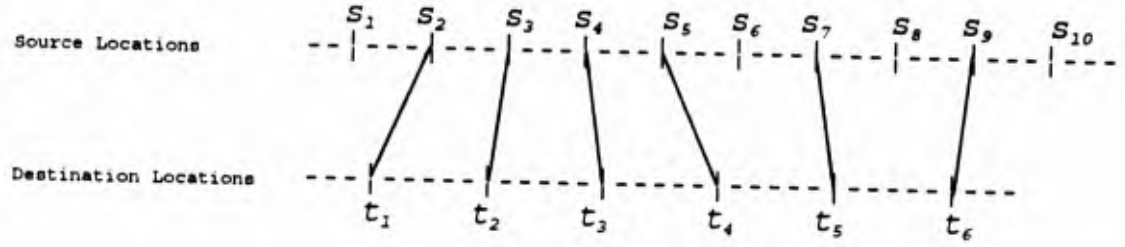

Figure 4.10 An assignment μ with constrained maximal blocks $[t_1, t_4]$ and $[t_6, t_6]$.

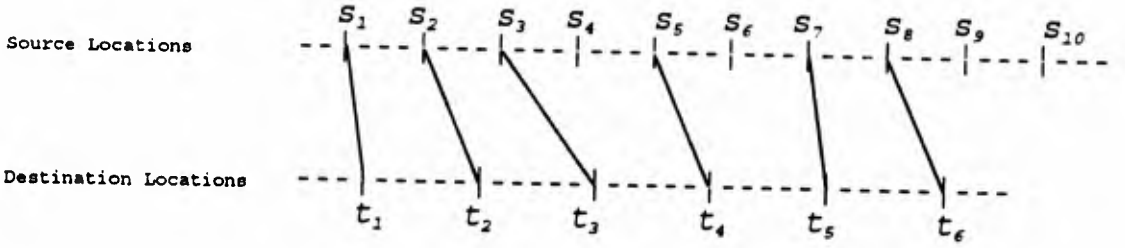

Figure 4.11 The assignment μ' resulting from a left shift of μ with respect to $[t_1, t_3]$.

Likewise, when $\mu(t_{|T|}) \prec s_{|S|}$, a **right shift of μ with respect to** $[t_q, t_{|T|}]$, with $j \leq q$, results in the assignment μ' defined as: for each destination t in $[t_q, t_{|T|}] \cup [t_1, t_i]$ with $\mu(t) \equiv s_j$: $\mu'(t) \equiv s_{j+1}$, and for every other destination t': $\mu'(t') \equiv \mu(t')$.

We now develop sufficient conditions for optimality. Let μ be a non-crossing assignment for $(S, T, f, \leq I)$. As in Chapter 3, an unconstrained maximal block $[a,b]$ of μ is **valid** when $\mu(a) \leq r(S^*(a))$ and $\ell(S^*(b)) \leq \mu(b)$. We say a constrained maximal block $[t_1, a]$ ($[b, t_{|T|}]$) is valid when $\ell(S^*(a)) \leq \mu(a)$ ($\mu(b) \leq r(S^*(b))$. As before, an assignment μ is **valid** only

when each maximal block of μ is valid. The following lemma, using the result of Lemma 3.9, shows that there is a valid non-crossing optimal assignment for $(S,T,f,\leq I)$ whenever $f:R\rightarrow R$ is convex.

Lemma 4.18 There is a valid non-crossing optimal assignment for the problem $(S,T,f,\leq I)$ when $f:R\rightarrow R$ is convex and $I \geq |T|$.

Proof: By Lemma 4.14, there is a noncrossing optimal assignment μ for $(S,T,f,\leq I)$. Note that a valid assignment for the problem $([\mu(t_1),\mu(t_{|T|})],T,f)$ is also valid for $(S,T,f,\leq I)$. By Lemma 3.9 there is a valid non-crossing optimal assignment μ' for $([\mu(t_1),\mu(t_{|T|})],T,f)$. Note that $\text{Cost}(\mu') \leq \text{Cost}(\mu)$. Therefore, μ' is a valid non-crossing optimal assignment for $(S,T,f,\leq I)$. □

As in Section 3.3, a non-crossing assignment μ for $(S,T,f,\leq I)$ is **locally optimal** if it is valid and no shift with respect to a block of μ leads to a lower cost assignment. We say that μ is locally optimal for a block $[a,b]$ of μ if the partial assignment μ' with $\mu'(t) \equiv \mu(t)$ for each $t \in [a,b]$ is locally optimal. An optimal solution is necessarily locally optimal. To show that this condition is also sufficient, we first consider the unconstrained maximal blocks of μ.

Lemma 4.19 Let μ be a non-crossing assignment for the problem $(S,T,f,\leq I)$ with $f:R\rightarrow R$ convex. If μ is locally optimal for an unconstrained maximal block $[a,b]$, then the partial assignment μ', defined by $\mu'(t) \equiv \mu(t)$ for each $t \in [a,b]$, is optimal.

Proof: Every assignment for $(S,[a,b],f,\leq I)$ is also feasible for the problem $(S,[a,b],f)$. By Theorem 3.13, μ' is optimal for $(S,[a,b],f)$; therefore, μ' is optimal for $(S,[a,b],f,\leq I)$. □

We now consider constrained maximal blocks of non-crossing locally optimal assignments and first examine changes in cost due to a series of left or right shifts.

Lemma 4.20 Let μ be a non-crossing assignment for the problem $(S,T,f,\leq I)$ where $f:R\rightarrow R$ is convex and $I > |T|$ and assume μ is locally optimal for maximal constrained blocks $[t_1,t_i]$ and $[t_j,t_{|T|}]$. If μ' is the partial assignment defined by $\mu'(t) \equiv \mu(t)$ for each $t \in [t_1,t_i] \cup [t_j,t_{|T|}]$, then each shift in a series of left (right) shifts of μ' with respect to $[t_1,t_i]$ ($[t_j,t_{|T|}]$) resulting in a valid assignment does not reduce assignment costs.

Proof: Consider a series of left shifts of μ' with respect to $[t_1,t_i]$. Let Δ_z denote the change in cost due to the z^{th} left shift of μ' and let $\Delta_{[p,z]}$ denote the change in assignment cost for t_p due to shift number z. We are given

that $\Delta_1 \geq 0$ and assume this is true for z-1 left shifts. Let μ" denote the assignment resulting from z left shifts of μ' with respect to $[t_1, t_i]$. We assume that i $\neq$ 1 and j $\neq$ |T|. Otherwise, delete the appropriate summations in equations (4.13) and (4.14) that follow. The changes in cost due to left shifts z-1 and z are:

$$(4.13) \qquad \Delta_{z-1} = \sum_{x=1,i-1}\Delta_{[x,z-1]} + \Delta_{[i,z-1]} + \sum_{x=j,|T|-1}\Delta_{[x,z-1]} + \Delta_{[|T|,z-1]}$$

$$(4.14) \qquad \Delta_z = \Delta_{[1,z]} + \sum_{x=2,i}\Delta_{[x,z]} + \Delta_{[j,z]} + \sum_{x=j+1,|T|}\Delta_{[x,z]}$$

We now show that $\Delta_z \geq \Delta_{z-1}$.

By Lemma 3.10, $\sum_{x=1,i-1}\Delta_{[x,z-1]} \leq \sum_{x=2,i}\Delta_{[x,z]}$ and $\sum_{x=j,|T|-1}\Delta_{[x,z-1]} \leq \sum_{x=j+1,|T|}\Delta_{[x,z]}$. Since μ" is valid, by Lemma 3.8, $\Delta_{[i,z-1]} \leq 0$ and $\Delta_{[j,z]} \geq 0$. Now consider $\Delta_{[1,z]}$ and $\Delta_{[|T|,z-1]}$.

Letting $s_{i-1} \equiv \mu"(t_1)$ and $s_q \equiv \mu"(t_{|T|})$, note that q = I+i-2. Then,

$$\Delta_{[|T|,z-1]} = f(s_{I+i-1}-t_{|T|}) - f(s_{I+i}-t_{|T|}), \text{ and}$$

$$\Delta_{[1,z]} = f(s_{i-1}-t_1) - f(s_i-t_1).$$

By Lemma 4.15, $\Delta_{[1,z]} \geq \Delta_{[|T|,z-1]}$.

For our induction we assumed that $\Delta_{z-1} \geq 0$ and by our above discussion, $\Delta_z \geq \Delta_{z-1}$. Therefore, $\Delta_z \geq 0$.

The argument for a series of right shifts of μ' with respect to $[t_j, t_{|T|}]$ is similar. □

Lemma 4.21 now extends the results of Lemma 4.20 to include subsets of these constrained blocks and additional shifts that do not result in valid assignments.

Lemma 4.21 Let μ be a non-crossing assignment for the problem $(S,T,f,\leq I)$ where $f:R\rightarrow R$ is convex and $I > |T|$ and assume μ is locally optimal for maximal constrained blocks $[t_1,t_i]$ and $[t_j,t_{|T|}]$. If μ' is the partial assignment defined by $\mu'(t) \equiv \mu(t)$ for each $t \in [t_1,t_i] \cup [t_j,t_{|T|}]$, then each shift in a series of left (right) shifts of μ' with respect to $[t_1,t_a]$, for $1 \leq a < i$, ($[t_b,t_{|T|}]$, for $j < b \leq |T|$) does not reduce assignment costs.

Proof: Consider a series of left shifts with respect to $[t_1,t_a]$ that do not change the mates of the destinations in $[t_{a+1},t_i]$. Since μ' is locally optimal, $\Delta_1 \geq 0$. Assume $\Delta_{z-1} \geq 0$ and let $\mu"$ be the assignment after z-1 shifts. If $\ell(S^*(t_a)) \prec \mu"(t_a)$, then, by Lemma 4.20, $\Delta_z \geq 0$. Otherwise, let t_p be the highest indexed destination in $[t_1,t_a]$ for which $\ell(S^*(t_p)) \prec \mu"(t_p)$. If t_p does not exist, then $\mu"(t_q) \leq \ell(S^*(t_q))$ for each $t_q \in [t_1,t_a]$ and, by Lemma 3.8, $\Delta_{[q,z]} \geq 0$, implying $\Delta_z \geq 0$.

When t_p exists, the change in assignment costs due to the z^{th} shift is:

$$\sum_{x=1,p}\Delta_{[x,z]} + \sum_{x=j,|T|}\Delta_{[x,z]} + \sum_{x=p+1,a}\Delta_{[x,z]}.$$

By Lemma 4.20, the sum of the first two terms is non-negative,

and by Lemma 3.8, the last term is non-negative. Therefore, $\Delta_z \geq 0$.

Since a series of left shifts which change the mates of $[t_{a+1}, t_i]$ can be decomposed into a series of left shifts with respect to $[t_1, t_i]$ followed by a series of left shifts with respect to $[t_1, t_a]$, the desired result follows for left shifts with respect to $[t_1, t_a]$.

The proof for a series of right shifts with respect to $[t_b, t_{|T|}]$ is similar. □

From these results we now show that local optimality is sufficient for the optimality of a partial non-crossing assignment of constrained maximal blocks.

Lemma 4.22 Consider a non-crossing assignment μ for the problem $(S,T,f,\leq I)$ with $f:R\rightarrow R$ convex and $I > |T|$. If $[t_1, t_i]$ and $[t_j, t_{|T|}]$ are constrained maximal blocks of μ and the partial assignment μ', defined by $\mu'(t) \equiv \mu(t)$ for each $t \in [t_1, t_i] \cup [t_j, t_{|T|}]$, is locally optimal, then μ' is optimal.

Proof· Let $T' = [t_1, t_i] \cup [t_j, t_{|T|}]$ and $S' = [\mu'(t_1), \mu'(t_{|T|})]$. Since the range constraint is binding, $|S'| = I$. Note that, since μ' is a locally optimal non-crossing solution for $(S,T',f,\leq I)$, it is non-crossing and locally optimal for the problem (S',T',f).

Now consider the problem $(S,T',f,\leq I)$. By Lemma 4.18, there is a valid non-crossing optimal assignment μ'' for this problem. Consider the case in which $\mu''(t_1) \prec \mu'(t_1)$ and let $S'' = [\mu''(t_1), \mu'(t_{|T|})]$. Since μ' is valid, $\ell(S^*(t_i)) \leq \mu'(t_i)$ and, by Lemma 3.12, it cannot reduce costs to match t_i with a source $s \succ \mu'(t_i)$. Therefore, we assume that $\mu''(t_i) \preceq \mu'(t_i)$. Also, because μ'' satisfies the range constraint, $\mu''(t) \prec \mu'(t)$ for each destination t in $[t_j, t_{|T|}]$. Since the block $[t_j, t_{|T|}]$ of μ' is locally optimal for (S',T',f), it is locally optimal for the problem (S'',T',f). Thus, by Lemma 3.12, each shift in a series of left shifts of μ' with respect to a subset of $[t_j, t_{|T|}]$ would not reduce costs. Therefore, we may assume that $[t_j, t_{|T|}]$ is a block of μ'' and the range constraint is binding for μ''. Based on these observations, a series of left shifts of μ' with respect to subsets of the constrained maximal block $[t_1, t_i]$ results in μ''. By Lemmas 4.20 and 4.21, these shifts cannot reduce assignment costs. Therefore, $\text{Cost}(\mu') \leq \text{Cost}(\mu'')$ and μ' is optimal.

The proof when $\mu'(t_1) \equiv \mu''(t_1)$ or $\mu'(t_1) \prec \mu''(t_1)$ is similar and we have shown that μ' is an optimal partial assignment. □

Therefore, by Lemmas 4.19 and 4.22, the partial assignment of each maximal block of a non-crossing locally optimal assignment is optimal. Theorem 4.23 follows directly

from this observation.

Theorem 4.23 If $f:R\to R$ is convex and $I > |T|$, then a non-crossing locally optimal assignment μ for the problem $(S,T,f,\leq I)$ is optimal.

Using these results, we now develop solution procedures for problems $(S,T,f,\leq I)$ with $f:R\to R$ convex and $|T| < I$. Since this is an assignment problem on the line with an added constraint, we take the approach of first using LEFT-SHIFT to generate an assignment μ for (S,T,f), then, as necessary, altering μ to satisfy the range constraint. We now use μ to define optimality conditions for assignments for $(S,T,f,\leq I)$.

Lemma 4.24 Let μ' be a non-crossing assignment for the problem $(S,T,f,\leq I)$ with $f:R\to R$ convex and $|T| < I$, and let μ be the assignment generated by LEFT-SHIFT for (S,T,f). Assume $[a,b]$ is a block of μ' and a maximal block of μ. If $\mu(t) \equiv \mu'(t)$ for each $t \in [a,b]$, then the partial assignment $\mu"$ for $(S,T,f,\leq I)$, defined by $\mu"(t) \equiv \mu(t)$ for each $t \in [a,b]$, is optimal.

Proof: Note that $|[a,b]| \leq |T| < I$. Therefore, $\mu"$ satisfies the range constraint for $(S,[a,b],f,\leq I)$. Note that $\mu"$ is optimal for the problem $(S,[a,b],f)$. Since every assignment for $(S,[a,b],f,\leq I)$ is feasible for $(S,[a,b],f)$, $\mu"$

is also optimal for $(S,[a,b],f,\leq I)$. □

We now consider the constrained blocks of an assignment for $(S,T,f,\leq I)$.

Lemma 4.25 Consider a non-crossing assignment μ' for the problem $(S,T,f,\leq I)$ with $f:R\rightarrow R$ convex and $|T| < I$, and let μ be the assignment generated by LEFT-SHIFT for (S,T,f). If the following three conditions are satisfied:

(1) $[t_1,t_i]$ and $[t_j,t_{|T|}]$ are maximal constrained blocks of μ' with $\mu(t) \leq \mu'(t)$ for each $t \in [t_1,t_i]$ and $\mu'(t) \leq \mu(t)$ for each $t \in [t_j,t_{|T|}]$,

(2) μ has maximal blocks $[a,t_i]$ and $[t_j,b]$,

(3) neither a left shift of μ' with respect to $[t_1,t_i]$ or a right shift of μ' with respect to $[t_j,t_{|T|}]$ reduce costs,

then the partial assignment μ'' with $\mu''(t) \equiv \mu'(t)$ for each $t \in [t_1,t_i] \cup [t_j,t_{|T|}]$ is optimal.

Proof: Since μ is generated by LEFT-SHIFT, $\ell(S^*(t_i)) \leq \mu(t_i) \leq \mu''(t_i)$ and $\mu''(t_j) \leq \mu(t_j) \leq r(S^*(t_j))$. Therefore, μ'' is a valid non-crossing assignment. We now show that μ'' is locally optimal.

Since t_i is the highest indexed destination in a maximal block of μ and $\mu(t) \leq \mu''(t)$ for each $t \in [t',t_i] \subseteq [t_1,t_i]$, by Lemma 3.12, a right shift of μ'' with respect to $[t',t_i]$ will not reduce assignment costs. Similarly, a left shift of μ''

with respect to a subset of $[t_j, t_{|T|}]$ will not reduce assignment costs.

Now consider a left shift of μ" with respect to $[t_1, t_k]$ $\subset [t_1, t_i]$ and assume $\mu(t_1) \equiv \mu"(t_1)$. Note that a left shift of μ" with respect to $[t_1, t_k]$ can be decomposed into a left shift with respect to $[t_j, t_{|T|}]$ followed by a left shift with respect to $[t_1, t_k]$. We have already shown that a left shift with respect to $[t_j, t_{|T|}]$ does not reduce costs, and, by Lemma 3.12, a left shift of μ with respect to $[t_1, t_i]$ does not reduce costs. Therefore, in this case, a left shift of μ" with respect to $[t_1, t_k]$ will not reduce costs.

Similar arguments apply when $\mu(t_1) \prec \mu"(t_1)$ (by decomposing the left shift into a left shift with respect to $[t_1, t_j]$ followed by a right shift with respect to $[t_{k+1}, t_j]$) and for right shifts of μ" with respect to subsets of $[t_j, t_{|T|}]$. Therefore, μ" is locally optimal and, hence, is an optimal assignment. □

We now present our solution algorithm, called the RESTRICTED LEFT-SHIFT procedure. In this procedure we first generate a solution using LEFT-SHIFT, then alter this assignment as necessary to satisfy the range constraint.

RESTRICTED LEFT-SHIFT

STEP 1: Initial Solution: Using LEFT-SHIFT, generate the assignment μ for (S,T,f). If μ satisfies the range constraint, STOP. Otherwise, continue with STEP 2.

STEP 2: Initial Feasible Solution: Let μ' be the assignment resulting from a right shift of the current assignment with respect to the maximal block $[t_1,b]$. If μ' satisfies the range constraint, go to STEP 3. Otherwise, return to STEP 2.

STEP 3: Adjust Constrained Blocks: Let $[t_1,t_a]$ be the largest block of μ' with $\mu(t_a) \prec \mu'(t_a)$. If it would reduce costs, perform a left shift of μ' with respect to $[t_1,t_a]$, redefine μ' as the current assignment, and return to STEP 3. Otherwise, STOP.

RESTRICTED LEFT-SHIFT terminates in a finite number of steps since LEFT-SHIFT is finite and at most $2*|S|$ shifts can take place in STEP 2 and 3. In fact, the work in STEP 2 and 3 is no more than the work required by LEFT-SHIFT, and these procedures have the same worst case computational complexity when f is piecewise linear. This analysis is presented in Section 4.5. The following theorem shows that RESTRICTED LEFT-SHIFT generates an optimal solution.

Theorem 4.26 If the function $f:R\rightarrow R$ is convex and $I > |T|$, RESTRICTED LEFT-SHIFT generates an optimal solution for the problem $(S,T,f,\leq I)$.

Proof: LEFT-SHIFT produces a locally optimal non-crossing assignment μ, which by Theorem 3.17, is an optimal solution for (S,T,f). Therefore, if μ satisfies the range

constraint, μ is an optimal assignment for $(S,T,f,\leq I)$. Otherwise, the range constraint is binding for μ', the solution at the end of STEP 3, and we consider each maximal block of μ'.

Note that $\mu(t) \prec \mu'(t)$ $(\mu'(t) \prec \mu(t))$ for destination t only if t is in the constrained maximal block of μ' containing t_1 $(t_{|T|})$. Therefore, when $[a,b]$ is an unconstrained maximal block of μ', it is also a maximal block of μ and $\mu'(t') \equiv \mu(t')$ for each $t' \in [a,b]$. By Lemma 4.24, the partial assignment $\mu"$, defined by $\mu"(t) \equiv \mu(t)$ for each $t \in [a,b]$ is optimal.

We now show that the three conditions of Lemma 4.25 are satisfied for the constrained maximal blocks $[t_1,t_i]$ and $[t_j,t_{|T|}]$ of μ'. First note that, if at any iteration of STEP 3, we had $\mu'(t_1) \equiv \mu(t_1)$, then, an additional left shift could have been decomposed into co secutive left shifts with respect to $[t_j,t_{|T|}]$ and $[t_1,t_i]$. Since $\mu'(t) \preceq \mu(t)$ for each destination t in these blocks, by Lemma 3.12, these left shifts would not have reduced costs. Therefore, $\mu(t) \preceq \mu'(t)$ for each $t \in [t_1,t_i]$. Also, in STEP 2 the assignment of block $[t_j,t_{|T|}]$ is not changed and in STEP 3 only left shifts are performed. Therefore, $\mu'(t) \preceq \mu(t)$ for each $t \in [t_j,t_{|T|}]$ and condition 1 of Lemma 4.25 is satisfied.

Note that at the end of STEP 2, if $[t_1,t']$ is a maximal block of μ', then t' is the highest indexed destination in a

maximal block of μ. Therefore, if there are no shifts in STEP 3, $t' \equiv t_i$ and condition 2 of Lemma 4.25 is satisfied. Otherwise, assume at least one left shift occurs in STEP 3. Each shift in STEP 3 is with respect to a block $[t_1, t_a]$ with t_a the highest indexed destination with $\mu(t_a) \prec \mu'(t_a)$ in the maximal block containing t_1. Therefore, after any shift, the highest indexed destination in the maximal block containing t_1 must be the highest indexed destination in a maximal block of μ. A similar argument shows that there is a maximal block $[t_j, b]$ of μ and condition 2 of Lemma 4.25 is satisfied.

We now consider the changes in cost from a left shift of μ' with respect to $[t_1, t_i]$. STEP 3 terminated when a left shift of μ' with respect to $[t_1, t_a]$, with t_a the highest indexed destination in the maximal block containing t_1 with $\mu(t_a) \prec \mu'(t_a)$, did not reduce costs. If $t_a \equiv t_i$, then a left shift of μ' with respect to $[t_1, t_i]$ does not reduce costs. Otherwise, $\mu'(t_{a+1}) \equiv \mu(t_{a+1})$, and $[t_{a+1}, t_i]$ is a maximal block of μ. Note that a left shift of μ' with respect to $[t_1, t_i]$ can be decomposed into consecutive left shifts with respect to $[t_1, t_a]$ and $[t_{a+1}, t_j]$. We are given that the first of these shifts does not reduce costs, and by Lemma 3.12, neither does the second. The argument that a right shift with respect to $[t_j, t_{|T|}]$ does not reduce costs is similar. Therefore, condition 3 of Lemma 4.25 is satisfied and there is no lower cost assignment of the maximal constrained blocks of μ'.

We have shown that there is no lower cost assignment for each maximal block of μ'. Hence, μ' is optimal. □

We now consider the computational complexity of these procedures.

4.5 **Computational Complexity Analysis**

We first examine the number of computational steps required to solve the problem (S,T,f,≤I) using RESTRICTED LEFT-SHIFT, then consider the number of steps required to solve the problem (S,T,f,H) with $f:R\rightarrow R$ convex and consistent.

RESTRICTED LEFT-SHIFT consists of two main steps, the first step using LEFT-SHIFT to generate an initial solution, which may not satisfy the range constraint. Then the procedure, through a series of right and left shifts, generates an optimal solution. This series of right shifts do not require cost calculations and no more than |S| right shifts are accomplished. Therefore, these right shifts take O(|S|) steps. Finally, there can be no more than |S| left shifts. If f:R→R is convex but not piecewise linear, then each shift requires no more than O(|T|) steps, and this entire procedure requires O(|S|log|S| + |S||T|) steps. Otherwise, when $f:R\rightarrow R$ is piecewise linear and convex, as was shown in Section 3.5, each shift requires O(1) steps. When the number of breakpoints can be bounded by a constant, this is an

$O(|S|\log|S|)$ procedure.

Note that the generation of the complementary problem on the line for (S,T,f,H) requires $O(|S|)$ steps and using equation (4.12) to specify the optimal assignment for (S,T,f,H) requires another $O(|S|)$ steps. Therefore, when $|S| = |T|$, we can solve this problem using BISECTION SEARCH in $O(|S|\log|S|)$ steps. Otherwise, using RESTRICTED LEFT-SHIFT takes $O(|S|\log|S| + |T||S|)$ steps for general convex functions f, or when f is piecewise linear with the number of breakpoints bounded by a constant, a total of $O(|S|\log|S|)$ steps are required.

APPENDIX A

THEORETICAL JUSTIFICATION FOR EVEN SPACING

In this appendix we show that if customer arrivals are completely random, then evenly spaced service times will minimize the average waiting time over all customers. To show this we now prove that evenly spaced service times will minimize the total expected waiting time for all customers.

As discussed in Beaumont (1983), completely random arrivals follow a Poisson process. Therefore, if λ is the rate of the Poisson process, then the number of arrivals in any interval of time of length t is Poisson distributed with mean λt and non-overlapping time increments are independent.

Our reasoning is based on the following well known results:

(1) Theorem: In a Poisson Process, under the condition that n events have occurred in an interval (0,t), the times at which events occur, considered as unordered random variables, are distributed independently and uniformly in the interval (0,t). (Ross, p. 224)

(2) If the waiting time of a customer is considered a random variable then the total wait time in an interval is the expectation of a random number of random variables. Letting N denote the number of customers that arrive in an interval and X_i the wait time for customer i with $E[X_i] = E[X]$ for each i, then

$$E[\ \sum_{i=1}^{N} X_i\ ] = E[N]*E[X].$$

(Ross, p. 96-97)

Given these results, note that the expected number of customers arriving in an interval of length t_i is $\lambda*t_i$ and the arrivals are uniformly distributed over this interval. Therefore, the expected wait is $t_i/2$ for any customer in this interval and the expected wait of all arrivals in a given interval is

$$\lambda*t_i^2/2$$

(Ross p. 230).

Therefore, if service times at an origin location are spaced t_i time units apart for i = 1,...,f, then the expected total waiting time of all customers is

$$\sum_{i=1,f} \lambda*t_i^2/2.$$

Therefore, the problem to minimize the expected total wait may be formulated as:

minimize $\sum_{i=1}^{f} \lambda*t_i^2/2$

s.t.

$$\sum_{i=1}^{f} t_i = H$$

$$t_i \geq 0$$

Letting $t_i = H/f - \epsilon_i$, where H is the length of the planning period and f is the number of services over that period, this problem is equivalent to:

minimize $\sum_{i=1}^{f} (H/f + \epsilon_i)^2$

s.t.

(A-1) $\sum_{i=1}^{f} \epsilon_i = 0$

(A-2) $\epsilon_i \leq (H/f)$

Note that this objective function equals:

$$\sum_{i=1}^{f} (H/f)^2 + \sum_{i=1}^{f} 2*(H/f)\epsilon_i + \sum_{i} \epsilon_i^2$$

The first term is a constant, and by (A-1), the second term must equal zero. Therefore, to minimize the objective function, the third term must be minimized.

Let $z = \sum_{i=1}^{f} \epsilon_i^2$. Note that $z \geq 0$ for any choice of ϵ_i terms. Since letting all of these terms equal zero is a feasible solution which achieves the lower bound, this is an optimal solution of this problem. Therefore, the expected total customer wait is minimized when all the intervals are of equal length.

APPENDIX B

GENERALIZATION OF THE DIRECTED LINEAR ARRANGEMENT PROBLEM

The Directed Optimal Linear Arrangement Problem, is defined in Garey and Johnson (1979) as follows:

INSTANCE: Directed Graph G = (V,A), and positive integer K,

QUESTION: Is there a one-to-one function $f:V\rightarrow\{1,2,\ldots,|V|\}$ s.t. $f(u) < f(v)$ for each arc (u,v) in A and such that: $\sum_{(u,v)\in A} (f(v) - f(u)) \leq K$?

Recall that the **mission disruption problem** is expressed as:

> Mission Disruption Problem: given a set of missions, minimize the cost of the mission disruption needed to provide evenly spaced services for each customer.

and was formulated in Chapter II as:

$$\text{Minimize} \quad \sum_j \sum_{i\in M_j} \text{COST}(MD_{ij}) \tag{1}$$

subject to:

$$IST_j + T_{ij} * PS_j = CS_{ij} \qquad \forall j;\ \forall i \in M_j \tag{2}$$

$$T_{ij} - T_{mj} \geq 1 \quad \text{or} \quad T_{mj} - T_{ij} \geq 1 \qquad \forall j;\ \forall i,m \in M_j \tag{3}$$

$$CS_{ik} - CS_{ij} = StdTIME_{jk} + MD_{ij} \qquad \forall j;\ \forall i \in M_j;\ k = A_{ij} \tag{4}$$

$$IST_j \geq 0 \qquad \forall j \tag{5}$$

$$MD_{ij} \text{ urs} \qquad \forall j;\ \forall i \in M_j \tag{6}$$

$$0 \leq T_{ij} \leq FREQ_j - 1 \text{ and integer} \quad \forall j;\ \forall i \in M_j \qquad (7)$$

We now show that every instance of a Directed Optimal Linear Arrangement Problem may be formulated and solved as an instance of the Mission Disruption Problem.

Given a directed graph G = (V,A), the set A can be partitioned into arc-disjoint paths (an easy partition would make each arc in A a path). Each path represents a mission, denoted $MISS_i$, with index i identifying different paths.

Each node v_j in V represents a different customer service of the same customer. Therefore, the frequency of service of this customer is |V|. To ensure that each service is assigned a time in the set {1,2,...,|V|} we set IST = 1, and PS = 1. Since we have a single customer, we do not have to use an index to identify different customers. However, we still must identify each customer service. Therefore, we label the nodes from 1 to |V|, letting CS_j denote the service time associated with node v_j. In this way we do not have to use another index to denote the multiple services of this single customer by each mission. Since IST = 1 and PS = 1, CS_j corresponds to the function f:V→{1,2,...,|V|} of the Directed Optimal Linear Arrangement Problem.

Let the set M contain each $MISS_i$, and A_{ij} denote the service following v_j on path (mission) i.

Finally, we let MD_{jk} denote the difference in times between nodes connected by an arc. To enforce the constraint that $CS_v > CS_u$ for each arc (u,v) and to let the objective function calculate the sum $\sum_{(u,v)\in A} (CS_v - CS_u)$, we define the cost function for this problem as:

$$COST(MD_{jk}) = \begin{cases} MD_{jk}, & \text{if } MD_{jk} \geq 0 \\ K+1, & \text{otherwise.} \end{cases}$$

For example, consider the graph $\{V,A\}$ with $V = \{v_1, v_2, v_3, v_4\}$ and $A = \{(v_1,v_2), (v_2,v_3), (v_2,v_4)\}$. We let $M = \{MISS_1 = v_1\text{-}v_2\text{-}v_3, MISS_2 = v_2\text{-}v_4\}$. Then $A_{1,1} = v_2$, $A_{2,1} = v_3$, and $A_{2,2} = v_4$. The formulation of this problem is:

Minimize $z = COST(MD_{1,1}) + COST(MD_{3,1}) + COST(MD_{2,2})$

s.t.

$$1 + T_1 = CS_1$$
$$1 + T_2 = CS_2$$
$$1 + T_3 = CS_3$$
$$1 + T_4 = CS_4$$

$$T_1 - T_2 \geq 1 \text{ or } T_2 - T_1 \geq 1$$
$$T_1 - T_3 \geq 1 \text{ or } T_3 - T_1 \geq 1$$
$$T_1 - T_4 \geq 1 \text{ or } T_4 - T_1 \geq 1$$
$$T_2 - T_3 \geq 1 \text{ or } T_3 - T_2 \geq 1$$
$$T_2 - T_4 \geq 1 \text{ or } T_4 - T_2 \geq 1$$
$$T_3 - T_4 \geq 1 \text{ or } T_4 - T_3 \geq 1$$

$$CS_2 - CS_1 = MD_{1,1}$$
$$CS_3 - CS_2 = MD_{2,1}$$
$$CS_4 - CS_2 = MD_{2,2}$$

$$0 \leq T_i \leq |V|-1, \text{ and integer, } \quad i=1,2,3,4$$

Now consider the solution of the Mission Disruption formulation of the Directed Linear Arrangement Problem. First note that constraints (2),(3) and (7) ensure that each node j is given a distinct label CS_j from the set $\{1,2,\ldots,|V|\}$. If the optimal objective function value z^* is less than a given K, then for each arc (u,v), $CS_v - CS_u > 0$. Also, if $z^* > K$, then there is no lower cost labeling of the nodes which satisfy the requirements of the Directed Linear Arrangement Problem.

Therefore, a solution procedure for the Mission Disruption problem also solves the Directed Linear Arrangement Problem.

APPENDIX C

MISCHED

This appendix contains a memo, provided by Captain Keith Ware, HQ AMC/XP, that describes the scheduling procedure MISCHED.

NOTES ON THE MISCHED PROGRAM

The MISCHED program takes a set of missions (a mission is a combination of a route and a plane type) and a set of driver locations, that is a subset of the sites those missions visit, and attempts to schedule the missions so that the minimum inter-visit time for each of the driver locations is maximized.

Assumptions:

1. 30 day months

2. Missions must be scheduled evenly, i.e., if a mission goes 6 times per month, then it will be scheduled at 5 day intervals.

3. Multiple visits to a driver location by a mission are "ignored. See comments in the algorithm.

Data Structures:

Parameters:

MXNAME: The maximum number of locations in the problem. Note, must be at least the number of sites visited by the routes.
MXACFT: The maximum number of aircraft types.
MXSTOP: The maximum number of stops on any route.
MXRTE: The maximum number of routes.

Data Structures:

NAME(MXNAME): Names of all locations in the problem.
ROUTE(MXRTE,MXSTOP): Stops for each route.
NUMSTOP(MXRTE): The number of stops for each route.
SERVICE(MXRTE, MXNAME): A route/location incidence matrix.

NEWRTS(MXRTE x MXACFT,2): The set of missions. (I know, bad naming convention) For each mission, the first element in the array is the route number, and the second is the acft type.
FREQ(MXRTE x MXACFT): The number of times a mission is to be scheduled each month.
STOPTIME(MXRTE x MXACFT, MXSTOP): The time a mission will arrive at each of its stops if it departs at time 0:00.

LOCSET(MXNAME): The set of locations which drive the scheduling.
RTESET(MXRTE x MXACFT): The set of missions to be scheduled.

Input Files:

BASES.DAT: The names of all sites in the system.
LOAD.DAT: The names of all driver locations.
GTIME.DAT: The groundtimes for each aircraft and stop code.
ROUTE.DAT: The routes in the problem. Each line represents a route and is in the format: name/stop code name/stop code .... In our work we use the 4 letter icao identifiers for locations and the following stop codes.

1 Mission origin
2 Onload
3 Offload
4 Enroute stop with no crew rest
5 Enroute stop with crew rest
9 Mission termination

Thus a mission which originated and onloaded at Charleston then flew to Ramstein and back with a crew rest at Mildenhall each way would be represented by the line:

KCHS1 KCHS2 EGUN5 EDAR3 EGUN5 KCHS9

Algorithm:

1. For each mission *r* in RTESET with only 3 stops.
   Let *l* = ROUTE(*r*,2)
   Schedule *r* using *l* as the driver location
   Delete *r* from RTESET
   If there are no other routes in RTESET which service *l*, delete *l* from LOCSET.

2. Compute the max inter-visit time for each location in LOCSET.

3. If LOCSET $\neq \{\}$, then go to 4, else go to 9

4. Choose the location *l* in LOCSET that is visited the least by the routes remaining in RTESET. Break ties by choosing the location with the largest max inter-visit time.

NOTE: Only the first visit by a mission to any location will be counted toward the remaining visit frequency for that location. For example, using the mission above. If the frequency of that mission is 10 per month, then it will contribute 10 to the remaining visit frequency for KCHS, EGUN and EDAR.

5. Choose the mission *r* servicing location *l* with the largest frequency. If none, go to 8.

6. Schedule *r*, using *l* as the driver location, such that the max inter-visit interval for *l* is split.

NOTE: If *l* is visited more than once by *r*, use the first visit to *l* to drive the scheduling.

7. Recompute the max inter-visit time for each location served by mission *r*, then delete *r* from RTESET. Go to 5.

8. Delete all locations from LOCSET which are now covered, i.e. there is no mission in RTESET which visits them. Go to 3.

9. Print out the schedule.

REFERENCES

Ahuja, Magnanti, and Orlin (1988), "Network Flows", MIT Operations Research Center Working Paper, OR 185-88.

Air Force Pamphlet (AFP) 76-2: "Military Airlift: Airlift Planning Factors", 29 May 1987.

Ball (1988), "Allocation/Routing: Models and Algorithms", Vehicle Routing: Methods and Studies, Golden and Assad (ed.), Elsevier Science Publishers.

Barnes, and Hoffman (1982), " Partitioning, Spectra and Linear Programming", Proc. Silver Jubilee Conference on Combinatorics, Univ. Waterloo, Waterloo, Ontario, Canada.

Barnes and Hoffman (1985), "On Transportation Problems with Upper Bounds on Leading Rectangles", SIAM J. Alg. Meth., vol 6, No. 3.

Bartholdi, Cullen, and Lofgren (1987), "The Periodic Delivery Problem", Computer Aided Planning and Scheduling, Inc. Report.

Bartholdi and Platzman (1988), "Heuristics Based on Spacefilling Curves for Combinatorial Problems in Euclidean Space", Mgt. Sci., Vol. 34, #3.

Beaumont, G. P. (1983), Introductory Applied Probability, John Wiley and Sons.

Beltrami and Bodin (1974), "Networks and Vehicle Routing for Municipal Waste Collection", Networks, Vol. 4, pp. 65-69.

Benders (1962), "Partitioning Procedure for Solving Mixed Variable Problems", Numerische Mathematik, Vol. 4, pp. 238-252.

Bodin, Golden, Assad, and Ball (1983), "Routing and Scheduling of Vehicles and Crews: The State of the Art", Computers and O.R., vol. 10, #1, pp. 63-211.

Bodin and Sexton (1986), "The Multi-Vehicle Subscriber Dial-a-Ride Problem", TIMS Studies Mgt. Sci., vol. 22, pp. 73-86.

Borsi, CAPT David R. (1993), Personal interview. Air Mobility Command, McGuire AFB, NJ, 1 August 1993.

Burdyuk and Trofimov (1976), "Generalization of the Results of Gilmore and Gomory on the Solution of the Traveling Salesman Problem", Izv.Akad.Nauk SSR, Tech.Kibernet.3, 16-22; translated as Eng. Cybernetics 14, 12-18.

Burkard (1985), "Assignment Problems: Recent Solution Methods and Applications", Lecture Notes in Control and Information Sciences: #84, System Modelling and Optimization.

Burkard and Derigs (1980), "Assignment and Matching Problems: Solution Methods with FORTRAN-Programs", Lecture Notes in Economics and Mathematical Systems #184, Beckmann and Kunzi ed., Springer-Verlag.

"Bury Airlift Bickering", Defense News, vol. 8 #15, April 19-25, 1993, p. 18.

Christofides (1976), "Worst Case Analysis of a New Heuristic for the Traveling Salesman Problem", Management Sciences Res. Rep. No. 388, Carnegie-Mellon University.

Christofides and Beasley (1984), "The Period Routing Problem", Networks, vol. 14, pp. 237-256.

Christofides and Eilon (1969), "Expected Distances in Distribution Problems", Operations Research, vol. 20, pp. 437-443.

Clarke and Wright (1964), "Scheduling of Vehicles from a Central Depot to a Number of Delivery Points", Operations Research, vol. 12, pp. 568-581.

Cullen, Jarvis, and Ratliff (1981), "Set Partitioning Based Heuristics for Interactive Routing", Networks, vol. 11, pp. 125-143.

Deineko and Filonenko (1979), "On the Reconstruction of Specially Structured Matrices", Aktualnye Problemy EVM; Programmirovanie. Dnepropetrovsk: DGU, 43-45.

Desrosiers, Dumas and Soumis (1986a), "A Dynamic Programming Solution of the Large-Scale Single Vehicle Dial-a-Ride Problem with Time Windows", Am. J. Math. Mgt. Sci., vol. 6, #3 and 4, pp. 301-325.

Desrosiers, Dumas and Soumis (1986b), "The Multiple Vehicle Many-to-Many Routing Problem with Time Windows", Ecole de Hautes Commerciales de Montreal, Cahiers du GERAD G-84-13.

Dumas (1985), "Confection d'Itineraires de Vehicules en Vue du Transport de Plusieurs Origines a Plusieurs Destinations", Master's Thesis, Publication 434 (in French), Centre de Recherche sur les Transports, Universite de Montreal.

Dumas and Desrosiers (1986), "A Shortest Path Problem for Vehicle Routing with Pick-up, Delivery and Time Windows", Ecole des Houtes Etudes Commerciales de Montreal, Cahiers du GERAD G-86-09.

Fisher (1988), "Lagrangian Optimization Algorithms for Vehicle Routing Problems", Operational Research '87, G.K. Rand (ed.), Elsevier Science Publishers, pp. 635-649.

Fisher and Jaikumar (1981), "A Generalized Assignment Heuristic for Vehicle Routing", Networks, vol. 11, pp. 109-124.

Garey and Johnson (1979), Computers and Intractability: A Guide to the Theory of NP-Completeness, W.H. Freeman and Company.

Gilmore, and Gomory (1964), "Sequencing a One State-Variable Machine: A Solvable Case of the Traveling Salesman Problem", Operations Research, vol 12, #5.

Hax, and Candea (1984), Production and Inventory Management, Prentice-Hall, Inc., 1984.

Held and Karp (1962), "A Dynamic Programming Approach to Sequencing Problems", J. SIAM, vol. 10, pp. 196-210.

Hoffman (1961), "On Simple Linear Programming Problems", Convexity, Proc. Symposia in Pure Mathematics, Vol 7, American Mathematical Society, Providence, RI.

Holt, Modigliani, Muth and Simon (1960), Planning Production, Inventories, and Work Force, Prentice-Hall, Inc..

Jacobs (1987), "The Vehicle Routing Problem with Backhauls", PhD. Dissertation, Georgia Institute of Technology.

Jaw, Odoni, Psaraftis and Wilson (1986), "A Heuristic Algorithm for the Multi-vehicle Advance Request Dial-a-Ride Problem with Time Windows", Trans. Res. B, vol. 20B, #3, pp. 243-257.

Johnson and Montgomery (1974), Operations Research in Production Planning, Scheduling, and Inventory Control, John Wiley and Sons.

Kalantari, Hill, and Arora (1985), "An Algorithm for the Traveling Salesman Problem with Pickup and Delivery Customers", European Journal of O.R., vol. 22, pp. 377-386.

Karp and Li (1975), "Two Special Cases of the Assignment Problem", Discrete Mathematics, 13.

Kearney, A.T. Inc. (1984), "Measuring and Improving Productivity in Physical Distribution", A report prepared for the National Council of Physical Distribution Management, NCPCM, Oak Brook, Il.

Kirca (1983), "Models and Procedures for the Pick-up and Delivery Problem", PhD. Thesis, Georgia Institute of Technology.

Kraemer, Davis, and Hilliard (1991), "A New Computer System Meets MAC's Airlift Scheduling Challenge", Airlift, Summer 1991.

Lin (1965), "Computer Solutions of the Traveling Salesman Problem", Bell System Technical Journal, 44, pp. 2245-2269.

Lin and Kernighan (1973), "An Effective Heuristic for the Traveling Salesman Problem", Operations Research, vol. 21, #2, pp. 498-516.

"Military Airlift: Airlift Planning Factors", Air Force Pamphlet (AFP) 76-2, 29 May 1987.

McMahon (1971), Aircraft Propulsion, Barnes and Noble Books.

Modigliani and Hohn (1955), "Production Planning Over Time and the Nature of the Expectation and Planning Horizon", Econometrics, Vol. 23, No. 1, pp. 46-66.

Psaraftis (1980), "A Dynamic Programming Solution to the Single Vehicle Many-to-Many Immediate Request Dial-a-Ride Problem", Tr. Sci., vol. 14, #2, pp. 130-154.

Psaraftis (1983a), "An Exact Algorithm for the Single Vehicle Many-to-Many Dial-a-Ride Problem with Time Windows", Tr. Sci., vol. 17, #3, pp. 351-357.

Psaraftis (1983b), "Analysis for an O(N^2) Heuristic for the Single Vehicle Many-to-Many Euclidean Dial-a-Ride Problem", Trans. Res. B, vol. 17B, #2, pp. 133-145.

Ross, Sheldon M. (1989), Introduction to Probability Models, 4th ed., Academic Press.

Russell (1977), "An Effective Heuristic for the M-Tour Traveling Salesman Problem with Some Side Conditions", Operations Research, vol. 21, #3, pp. 517-524.

Russell and Gribbin (1991), "A Multiphase Approach to the Period Routing Problem", Networks, Vol. 21, pp. 747-765.

Russell and Igo (1979), "An Assignment Routing Problem", Networks, vol. 9, pp. 1-17.

Sexton and Choi (1986), "Pickup and Delivery of Partial Loads with 'Soft' Time Windows", Am. J. Math. Mgt. Sci., vol. 6, #3 and #4, pp. 369-398.

Shepherd (1990), "Peacetime Airlift: Job #1, Too!", Defense Transportation Journal, vol. 46, #5, pp. 11-20.

Solomon and Desrosiers (1988), "Time Window Constrained Routing and Scheduling Problems", Tr. Sci., vol. 22, #1, pp. 1-13.

Stein (1978), "An Asymptotic Probabilistic Analysis of a Routing Problem", Math. O.R., vol. 3, pp. 89-101.

Tan and Beasley (1984), "A Heuristic Algorithm for the Period Vehicle Routing Problem", Omega, vol. 12, #5, pp. 497-504.

Ware (1991), Memo to Capt Borsi, from HQ AMC/XP.

VITA

John J. Borsi [redacted] He graduated from the U.S. Air Force Academy with a Bachelor of Science in Engineering Sciences in May 1980. Upon graduation, he received a regular commission in the United States Air Force and was assigned to Edwards Air Force Base, California as a structural flight test engineer. In 1982, he was reassigned to Cornell University, where he received a Master of Engineering degree in Operations Research and Industrial Engineering in May 1983. From 1983 to 1986 he conducted space system logistics analysis for the Air Force Operational Test and Evaluation Center at Kirtland Air Force Base, Albuquerque New Mexico. In 1986, he entered the Ph.D. program in Operations Research at the Georgia Institute of Technology. From 1990 to the present, he has taught classes and conducted research as an instructor in the Department of Operational Sciences, Air Force Institute of Technology. In 1994, he completed all requirements for the Doctor of Philosophy degree.

[redacted]

28 Jan 94