International Journal of Industrial Engineering, 15(1), 36-44, 2008.

Lower Bounds For Hierarchical Chinese Postman Problem

Purushothaman Damodaran[1], Murali Krishnamurthi[2], and Krishnaswami Srihari[3]

[1]Department of Industrial and Systems Engineering
Florida International University
Miami, FL 33174

[2]Department of Industrial Engineering
Northern Illinois University
DeKalb, IL 60115

[3]Department of Systems Science and Industrial Engineering
State University of New York
Binghamton, NY 13902

Arc routing problems aim at finding a least cost traversal on a network with or without additional constraints. The Hierarchical Chinese Postman Problem (HCPP) is an arc routing problem. HCPP is NP-hard and several heuristics have been developed to solve this problem. The Chinese Postman Problem (CPP) tour solution is a known lower bound for the HCPP. This paper presents a heuristic that will prescribe improved lower bounds for the HCPP when compared to the CPP solutions. Better lower bounds aid exact search methods, such as branch-and-bound, to find an optimal solution in a shorter run time. It can also be used to determine the quality of a heuristic solution. Several problem instances were generated to evaluate the proposed heuristic. Experimental results indicate that our lower bounds are better than the CPP solution for all the sample problems chosen.

Significance: The itinerary for a snow plow truck to follow during a snowfall is determined by solving a HCPP. A good lower bound can aid decision makers to effectively serve the taxpayers. Proper planning can reduce the number of accidents and property damage during a snow storm.

Keywords: Arc routing problems; Chinese Postman Problem; Hierarchical Chinese Postman Problem; Snow removal planning; Lower bounds

(Received 27 February 2006; Accepted in revised form 5 March 2007)

1. INTRODUCTION

The Vehicle Routing Problem (VRP) can be described as the problem of designing optimal delivery or collection routes from one or several depots to a number of geographically scattered cities or customers subject to side constraints (Laporte, 1992). The VRP covers activities such as retail distribution, school bus scheduling, mail delivery, street sweeping, snow removal, waste collection, and communication distribution management. The VRP can be broadly divided into the arc-covering problem and the node-covering problem. In an arc-covering problem, the demand for service is along any arc in a given network. Examples of the arc-covering problem are the Chinese Postman Problem (CPP), the Rural Postman Problem (RPP), and the Hierarchical Chinese Postman Problem (HCPP). The objective in the CPP is to determine a shortest tour on a network such that each arc on the network is covered at least once. Unlike the CPP, the objective of a RPP is to determine a minimum cost traversal on only a subset of arcs in a network. Whereas, in HCPP, the objective is to determine least cost traversal in a hierarchical network such that the higher priority arcs are serviced before lower priority arcs and each arc in the network is serviced at least once. An example for HCPP is the snow removal problem; Section 2 discusses this real-life application in detail. In a node-covering problem, the demand for service is at the nodes in a given network. Examples for node-covering problems are the Traveling Salesman Problem and the Transportation Problem.

A hierarchical network is a network that contains arcs with different priority levels and each level constitutes one hierarchy of the network. Consider a network $G(N,A)$ with a set of nodes N and a set of arcs A. $G(N,A)$ is a hierarchical network if it can be broken into k different sub-networks such that each sub-network represents a hierarchical level. That is, $G(N,A) = G_1(N_1,A_1) \cup G_2(N_2,A_2) \cup \ldots \cup G_k(N_k,A_k)$, where G_k is the k^{th} sub-network induced by all arcs with priority k and corresponding nodes of these arcs. If arcs with priority p (A_p) have higher priority over arcs with priority q (A_q) in the hierarchical network, then the objective of the HCPP is to determine a minimum cost traversal such that arcs with priority p

are included in the tour before arcs with priority q. However, if the sub-network G_p is not connected, then in order to connect G_p, arcs from other priority levels may be used. For example, the network shown in Figure 1 has two hierarchy levels, and therefore, has two sub-networks, G_p and G_q as shown in Figure 2.

The sub-networks induced by priority p arcs and priority q arcs are not connected. The sub-network G_p has two connected components; Arcs (1,2), (2,3), and (3,1) are connected and form a component and arcs (4,5), (5,6), and (6,4) are connected and form another component. Some arcs with priority q, dotted lines shown in Figure 3, have to be used to connect these two components in order to make the sub-network G_p connected. As arcs with priority p have to be served before serving the arcs with priority q, the arcs with priority q that are used for connecting G_p will be traversed, but will not be serviced until all priority p arcs are served. While serving priority p arcs, any priority q arc traversed will not be served. This travel along priority q arcs, when serving priority p arcs is called "deadhead travel". If any arc in A_p that has already been served is traversed for the purpose of serving other priority p arcs, then they too will be used only for the purpose of traversing, and no service will be provided. Since deadhead travel does not accomplish any useful work, it should be minimized.

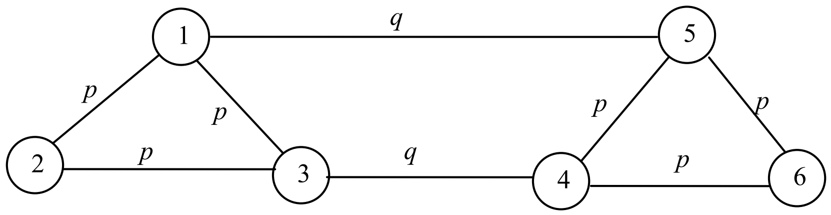

Figure 1. A hierarchical network with two hierarchy levels

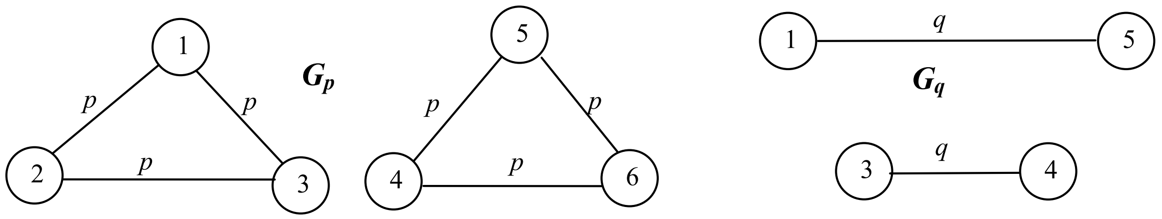

Figure 2. Sub-networks induced by priority p and priority q arcs

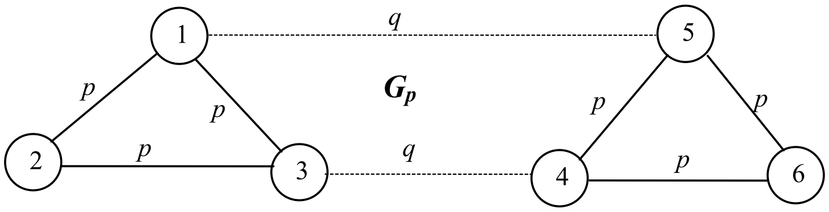

Figure 3. Sub-network G_p after connecting the disconnected components

Dror, Stern, & Trudeau (1987) showed that the HCPP is NP-hard. Consequently, Lemieux & Campagna (1984), Alfa & Liu (1988), Dror, Stern, & Trudeau (1987), and Krishnamurthi & Damodaran (1998) proposed heuristics for the HCPP. The HCPP is a variation of the CPP. The optimum solution to the CPP is the least cost tour that is obtained when each arc in the network is traversed exactly once (Christofides, 1975). If such a tour exists in a network, then the network is an Eulerian network and the tour is an Euler tour. If a given network consist of only even degree nodes (in an undirected network, the degree of a node v is the number of arcs incident to v), then it is possible to trace an Euler tour. However, if a network has odd degree nodes (the number of odd degree nodes in a network is even), then it is not possible to trace an Euler tour. In such a network, some of the arcs have to be traversed more than once and hence, deadhead travel is inevitable.

In the HCPP, if the sub-network induced for any hierarchical level has some disconnected components then deadhead travel is inevitable. However, if the sub-network induced for each hierarchical level is connected and each node in the sub-network is of even degree then it is possible to trace an Euler tour. In this case, the HCPP tour is obtained by concatenating the Euler tours obtained for each hierarchy level in the order of their hierarchy. Even in this situation, deadhead travel cannot be avoided if there is no arc directly connecting two successive hierarchy levels. For example, if each node in the sub-network induced by priority 1 and priority 2 arcs is of even degree and if there is no node common to both these sub-networks then arcs from other priority have to be used for concatenating the Euler tours obtained for G_1 and G_2.

Edmonds and Johnson (1973) have shown that the lower bound for the CPP is the Euler tour traced on the network. As HCPP is a variation of CPP, the lower bound for HCPP is the CPP tour traced on the network. The primary objective of this paper is to propose a heuristic to find better lower bounds for the HCPP. The secondary objective is to compare the lower bounds obtained from the proposed heuristic with the CPP solution and show the improvements that can be achieved by adopting the proposed heuristic. The benefits of this research are (1) a better lower bound for the HCPP will reduce the computational effort when a branch-and-bound procedure is adopted to solve HCPP or its applications and (2) to test the performance of the heuristics proposed to solve the HCPP more accurately.

The rest of the paper is organized into six sections. Section 2 gives a practical example for the HCPP. Section 3 presents the literature reviewed. Section 4 presents our heuristic to determine the lower bound for the HCPP. Section 5 presents our experimental results and Section 6 presents our conclusions.

2. SNOW REMOVAL PROBLEM AS HCPP

For the purpose of providing road service during a snowfall, a city or county is usually divided into several zones, and the equipment and crew necessary for handling the task are located in one or several zones to provide the best service possible. Each zone is composed of a network of roads and public places, which are prioritized for the purpose of snow removal. The highest priority during snow removal is given to main roads (business districts), schools, and hospital areas. Second priority is given to snow removal routes that lead to main roads, schools, and hospitals. Third priority is given to residential streets, and fourth priority to public parking lots, sidewalks, etc. Snow on higher priority roads should be cleared before lower priority roads in the zone.

Since roads with the same priority are not necessarily located adjacent to each other, while serving the higher priority roads some of the lower priority roads may have to be traversed for reaching the higher priority roads. Though the lower priority roads are traversed to reach higher priority roads, the snow removal team will not serve the lower priority roads. The distance traversed along a road when the road is not served is referred to as "Deadhead" travel. Deadhead travel can considerably slow down the completion of the snow removal process and hence, should be minimized. Deadhead travel increases the cost of road maintenance as it includes the cost of labor, delay in delivery of service, etc. Hence, there is an opportunity to reduce the cost of road maintenance by reducing the deadhead travel distance. By reducing the deadhead travel, it is obvious that the damage caused to the property, public, and the cost of snow removal process can be reduced.

By treating each street as an arc and each street intersection as a node, each zone can be represented by a network. As the streets are prioritized, the network is actually a hierarchical network. By minimizing the deadhead travel, the time taken to plow or spray salt can be minimized. Thus, the problem of determining optimal routes for the snow removal team is a HCPP.

3. LITERATURE REVIEW

Edmonds and Johnson (1973) developed an algorithm for finding an optimum solution to the CPP. As a first step in the algorithm, all odd degree nodes in the network are converted into even degree nodes by applying the minimum-matching algorithm. Artificial arcs are added to the original network in order to connect the odd degree nodes. In the second step, an Euler tour is traced on the modified network that would be the optimal solution to the CPP. This optimal solution to the CPP can be used as a lower bound for the HCPP. The total length of an optimal postman problem route is the same for every starting node (Minieka, 1978).

Edmonds & Johnson (1973) described the necessary and sufficient condition for the existence of an Euler tour and the solution procedure to the CPP using matching theory (we refer to Gondran & Minoux (1984) for a detailed discussion on matching algorithms). They presented solutions for the postman problem when the graph is undirected, directed, or mixed. Minieka (1979) solved this case of the CPP as an "integer minimum flow with gains" problem using integer linear programming. This method is not elegant but it at least suggests a method to find feasible solutions. Papadimitriou (1976) has shown that the solution for the postman problem on mixed graphs with some odd degree nodes is NP-complete. A detailed survey of arc routing problems (ARP) and main algorithm results for the undirected CPP, the directed CPP, the windy postman problem (determining a least cost traversal of all arcs in an undirected graph in which the cost of traversing an arc depends on the direction of travel), the mixed CPP, and the hierarchical CPP can be found in Eiselt, Gendreau & Laporte (1995). We refer to Dror (2000) for state-of-the-art exposition of arc routing problems; it presents a number of solution methodologies in a variety of application settings.

Dror, Stern & Trudeau (1987) developed an algorithm to determine an optimal postman tour on a graph in which the arcs are partitioned and a precedence relation is defined. The drawback of this algorithm is that all arcs of the same priority level should be connected. If the arcs are not connected, and if there exists two or more components, the procedure to move between the components is not addressed. In reality, arcs of the same priority level may not always be connected.

Alfa and Liu (1988) presented a technique to arrive at a very good solution to the CPP in a directed, hierarchical network. Their work also presents a procedure for selecting arcs from another hierarchy to include in the unconnected hierarchy in order to make it connected. The postman route obtained in this procedure may not be the global optimum since the problem of connection and evenness were treated independently. To ensure that the route obtained is the global optimum, the problem of connecting the components of the network and making the network even should be combined and analyzed simultaneously. Damodaran (1997) developed a heuristic for the HCPP that handles the problem of connecting the components and making the network even simultaneously. The heuristic was shown to perform better that Alfa and Liu's heuristic when the sub-network induced by each hierarchy level is disconnected.

Ghiani & Improta (2000) proposed an exact algorithm to solve a HCPP with some special properties. In general, HCPP is NP-hard, however polynomial-time algorithms (Dror, Stern & Trudeau, 1987) are available to solve special cases. One such special case is when all the components are connected and the precedence relationship is linear. Cabral, Gendreau, Ghiani, & Laporte (2003) proposed a method to transform the HCPP to RPP and use branch-and-cut exact procedures to solve the HCPP when the precedence relationships are linear.

After making suitable modifications to the basic ARP, CPP or RPP, a number of practical problems such as street sweeping (Bodin & Kursh, 1978), garbage collection (Beltrami & Bodin, 1974), snow plowing (Lemieux & Campagna, 1984; Alfa & Liu, 1988; Haslam & Wright, 1991; Damodaran & Krishnamurthi, 2005), meter reading (Stern & Dror, 1979), police patrol scheduling, routing of road crews, school bus scheduling (Swersey & Ballard, 1984), etc. have been addressed. Udi & Israni (1984) have developed an algorithm that finds a sequence of torch paths to cut a nested stock sheet with a minimum number of pierce points. Stern & Dror (1979) developed a heuristic to solve a special type of edge oriented multiple open tour routing problem. This heuristic was developed to reduce the cost incurred in routing the electric meter readers in Israel.

Though the CPP solution obtained for a network is the lower bound for the HCPP, this lower bound is not tight. Hence, this paper prescribes a heuristic to determine a better lower bound for the HCPP. HCPP is NP-hard. Therefore, in order to evaluate any new and/or existing heuristic's performance, our lower bound results can be very useful. One another important reason to have a good lower bound is to quicken the search process in any branch-and-bound procedure. If branch-and-bound is used to find optimal solutions for various applications (such as snow plowing, street sweeping, etc.) of the HCPP, a good lower bound will aid in reducing the search process.

4. A HEURISTIC TO FIND A LOWER BOUND FOR THE HCPP

In HCPP higher priority arcs have to be included before lower priority arcs in the tour, and if the sub-network generated by a particular hierarchy is not connected, then deadhead travel is inevitable. Hence, when the sub-network induced by any level in the hierarchy is disconnected, the total distance traversed to serve each arc at least once is always greater in HCPP than the total distance traversed to serve each arc at least once in CPP. Even when the sub-network induced by all hierarchical levels is connected, the total distance traversed in HCPP is at least same as the total distance traversed in CPP. Hence, an exact (optimal) CPP solution can always be the lower bound for the HCPP. Damodaran (1997) proposed a heuristic for the HCPP and showed that it performed better than Alfa and Liu's heuristic when the sub-network induced by each hierarchy level is disconnected. By relaxing the arc priorities and suitably modifying Damodaran's heuristic, a better lower bound for the HCPP can be obtained. The heuristic to determine the lower bound for the HCPP is discussed in this section.

The starting node is the node from which a postman tour has to begin. Since, the heuristic is aimed at determining a closed postman tour, the starting node is also the final node in the tour. If a node is incident to several arcs, then it is always possible to choose an arc which does not divide the network because it has been proven that there can be at most one arc incident to a node which divides the network into components (Gibbons, 1985). Whenever an arc is included in a tour, that arc can be assumed to be removed from the original network to check if this arc removal leads to disconnected components or not. The proposed heuristic makes use of the above fact and strives to include arcs that will not divide the network into components. However, if there is only one arc incident to a node then that arc is included into the tour, which divides the network. In order to proceed further with the remaining arcs, a node that is incident to one or more arcs and closer to the current node is searched and the shortest path connecting them is included in the tour.

The inputs required for the proposed heuristic are the distance between two nodes, number of nodes in the network, and the starting and end node for the closed postman tour. The notation used in the heuristic are:

S*	Starting and end node for the closed postman tour
S	Starting node
CV	Current node
EC	List of nodes ordered according to the sequence in which they are visited
E'	Set of arcs already traced
A(v)	List of all nodes which are directly adjacent to node v

The pseudo code for the proposed heuristic is:

```
1.  S ← S*, EC ← [S], CV← S, v ← S, E' ← ∅
2.  Find the shortest path and distance between any two nodes in the network using Floyd's algorithm
3.  Find the Adj(v) for all nodes v in G
4.  While | Adj(v) | > 0 for any v do begin
5.  While | Adj(S) | > 0 do begin
    5a.   if | Adj(CV) | > 1 then
                  find a node v ∈ Adj(CV) such that | Adj(v) | > 1, if there is no such v then
                  choose any v from Adj(CV) arbitrarily
    5b.   else if  | Adj(CV) | == 1 then
                  let the node in  Adj(CV) be denoted by v
    5c.   else
                  find a node v in G such that | Adj(v) | > 0 and  the distance between CV and v is  the minimum S
                  ← v, CV ← S, add the shortest path sequence from CV to v to the end of EC list, go to step 7
              endif
6.     Delete v in Adj(CV) and CV in Adj(v), E' ← E' ∪ {(CV,v)}, CV ← v, add v to the end of EC list
         end
7.     Find a node with | Adj(v) | > 0 such that the distance between CV and v is  the minimum
    7a. if no such v then
                go to step 4
    7b. else
                CV ← v, S ← v, add the shortest path sequence from CV to v to the end of EC list
            endif
       end
8.  If CV == S* then
       go to step 9
    else
       add the shortest path sequence from CV to S* to the end of the EC list
    endif
9.  Print EC
```

Step 1 initializes all the variables. In step 2, the shortest distance and shortest path between any two nodes in a given network is determined using Floyd's algorithm (Foulds, 1984). In step 3, a list of adjacent matrices for all the nodes in the network *G* is determined. The adjacency matrix list gives the set of all arcs incident to a node. A node can be incident to one, more than one, or no arcs. If the number of arcs incident to a node is more than one then in step 5a, an arc that does not divide the network into components is searched and is included in the tour. But if the number of arcs incident is exactly one, then in step 5b, that particular arc is included in the tour and this inclusion will divide the network into components. If there is no arc incident to a node, then in step 5c, a node that is incident to one or more arcs is selected and the shortest path connecting them is included in the tour. The arcs that are included in the tour are deleted from *G* in step 6. Steps 5 and 6 are repeated until all the arcs incident to S are included in the tour.

Step 4 checks if all the arcs in *G* are included in the tour. If there are some arcs yet to be covered and if there is no arc incident to S, then in step 7 a node, say v, which is incident to one or more arcs and close to CV is searched. The shortest path connecting these two nodes is included in the tour and CV and S are set to v. Steps 4 through 7 are repeated until all arcs are included in the tour at least once. Once all the arcs are included, if the node incident to the last arc in the tour is not S*, then the shortest path connecting the current node and S* is included in the postman tour. Step 9 prints the postman tour (lower bound) for the entire network. The postman tour obtained by applying the heuristic is a lower bound because the heuristic does not construct the tour following arc priorities.

5. EXPERIMENTATION

In order to evaluate the performance of the proposed heuristic, the lower bounds for 31 sample problems were determined by applying the proposed heuristic and compared with the CPP solution. The sample problems were generated from six different networks (networks 1-6). Keeping in mind the page limitations, all the networks and their corresponding arc parameters are not included in this paper. Any interested reader can find this information in Damodaran (1997). Table 1 indicates the network from which a sample problem was generated. The cost or distance associated with the arcs for all the sample problems generated from a network will be the same. However, the sample problems generated from a network differ in their priorities along the arcs. For example, the sample problems 1-7 and 29 are generated for network 1. The cost or distance associated with the arcs will remain the same, but their priority differs between sample problems.

Table 1. Sample problem information

Network	Sample Problems
Network 1	1-7 and 29
Network 2	8-13 and 30
Network 3	14-17 and 31
Network 4	18-21 and 25
Network 5	22-24
Network 6	26-28

Table 2 gives the lower bounds obtained from the proposed heuristic, CPP solution, and the improvements in percentage. The results reported in Table 2 clearly indicate that the proposed heuristic prescribes a better lower bound for all the sample problems. Since, the heuristic and CPP ignore the priorities along the arc while tracing the postman tours, the lower bound from both the heuristic and CPP are the same for all the sample problems generated from a network. For example, the lower bounds obtained from the heuristic are the same for sample problems 1-7 and 29 that were generated from network 1.

Table 3 presents the lower bounds and the HCPP tour distances obtained by applying two heuristics for all the 31 sample problems. It is evident from these results that our heuristic prescribed better and valid lower bounds. The GAP indicates the performance of the HCPP heuristics and gives an idea which heuristic is better. On the average, the GAP is 14.35% and 16.67% for Damodaran (1997) and Alfa and Liu (1988) heuristics, respectively. Figure 4 shows the GAP1 and GAP2 values for all the sample problems.

The comparisons indicate that our lower bounds are valid. The GAP values help to measure the performance of any heuristic proposed for the HCPP. If the solution from a heuristic were to be compared with the CPP solution to measure its quality, the measures would be quite off. For example, the HCPP solution for sample problem 22 by applying Alfa and Liu heuristic is 11000, the CPP solution is 7400, and, hence, the GAP is 48.65%. When the improved lower bounds are used, the GAP is only 30.95%. Our lower bound calculations also help us to conclude that the solutions obtained by Damodaran for sample problems 4, 7, 22 and 23 is optimum (GAP1 = 0%). Without these lower bound results, it is difficult to prove optimality. A comparison of GAP1 and GAP2 helps us to decide which heuristic performs better.

6. CONCLUSIONS

HCPP is NP-hard and therefore several heuristics are available to solve these arc routing problems. Although CPP solution may be treated as a lower bound for the HCPP, we have demonstrated that our heuristic can be used to find improved lower bounds. Our heuristic was applied to a set of 31 sample problems and the results were compared with the CPP solution. The results indicate that our heuristic finds a better lower bound for all the sample problems. Another important contribution is that our lower bounds will be very useful to limit the search in a search tree, when a researcher resolves to use a branch-and-bound approach to solve the HCPP or any applications that fall under the umbrella of HCPP. It can also be used to compare the performance of different heuristics proposed to solve HCPP. It is evident from our experimental study that the lower bounds can help us to confirm optimality. With the improved lower bounds, four sample problems solved using Damodaran's heuristic have been confirmed to be optimal.

Table 2. Lower bounds from the heuristic and CPP

Sample Problem	Heuristic (1)	CPP (2)	% Improvement (1)-(2)/(2)
1	314	294	6.80
2	314	294	6.80
3	314	294	6.80
4	314	294	6.80
5	314	294	6.80
6	314	294	6.80
7	314	294	6.80

8	87	87	0.00
9	87	87	0.00
10	87	87	0.00
11	87	87	0.00
12	87	87	0.00
13	87	87	0.00
14	118	118	0.00
15	118	118	0.00
16	118	118	0.00
17	118	118	0.00
18	150	128	17.19
19	150	128	17.19
20	150	128	17.19
21	150	128	17.19
22	8400	7400	13.51
23	8400	7400	13.51
24	8400	7400	13.51
25	118	118	0.00
26	134	122	9.84
27	134	122	9.84
28	134	122	9.84
29	314	294	6.80
30	87	87	0.00
31	118	118	0.00

Table 3. Comparing lower bounds to HCPP heuristics

Sample Problem	**Heuristic (1)**	**CPP (2)**	**Damodaran (3)**	**%GAP1 (3)-(1)/(3)**	**Alfa & Liu (4)**	**%GAP2 (4)-(1)/(4)**
1	314	294	322	2.55	321	2.23
2	314	294	328	4.46	354	12.74
3	314	294	343	9.24	318	1.27
4	314	294	314	0.00	320	1.91
5	314	294	351	11.78	333	6.05
6	314	294	322	2.55	334	6.37
7	314	294	314	0.00	361	14.97
8	87	87	101	16.09	92	5.75
9	87	87	102	17.24	105	20.69
10	87	87	100	14.94	100	14.94
11	87	87	101	16.09	91	4.60
12	87	87	95	9.20	96	10.34
13	87	87	91	4.60	92	5.75
14	118	118	163	38.14	135	14.41
15	118	118	164	38.98	151	27.97
16	118	118	155	31.36	142	20.34
17	118	118	158	33.90	158	33.90
18	150	128	184	22.67	172	14.67
19	150	128	152	1.33	166	10.67
20	150	128	156	4.00	156	4.00
21	150	128	174	16.00	164	9.33
22	8400	7400	8400	0.00	11000	30.95

23	8400	7400	8400	0.00	12800	52.38
24	8400	7400	11800	40.48	12200	45.24
25	118	118	132	11.86	165	39.83
26	134	122	162	20.90	191	42.54
27	134	122	156	16.42	192	43.28
28	134	122	137	2.24	135	0.75
29	314	294	336	7.01	354	12.74
30	87	87	95	9.20	98	12.64
31	118	118	167	41.53	147	24.58

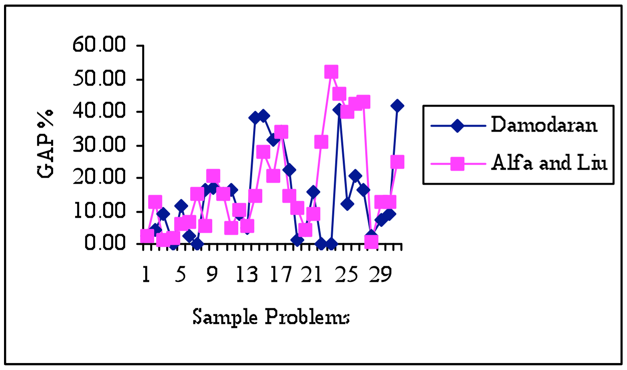

Figure 4. GAP comparisons

7. REFERENCES

1. Alfa, A.S. and Liu, D.Q. (1988). Postman routing problem in a hierarchical network. Engineering Optimization, 14: 127-138.
2. Beltrami, E. and Bodin, L. (1974). Networks and vehicle routing for municipal waste collection. Networks, 4: 65-94.
3. Bodin, L.D. and Kursh, S.J. (1978). A computer-assisted system for the routing and scheduling of street sweepers. Operations Research, 26 (4): 525-537.
4. Cabral, E.A., Gendreau, M., Ghiani, G. and Laporte, G. (2004). Solving the hierarchical Chinese postman problem as a rural postman problem. European Journal of Operational Research, 155(2): 44-50.
5. Christofides, N. (1975). Graph Theory: An Algorithmic Approach. Academic Press, New York.
6. Damodaran, P. (1997). A methodology for dynamic planning of road service during a snow fall. M.S. Thesis, Northern Illinois University, DeKalb, IL.
7. Damodaran, P. and Krishnamurthi, M. (2005). A continuous simulation model for snow removal. International Journal of Industrial Engineering, 12(2): 188-197.
8. Dror, M., Stern, H. and Trudeau, P. (1987). Postman tour on a graph with precedence relation on arcs. Networks, 17: 283-294.
9. Dror, M. (ed.) (2000). Arc Routing: Theory, Solutions and Applications. Kluwer Academic Publishers, Boston.
10. Edmonds, J. and Johnson, E.L. (1973). Matching, Euler tours and the Chinese postman problem. Mathematical Programming, 5: 88-124.
11. Eiselt, H.A., Gendreau M. and Laporte, G. (1995). Arc routing problems, part I: The Chinese Postman Problem. Operations Research, 43 (2): 231-242.
12. Foulds, L.R. (1984). Combinatorial Optimization for Undergraduates. Springer-Verlag, Inc., New York.
13. Ghiani, G. and Improta, G., (2000). An algorithm for the hierarchical Chinese postman problem. Operations Research Letters, 26: 27-32.
14. Gibbons, A. (1985). Algorithmic Graph Theory. Cambridge University Press, New York.
15. Gondran, M. and Minoux, M. (1984). Graphs and Algorithms. John Wiley & Sons, New York.
16. Haslam, E. and Wright, J.R. (1991). Application of routing technologies to rural snow and ice control. Transportation Research Record, 1304, 202-211.
17. Laporte, G. (1992). The vehicle routing problem: An overview of exact and approximate algorithms. European Journal of Operations Research, 59: 345-358.

18. Lemieux, P.F. and Campagna, L. (1984). The snow ploughing problem solved by a graph theory algorithm. Civil Engineering Systems, 1: 337-341.
19. Krishnamurthi, M. and Damodaran, P. (1998). A modified postman tour heuristic for efficient snow removal planning. Proceedings of 7th Industrial Engineering Research Conference, Banff, Canada.
20. Minieka, E. (1979). The Chinese Postman Problem for mixed networks. Management Science, 25 (7): 643-648.
21. Minieka, E. (1978). Optimization Algorithms for Networks and Graphs. Marcel Dekker, Inc, New York.
22. Papadimitriou, C. (1976). On the complexity of edge traversing. Journal of the Association of Computing Machinery 23: 544- 554.
23. Stern, H. and Dror, M. (1979). Routing electric meter readers. Computers and Operations Research, 6: 209-223.
24. Swersey, J.A. and Ballard, W. (1984). Scheduling school buses. Management Science, 30 (7): 844-853.
25. Udi, M. and Israni, S. (1984). Pierce point minimization, and optimal torch path determination in flame cutting. Journal of Manufacturing Systems, 3 (1): 81-89.

BIOGRAPHICAL SKETCH

Purushothaman Damodaran is an Assistant Professor in the Department of Industrial & Systems Engineering at Florida International University, Miami, FL. His current research interests include large-scale optimization, scheduling, simulation, logistics, and electronics manufacturing. He received his Ph.D. from Texas A&M University in Industrial Engineering. He is a member of INFORMS and IIE. Dr. Damodaran has worked and continues to work closely with companies such as IBM, Motorola, Sanmina-SCI, Jabil Circuits, Maines Paper & Food Services, and Innovative Scheduling, Inc., on sponsored research projects which are critical to their needs.

Murali Krishnamurthi is Associate Professor of Industrial and Systems Engineering at Northern Illinois University. He received his B.E. in Mechanical Engineering from University of Madras (India), M.S. in Industrial and Systems Engineering from Ohio University and Ph.D. in Industrial Engineering from Texas A&M University. His teaching and research interests include optimization techniques, system simulation, information systems, project management, engineering ethics, environmental management systems, and expert systems. He has received more than $2 million in sponsored project funding from federal, state, industry and professional society sources. He currently serves also as the Director of Faculty Development and Instructional Design Center at Northern Illinois University.

Krishnaswami (Hari) Srihari joined the State University of New York at Binghamton, New York in August 1988. He received his M.S. (1985) and Ph.D. (1988) in Industrial Engineering and Operations Research from Virginia Polytechnic and State University, Blacksburg, Virginia. Dr.Srihari's research is focused on the electronics manufacturing domain. A recent review of his group indicated that he had received over 16 million dollars in external research funding for the past few years, has published over 325 research papers, and authored over 950 technical reports.Dr. Srihari is the Director of Watson Institute for Systems Excellence (WISE).