FLOW BASED HEURISTICS FOR THE MULTIPLE TRAVELING SALESMAN PROBLEM WITH TIME WINDOWS

MYUNG SUB LEE
KOREA MANAGEMENT & CREDIT RATING CORP.

ABSTRACT

In this paper, new algorithms for solving the multiple traveling salesman problem with time windows are presented. These algorithms are based on the flow based algorithms for solving the vehicle scheduling problem. Computational results on problems up to 750 customers indicate that these algorithms produce superior results to existing heuristic algorithms for solving the vehicle routing problems when the time windows are 'tight enough' where 'tight enough' is based on a metric proposed by Desrosiers et al.(1987).

1.0. INTRODUCTION

The Multiple Traveling Salesman Problem with Time Windows (m-TSPTW) involves the determination of a minimum cost set of routes for a fleet of vehicles that service a set of customers with known demands within their specific time windows. Each route begins and ends at the same depot. Furthermore, each customer is serviced exactly once by a vehicle and all the customers are assigned to vehicles such that the time window constraints are not violated. The vehicles have unlimited capacity and there is no upper bound on the length of the routes.

$[TL_i, TU_i]$ is the time window for customer i where TL_i and TU_i are the earliest possible time and the latest possible time for beginning the service of customer i, respectively. In the m-TSPTW, both the lower and upper bounds on the time windows are hard so that a vehicle cannot begin to service a customer either before the lower bound nor after the upper bound of the given time windows of the customers. WT_{ij}, the delay in time to begin the service at customer j, when customer j is immediately serviced after customer i by the same vehicle, is defined to be the following:

$$WT_{ij} = \text{Max } \{0,\ TL_j - \text{Arrival Time at customer } j\}. \quad (1.1)$$

Furthermore, the time window width at customer i is defined as follows:

$$\text{Width}_i = TU_i - TL_i \quad (1.2)$$

There are two special boundary cases of the m-TSPTW.

Case 1: When $TL_i = -\infty$ and $TU_i = \infty$ for each customer i, then the problem is equivalent to the single depot Multiple Travelling Salesman Problem(m-TSP). there exist many algorithms for solving the m-TSP, which is NP-hard. Most of these algorithms are based on procedures to solve either the traveling salesman problem(TSP) or the Vehicle Routing Problem(VRP).

Case 2 : When $TL_i = TU_i$ for each customer i, then this problem is equivalent to the Vehicle Scheduling Problem(VSP). The VSP is solved optimally in polynomial time as a Minimum Cost Flow Problem(MCFP). Algorithms for solving the VSP are further discussed in Section 3.

Traditionally, the m-TSPTW is solved as a VRP with time windows(VRPTW) regardless of the time window width. In this paper, we attempt to adapt the VSP algorithms to solve the m-TSPTW. The following questions are addressed in this paper:

1. How should the VSP algorithms be modified to produce effective results to the m-TSPTW?
2. For what time window width will the modified VSP algorithms produce better solutions than the VRP algorithms?

A key factor in this research is connected with the definition of the tightness of the time windows. It is expected that the modified VSP algorithms will give better solutions to the m-TSPTW than the VRP algorithms when the time window width are small. Desrochers M. et. al. (1987) offers some definitions for the tightness of the VRPTW.

The typical applications of the m-TSPTW are the scheduling of buses for mass transit systems and the scheduling of school buses.

2.0. The m-TSPTW PROBLEM

A key feature of the m-TSPTW problem is the m-TSPTW Network. The construction of this network and the mathematical programming formulation of the m-TSPTW are now described.

2.1. The m-TSPTW Network

V is the set of vertices and A is the set of arcs in a network G = (V,A). In G, we have V = {s,t,N}, where s and t are the supersource and supersink nodes and represent the same single depot in the m-TSPTW. A node in the node set N represents a customer to be serviced. Each customer $i \in N$, has a time window $[TL_i, TU_i]$ and a service time ST_i which is the time required to service customer i. There is no restriction on the time windows of nodes s and t since the length of the route of any vehicle is assumed to be unrestricted.

A potential arc from customer i to customer j in the arc set A can be classified as follows:

Feasible Arc : An arc (i,j), $(i,j) \in A$, is called a Feasible Arc if the following condition is satisfied:

$$TT_{ij} \le TL_j - (TU_i + ST_i) \qquad (2.1)$$

A feasible arc is always feasible on any route.

Sometimes Feasible Arc : An arc (i,j), $(i,j) \in A$, is called a Sometimes Feasible Arc if the following condition is satisfied:

$$TL_j - (TU_i + ST_i) < TT_{ij} \le TU_j - (TL_i + ST_i) \qquad (2.2)$$

If arc (i,j) is on a path and arc (i,j) is a Sometimes

Feasible Arc, then a (s-t) path in G containing arc (i,j) may or may not represent a feasible schedule for a vehicle.
Infeasible Arc : An arc (i,j) is called an Infeasible Arc if the following condition holds:

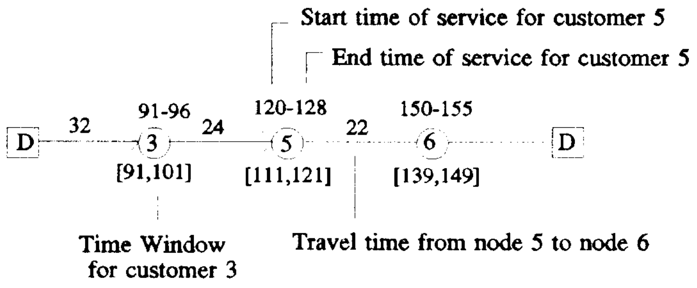
$$TT_{ij} > TU_j - (TL_i + ST_i) \quad (2.3)$$

If this arc were to be included in A and this arc existed on an (s-t) path, then this path would always be infeasible. Infeasible Arcs are not included in the network G.

Besides the arcs between customers, G contains an arc (s,i) from the supersource s to each node i, i ∈ N, and an arc (i,t) from each node i, i ∈ N, to the supersink t. Since there are no time window restrictions on these arcs, these arcs are always feasible in the solution to the m-TSPTW.

The solutions found to the m-TSPTW by applying the VSP algorithms can be infeasible in that the routes and schedules in the solution may not satisfy all of the time window constraints. Every path in the m-TSPTW Network from the depot at the beginning of the day to the depot at the end of the day is a feasible route if all of the arcs in the path are Feasible Arcs. A path containing customer i is an infeasible route if the vehicle arrival time at customer i is greater than TU_i.

For example, suppose that we have a path including two Sometimes Feasible Arcs as follows:


Start time of service for customer 5
End time of service for customer 5
91-96
120-128
150-155
D
32
3
24
5
22
6
D
[91,101]
[111,121]
[139,149]
Time Window for customer 3
Travel time from node 5 to node 6

Arc (3,5) and arc (5,6) are Sometimes Feasible Arcs since the travel time on each arc satisfies (2.2). The vehicle departs customer 5 at the time of 128 units, travels the arc (5,6) for 22 units of time and then arrives at customer 6 at the time of 150. This route is infeasible since the arrival time at customer 6 violates the time window of the customer, [139,149]. If all of the routes in the obtained solution by adapting the VSP algorithms are feasible, then the solution is feasible.

The fleet size can be unrestricted or an upper bound on the fleet size can be specified. If the fleet size is unrestricted, the fleet size is determined simultaneously with the best set of routes and schedules rather than being fixed a priori.

The primary objective is to minimize the total travel time of the vehicles. Secondary considerations in the m-TSPTW are minimizing total vehicle wait time and the number of vehicles required. Given a set of vehicle schedules, the total vehicle wait time is the sum of the delay times at all of the customers. In Section 5.2, metrics for comparing solutions, which combine total vehicle travel time and wait time, are proposed.

2.2. Mathematical Programming Formulation of the m-TSPTW

The decision variables in the mathematical programming formulation of the m-TSPTW are the following:

(1) X_{ij}, $(i,j) \in A$, is the flow variable which equals 1 if the arc (i,j) exists on an (s,t) path and equals 0, otherwise.

(2) T_i is the starting time to service customer i, $i \in N$.

The mathematical programming formulation of the m-TSPTW is the following:

$$\text{Min } Z = \sum_{(i,j)\in A} C_{ij}X_{ij} \qquad (2.4)$$

subject to

$$\sum_{i:(i,j)\in A} X_{ij} = 1 \qquad \text{for } j \in N, \qquad (2.5)$$

$$\sum_{i:(i,j)\in A} X_{ij} - \sum_{i:(j,i)\in A} X_{ji} = 0 \qquad \text{for } j \in N, \qquad (2.6)$$

$$T_i + ST_i + TT_{ij} - T_j \leq (1-X_{ij})M_{ij} \qquad \text{for } i,j \in N \qquad (2.7)$$

$$TL_i \leq T_i \leq TU_i \qquad \text{for } i \in N, \qquad (2.8)$$

$$X_{ij} \in \{0,1\} \qquad \text{for } (i,j) \in A, \qquad (2.9)$$

where $M_{ij} \geq TU_i + ST_i + TT_{ij} - TL_j$.

In this formulation, the objective function (2.4) represents the total travel cost in the resulting vehicle schedules. Constraints (2.5) express the fact that every customer must be serviced exactly once by a vehicle. Constraints (2.6) express the usual network flow conservation conditions. Constraints (2.7) and (2.8) ensure the feasibility of the vehicle schedules. Constraints (2.9) ensure that each arc is on at most one vehicle schedule.

Constraints (2.5),(2.6) and (2.9) with the objective function (2.4) is a single commodity MCFP since each route begins and ends at the same depot. The algorithm for solving the MCFP gives an integer solution. This is the basis for the minimum cost flow formulation of the VSP described in Section 3. Constraints (2.7) and (2.8) are needed for the Sometimes Feasible Arcs defined previously. Since solving the linear programming relaxation of (2.4) - (2.9) is not guaranteed to produce integer answers, the m-TSPTW has to be solved as an integer program.

2.3 Vehicle Routing Problems

There is a vast literature for solving the VRP. We refer the reader to Bodin et al. (1983) for an extensive survey of optimal and approximation methods for solving vehicle routing and scheduling problems. In Golden and Assad (1988), there are many new advanced papers on vehicle routing and scheduling. In Desrosiers et al. (1992), there is an extensive overview of the algorithms for time constrained routing and scheduling. Baker and Schaffer (1986), Bodin et al. (1979), Christofides et al. (1981a,1981b), Desrochers et al. (1987,1990), Desrosiers et al. (1984,1985, 1986, 1988), Kolen et al. (1987), Lin (1965), Lin et al. (1973), Solomon (1987), Solomon et al. (1988) and Swersey et al. (1984) give algorithms to solve the routing and scheduling problems with time windows. Column generation approaches in Desrochers et al. (1990) and Desrosiers et al. (1984) solve the m-TSP with tight time window very effectively when the problem size is small. As the problem size grows larger, the column generation approaches may not be preferred to our algorithms because the approaches have to generate large number of columns.

An effective sequential insertion approach for solving VRPTW in Solomon(1987) and the Baker and Schaffer's Branch Exchange Procedure (Baker and Schaffer (1986)) are compared with our algorithms in computation study in Section 5.

3.0 VEHICLE SCHEDULING PROBLEM AND ALGORITHMS

Our research has focused on adapting VSP algorithms to solve the m-TSPTW when the time window is tight. The VSP is to design a set of routes to service all of the trips while minimizing total travel cost. Each trip is serviced by exactly one vehicle. In the single depot vehicle scheduling problem, an acyclic network called the VSP Network is created. The VSP Network is transformed to a second network called the minimum cost flow problem network(MCFP Network) and a MCFP is solved over this network. The solution to the MCFP is a set of paths over the VSP Network. These paths partition the nodes(trips) in the VSP Network such that each node is on one path, and the total travel cost is minimized.

The VSP Network is a special case of the m-TSPTW Network where $TL_i = TU_i$ for each i. The VSP Network only contains Feasible Arcs.

3.1 Mathematical Programming Formulation of the VSP

We now present the mathematical formulation to the VSP which was presented in Bodin et al. (1983). The variables in this formulation are X_{ij} where $X_{ij} = 1$ if trip j is to follow trip i on a vehicle schedule and $X_{ij} = 0$ otherwise. With this variable definition, the VSP is formulated as a MCFP as follows:

$$\text{Min } Z = \sum_{(i,j)\in A} C_{ij}X_{ij} \tag{3.1}$$
subject to
$$\sum_{i:(i,j)\in A} X_{ij} - \sum_{i:(j,i)\in A} X_{ji} = 0 \qquad \forall\ j \in N \tag{3.2}$$
$$\sum_{i:(i,j)\in A} X_{ij} = 1 \qquad \forall\ j \in N \tag{3.3}$$
$$X_{ij} \in \{0,1\} \qquad \forall\ (i,j) \in A \tag{3.4}$$

Constraints (3.2) are the standard conservation flows and constraints (3.3) ensure that exactly one path (vehicle schedule) covers each node. In this formulation, each path is a vehicle schedule and there is no restriction on the number of paths that can be found.

3.2. Transformation of the VSP Network

In order to transform the VSP into the MCFP, the VSP

Network, $G = (V,A)$, is expanded into a larger network, $G' = (V',A')$, as follows:

1. Every node i in G is replaced by a starting node SN_i and an ending node EN_i, in the expanded graph G'.
2. There is an arc drawn from SN_i to EN_i with an arc cost $C_{ii} = 0$. Both the lower and upper bound on flow on this arc are one to ensure that each customer has to be visited exactly once by a vehicle.
3. Each arc (i,j), $(i,j) \in A$, is drawn in G' from EN_i to SN_j. Furthermore there are arcs from the supersource s to each node SN_i and each node EN_i to the supersink t. The arc costs are as defined in G except that the capital cost FC is not included on the arcs (s,SN_i).
4. An arc from t to s is added. This arc is called reverse arc and denoted as (L,U,FC) where L and U represent the lower bound and the upper bound on flow, respectively and the arc cost, FC, is equal to the capital cost of a vehicle. If we have no restriction on the fleet size, then the reverse arc would be denoted by (0,U,0) where U is a large constant.

The MCFP formulation of the VSP gives an optimal solution to the VSP and every path from the depot to the depot is always feasible since the MCFP Network is acyclic. The only difference between the VSP and the m-TSPTW is that there are no Sometimes Feasible Arcs in the VSP Network.

Bertsekas and Tseng (1988), Aashtiani and Magnanti (1976), and Bradley, et al. (1977) provide some useful key algorithms for the MCFP. We used Bertsekas and Tseng's relaxation code RELAXT-II in this research.

4.0. ALGORITHMS FOR SOLVING THE m-TSPTW

We now develop some heuristics for solving the m-TSPTW based on the VSP algorithms and a procedure for finding a lower bound on the total travel time(cost). A feasible solution to the m-TSPTW is a set of paths where the time to begin service on each customer i on each path lies between TL_i and TU_i. We call the combined set of Feasible Arcs and Sometimes Feasible Arcs, the set of Possible Arcs.

Our algorithms for solving the m-TSPTW consists of two steps. The first step is to construct the initial solution by using the Flow Based Route Construction Algorithm (FBRCA) described in Section 4.2. The second step improves the obtained initial solution using the route improvement heuristic, FIMP.

4.1. Lower Bound Solution

If only the Feasible Arcs are included in the m-TSPTW Network, then the solution will be always feasible but it may be suboptimal. If we include all of the Possible Arcs in the m-TSPTW Network and its solution is feasible, then the optimal solution has been found. However, if the solution with all Possible Arcs is not feasible, the value of the solution is a lower bound solution to the optimal solution of the m-TSPTW.

4.2. Flow Based Route Construction Algorithm (FBRCA)

As noted above, if only Feasible Arcs are used in solving the MCFP formulation of the m-TSPTW, then a feasible but generally suboptimal solution to the m-TSPTW is found. If some Sometimes Feasible Arcs are included in the formulation of the MCFP and this solution is feasible to the m-TSPTW, then this solution is no worse than the solution with only Feasible Arcs. However, it is possible that too many Sometimes Feasible Arcs are added to the MCFP formulation so that the solution to the MCFP is infeasible to the m-TSPTW. The problem is to determine how many Sometimes Feasible Arcs should be added to the MCFP formulation in order to find a feasible solution with a minimum total travel cost to the m-TSPTW. This method to solve the m-TSPTW is called the FLOW Based Route Construction Algorithm (FBRCA). A key to the FBRCA is the step size.

4.2.1. Step Size

Let arc (i,j) be a Sometimes Feasible Arc. The Infeasible Time, IT_{ij}, is defined as follows:

$$IT_{ij} = TT_{ij} - (TL_j - TU_i - ST_i). \quad (4.1)$$

We believe that as the value of IT_{ij} gets larger, it becomes more likely that an (s-t) path containing this arc is infeasible to the m-TSPTW. An arc (i,j) will exist in the MCFP Network on iteration k of the FBRCA if $IT_{ij} \leq STEP_k$ where $STEP_k$ is called the step size of iteration k. If the time window width for each customer is identical, then $STEP_k$ is between zero and twice the time window width. When the time windows are not identical, $STEP_k$ is between zero and two times the width of the widest time window.

4.2.2. The FBRCA Algorithm

Step 1 :	Set the initial step size equal to 2 × Width and form the initial MCFP Network which contains all of the possible arcs of the m-TSPTW.
Step 2 :	Solve the MCFP.
Step 3 :	Check if the solution at Step 2 is feasible or not. If it is feasible, the solution is optimal. Then, terminate the procedure. Otherwise, set LOW = 0 and HIGH = 2×Width.
	(At the kth iteration for $k \geq 2$)
Step 3 :	Set $STEP_k = \lceil (LOW + HIGH)/2 \rceil$, where $\lceil x \rceil$ means the next integer greater than x if x is not integer and equals x otherwise.
Step 4 :	Construct the new MCFP Network including all of the feasible arcs and some Sometimes Feasible arcs satisfying $IT_{ij} \leq STEP_k$.
Step 5 :	Solve the MCFP.
Step 6 :	Check if the solution at Step 5 is feasible or not. (1) If it is feasible, then set LOW = $STEP_k$ and check if the difference between LOW and HIGH is equal to 1 or not. If it is, then terminate the procedure since the current solution is the best feasible solution to the m-TSPTW. Otherwise, set k

= k +1 and go to Step 3. (2) If the solution is infeasible, then set HIGH = $STEP_k$. If the difference between LOW and HIGH is equal to 1, then retrieve the best feasible solution generated in the previous iteration and go to Step 7. If not, set k = k +1, and go to Step 3.

Step 7 : Let the number of vehicles of the best feasible solution be NVEH.

Step 8: Set the upper bound of the reverse arc, $U \leq$ NVEH-1, on the current MCFP Network, i.e., U = NVEH -1.

Step 9: Solve the MCFP with the changed U on the reverse arc.

Step 10: Check if the new solution is feasible or not. If it is feasible, then set NVEH = NVEH - 1 and go to Step 9. Otherwise, retrieve the feasible solution requiring the fewest number of vehicles from the previous iteration and terminate the procedure.

The FBRCA can search for the minimum number of vehicles as well as the minimum total travel time. We can reduce the fleet size of the current best feasible solution by setting the upper bound of the reverse arc of the MCFP Network to be smaller by one than the number of vehicles found on the current best feasible solution and repeat the FBRCA. This procedure can be repeated until the fleet size is set so small that no feasible solution to the MCFP exists.

4.3. Route Improvement Heuristic Algorithms

Given an initial solution to the m-TSPTW, the route improvement algorithm operates as follows: Select two or three routes from the existing solution and solve a m-TSPTW consisting of only customers on these selected subset of routes by using the FBRCA. If 2 routes are selected, this improvement procedure is called a 2-route combine procedure, and if 3 routes are selected, this procedure is called a 3-route combine procedure. When the 3-route combine procedure is applied, some three routes from the given initial solution are combined to make a subset of customers and the FBRCA is applied to this subset of customers. If the solution found from this application is better than the old one, then the old 3 routes are replaced by the new 3 routes. This procedure is repeated until all of the combinations to select 3 routes is carried out. These route combine procedures are repeated until no further improvements are found. We call this improvement solution algorithm FIMP. Details on this algorithm is presented in Lee (1992).

5.0. COMPUTATION STUDY

The three test problems are randomly generated for the computation study of our algorithms. The characteristics of the test problems are given as follows: (1) the location of the customers are generated from a uniform distribution, (2) The travel time between two customers is a linear function of the Euclidean distance between their locations, (3) TL_i is randomly generated according to a uniform distribution, (4) TU_i is then set equal to TL_i + Width where Width is a constant number for each test problem and (5) No Possible Arc longer than 60 minutes is allowed in the associated network.

For each test problem, we executed 2 initial solution algorithms and 2 improvement solution algorithms for each initial solution. We used the following metrics to compare these 6 solutions: (1) Total travel time (TT), (2) $CT_1 = (0.75)\times TT + (0.33)\times WT$, (3) $CT_2 = (0.75)\times TT + (0.50)\times WT$ where WT is total wait time in the solution and (4) Number of vehicles. CT_1 and CT_2 attempt to capture operating cost by weighing both travel time and wait time.

5.1. Computational Results

Computer programs written in FORTRAN were developed to implement our FBRCA and FIMP and executed on an IBM 3081/D computing system. The FORTRAN codes of the VRP algorithms, INS1 and BIMP, were provided by Dr.Edward K. Baker and slightly modified to test our randomly generated problems. INS1 is the implementation of a VRPTW for finding an initial solution to a vehicle routing problem and BIMP is the implementation of a VRPTW for finding the improved solution from an initial solution to a VRP.

In the ensuing tables, we report more than one best solution when one best solution has less total travel time or less total operating cost than another best solution but requires more vehicles.
The number of vehicles required for each best solution is marked in a parenthesis next to the corresponding solution. RT_1 in Desrochers M. et. al. (1987) for the tightness of the VRPTW is the ratio between the average time window width and the average travel time between customers.

The computational results on the test problems are reported in Table 5.1 through 5.4. The problems are called P250, P500 and P750 according to the number of customers in the problem. In Table 5.1, most of the best solutions in terms of total travel time have been found by using the FIMP improvement procedure with the given initial solutions. An optimal solution has been obtained where the time window width was 5 minutes. With respect to the two operating costs, the best solutions have been also found by using the FIMP improvement procedures in the all of the time window cases. The best solutions in terms of minimum number of vehicles have been found by the FBRCA or FIMP improvement procedures when the time windows are tight. If the time windows become wide, the best solution with minimum number of vehicles can be obtained by using the INS1 initialization procedures.

In Tables 5.3 and 5.4, we have found that the FIMP is superior to the other improvement heuristic to find the minimal travel time solutions regardless of the time window width. These tables also show the consistency of our conclusion that the FIMP is superior to the BIMP to obtain the best feasible solution when the time windows are tight such as 5 or 10 minutes. However, the INS1 is always better than the FBRCA to find an initial solution.

Table 5.1. Computational Results of Algorithms on P250

ALG	Width=1, RT_1=0.026					Width=5, RT_1=0.132				
	TT	WT	RT	CT_1	CT_2	TT	WT	RT	CT_1	CT_2
LB	6906	6980	31	7503	8669	6753	6607	30	7264	8368
	6851	7422	32	7609	8993	6683	6929	31	7319	8476
FBRCA	7068	7133	32	7676	8867	6888	6188	29	7226	8260
FIMP	6977	7242	32	7644	8853	6888	6188	29	7226	8260
BIMP	7037	7164	32	7663	8859	6888	6188	29	7226	8260
INS1	8836	4564	34	8146	8909	8887	3342	32	7778	8336
FIMP	6994	6883	32	7537	8687	6780	6722	31	7328	8446
BIMP	7203	7241	33	7813	9022	7054	6137	31	7334	8359
ALG	Width=10, RT_1=0.263					Width=15, RT_1=0.394				
	TT	WT	RT	CT_1	CT_2	TT	WT	RT	CT_1	CT_2
LB	6196	5544	26	6493	7419	5821	4979	24	6023	6855
	6159	5912	27	6587	7575	5817	5500	26	6194	7112
FBRCA	6662	5754	28	6912	7873	6568	5339	27	6703	7595
FIMP	6321	5824	28	6680	7652	5933	5235	26	6193	7067
BIMP	6378	5799	28	6714	7683	6121	5392	27	6386	7286
INS1	7780	2937	27	6813	7304	7608	1941	24	6352	6676
FIMP	6302	5378	27	6517	7415	6098	4289	24	6001	6718
BIMP	6510	5263	27	6635	7514	6187	4252	24	6056	6766

Table 5.2. Summary of Comparison of Algorithms on P250

COST	Width=1, RT_1=0.026	Width=5, RT_1=0.132
Minimize TT	FBRCA/FIMP(32)	INS1 /FIMP(31) FBRCA ALL (29)
Minimize CT_1	INS1 /FIMP(32)	FBRCA ALL (29)
Minimize CT_2	INS1 /FIMP(32)	FBRCA ALL (29)
No. of Vehicles	FBRCA ALL (32) INS1 /FIMP(32)	FBRCA ALL (29)
COST	**Width=10, RT_1=0.263**	**Width=15, RT_1=0.394**
Minimize TT	INS1 /FIMP(27)	FBRCA/FIMP(26) INS1 /FIMP(24)
Minimize CT_1	INS1 /FIMP(27)	INS1 /FIMP(24)
Minimize CT_2	INS1 /FIMP(27)	INS1 /FIMP(24)
No. of Vehicles	INS1 ALL (27)	INS1 ALL (24)

With respect to the operating costs, the FIMP is superior to the BIMP in both cases of the P500 problem while the BIMP is better for the P750 with 5 minute time window width and is also superior in terms of CT_2 in the 10 minute width case. From these analyses, we conclude that the FIMP is superior to the BIMP when the size of problem becomes smaller with the fixed time window width, or when the time windows become tighter with the fixed size of problem.

Table 5.3. Summary of Comparison of Algorithms on P500

COST	P500	
	Width=10 (RT_1=0.286)	Width=15 (RT_1=0.417)
Minimize TT	FBRCA/FIMP(35)	FBRCA/FIMP(35)
	INS1 /FIMP(33)	INS1/ FIMP(29)
Minimize CT_1	INS1 /FIMP(33)	INS1 /FIMP(29)
Minimize CT_2	INS1 /FIMP(33)	INS1 /FIMP(29)
No. of Vehicles	INS1 ALL (33)	INS1 ALL (29)

Table 5.4. Summary of Comparison of Algorithms on P750

COST	P750	
	Width=5 (RT_1=0.136)	Width=10 (RT_1=0.263)
Minimize TT	FBRCA/FIMP(65)	FBRCA/FIMP(62)
	INS1 /FIMP(63)	INS1 /FIMP(54)
Minimize CT_1	INS1 /BIMP(63)	INS1 /FIMP(54)
Minimize CT_2	INS1 /BIMP(63)	INS1 /BIMP(54)
No. of Vehicles	INS1 ALL (63)	INS1 ALL (54)

6.0. CONCLUSION AND FUTURE RESEARCH

The computational results indicate that the FIMP is very successful at improving an initial solution when the time windows are tight.

Our analyses of the computational experiments provide the following conclusions:

Main Conclusion :

(1) The FIMP works best for small values of Width regardless of initial solution algorithms given a fixed scheduling horizon, SH, and a fixed number of customers to be serviced.
(2) The FIMP works best when RT_1 is small. In the test cases studies, if $RT_1 \leq 0.5$, then in most cases, the FIMP is the preferred improvement procedure. In the cases where the FIMP did not provide the best solution, then its solution was generally close to the best. More tests are needed to further quantify this result. However, the FIMP appears to be a

relatively risk-free algorithm to use as an improvement procedure where $RT_1 \le 0.5$.

Conclusions (1) and (2) can be explained as follows: The FIMP works best when the density of customers is small. In this case, the tight time windows constrain the solution so that the temporal aspects of the problem play a pivotal role. As the density increases, the temporal aspects of the problem become less significant and the spatial aspects become more important and the routing algorithms tend to give a better solution.

(3) The FIMP absolutely finds the best solutions in terms of the total travel time over most of cases we tested.

The FIMP is a repetition of the same algorithm as the FBRCA except that, on each repetition, the FIMP is applied to a subset of the customers, which are the customers on a subset of the routes. In contrast, the INS1 and the BIMP are different algorithms. These results might suggest an approach for solving vehicle routing and scheduling problems. First, an algorithm is applied to the total set of customers in order to partition these customers into routes. This can be thought of an initialization procedure. Then, the same algorithm is repeatedly applied to the customers on a subset of the routes. If an improved solution is found, then these new routes replace the old routes. This second set can be thought of as an improvement procedure. This approach will probably be most effective when the problem is highly constrained.

Reference

Aashtiani, H.A. and Magnanti, T.L. (1976). Implementing primal-dual network flow algorithms, Oper. Res. Center Report 055-76, M.I.T.

Baker, E. and Schaffer, J. (1986). Solution Improvement Heuristics for the Vehicle Routing and Scheduling Problem with time Window constraints, American Journal of Mathematical and Management sciences, 6, 261-300

Bertsekas, D.P. and Tseng, P. (1988). The Relax Codes for Linear Minimum Cost Network Flow Problems, Annals of Operations research 13, 125-190

Bodin, L. and Berman L. (1979) Routing and Scheduling of School Buses by Computer, Transportation Science, 13, 113-129

Bodin, L., Golden, B., Assad, A., and Ball M. (1983). Routing and Scheduling of Vehicles and Crews: The state of the Art, Computer and Operations research, 10, 63-211.

Bradley, G.H., Brown, G.G. and Graves, G.W. (1977). Design and Implementation of large scale primal transshipment algorithms, Management Science, 24, 1-33

Christofides, N., Mingozzi, A. and Toth, P. (1981a). Exact Algorithms for the vehicle Routing Problem based on spanning Tree and Shortest Path Relaxations,Mathematical Programming, 20, 255-282

Christofides, N., Mingozzi, A. and Toth, P. (1981b). State

Space Relaxation Procedures for the Computation of Bounds to Routing Problems, Networks, 11, 145-161

Desrochers, M., Lenstra, J. K., Savelsbergh, M.W.P. and Soumis, F. (1987). Vehicle Routing with Time Windows: Optimization and Approximation. Vehicle Routing : Methods and studies (edited by Golden and Assad)

Desrochers, M.,Desrosiers, J. and Solomon M. (1990). A New Optimization Algorithm for the Vehicle Routing Problems with Time Windows, Les cahiers du GERAD, G-90-30, Montreal, Canada

Desrosiers, J., Soumis, F., Desrochers, M. and Sauve, M. (1984). Routing With Time Windows by Column Generation, Networks, 14, 545-565

Desrosiers, J., Soumis, F., Desrochers, M., and Sauve, M. (1985). Routing and Scheduling with Time Windows solved by Network Relaxation and Branch and Bound on time Variables, Computer Scheduling of Public Transport 2, 451-469, Rousseau, J. M.(ed.), North-Holland, Amsterdam

Desrosiers, J., Ferland, J., Rousseau, J., Lapalme, G. and Chapleau, L. (1986). TRANSCOL: A Multi-Period School Bus Routing and Scheduling System, TIMS Studies Mgmt. Sci. 22, 47-71

Desrosiers, J., Sauve M. and Soumis F. (1988). Lagrangian Relaxation Methods for Solving the Minimum Fleet Size Multiple travelling salesman Problem with Time Windows, Management Science, 34, 1005-1022

Desrosiers, J., Solomon, M. and Soumis, F. (1992). Time Constrained Routing and Scheduling, to be published in Handbook of Operations Research and Management Science: Networks (edited by Ball, M., Magnanti, T., Monma, C. and Nemhauser, G.,North-Holland.

Golden, B. and Assad, A. (Eds.) (1988). Vehicle Routing: Methods and Studies, North-Holland.

Kolen, A.W.J., Rinnooy Kan, A.H.G. and trienekens H.W.J.M. (1987). Vehicle Routing with Time Windows, Operations Research, 35, 266-273

Lee, Myung S. (1992). New Algorithms for the m-TSPTW, Ph.D. dissertation, University of Maryland

Lin, S. (1965). Computer Solutions of the travelling salesman Problem, Bell Syst. Tech. Journal, 44, 2245-2269

Lin, S. and Kernighan, B. (1973). An Effective Heuristic Algorithm for the Traveling Salesman Problem, Operations research, 21, 498-516

Solomon, M. (1987). Algorithms for the vehicle Routing and Scheduling Problems with Time window Constraints, Operations Research, 35, 254-265

Solomon, M., Baker, E. and Schaffer, J. (1988). Vehicle Routing and scheduling Problems with Time Window Constraints: Efficient Implementation of Solution Improvement Procedures, Vehicle Routing: Methods and studies, North-Holland

Swersey, A. and Ballard, W. (1984). Scheduling School Buses, Management Science, 30, 844-853