An Approximation Algorithm for the Capacitated Arc Routing Problem

Sanne Wøhlk*

Department of Business Studies, Aarhus School of Business, University of Aarhus, Fuglesangs Alle 4, DK-8210 Aarhus V, Denmark

Abstract: In this paper we consider approximation of the Capacitated Arc Routing Problem, which is the problem of servicing a set of edges in a graph using a fleet of capacity constrained vehicles. We present a $\frac{7}{2}-\frac{3}{W}$ approximation algorithm for the problem and prove that this algorithm outperforms the only existing approximation algorithm for the problem. Furthermore, we give computational results showing that the new algorithm performs very well in practice.

1. INTRODUCTION

When solving an optimization problem to suboptimality, two goals are followed. The first goal is to obtain a solution that is as close to the optimal as possible, which leads to the construction of problem specific heuristics and meta heuristics, where the latter often outperforms the former. The second goal is to obtain a solution, which is guaranteed to be within a certain factor of the optimal. This goal leads to the construction of approximation algorithms.

In this paper we will present an algorithm for the Capacitated Arc Routing Problem (CARP) with the triangle inequality preserved by the cost matrix, which is an approximation algorithm with at most the same approximation factor as the only existing approximation algorithm for the problem, and which performs very well in practice, in that it is highly competitive to the existing problem-specific heuristics for the problem on the set of 143 benchmark instances. We refer to the algorithm as A-ALG.

The CARP is the problem of servicing a set of demand edges in a graph using a fleet of capacity constrained vehicles. CARP occurs in practice in problems such as street sweeping and refuse collection. The problem was first suggested by Golden and Wong in 1981 [1] and has since been the target for heuristics and lower bounding procedures. Among the best performing heuristics are a Tabu Search algorithm by Hertz *et al*. [2], a Genetic Algorithm, [3], and a Memetic Algorithm, [4], both by Lacomme *et al*. and a Variable Neighborhood Descent Algorithm by Hertz and Mittaz [5]. More recently, the problem has also been considered from an LP based point of view. We refer the reader to [6-8] for a survey of the literature regarding CARP.

Formally, the CARP is stated as follows: Given a connected undirected graph $G = (N,E,C,Q)$, where N is the set of nodes, E is the set of edges, C is a cost matrix, and Q is a demand matrix, and given a number of identical vehicles each with capacity W, find a number of tours such that 1) Each edge with positive demand is serviced by exactly one vehicle, 2) The sum of demands of those edges serviced by each vehicle does not exceed W, and 3) The total cost of the tours is minimized. Throughout this text we assume that edge demands and W are integral and $W \geq 3$.

The CARP has been proved to be *NP*-hard by Golden and Wong 1981 [1].

Definition 1. *Let $\alpha > 1$ be a constant. A polynomial time algorithm, A is an α-approximation algorithm if $c(A(\sigma)) \leq \alpha \cdot c(OPT(\sigma))$ for all instances, σ, of the problem, where $c(A(\sigma))$ is the cost of the solution returned by A, and $c(OPT(\sigma))$ is the cost of an optimal solution.*

If the cost matrix does not respect the triangle inequality, the CARP is even hard to approximate. This can easily be proved by noting that the TSP is a special case of the VRP, which was proved to be a special case of the CARP by Golden and Wong 1981 [1]. By noting that the general TSP has been proved not to be in *APX*, the following theorem follows.

Theorem 2. *If C does not satisfy the triangle inequality, finding an α-approximation of the CARP is NP-hard for all $\alpha > 0$.*

Problems are often easier to approximate when the cost matrix respects the triangle inequality. This is, for instance, true for the Traveling Salesman Problem (TSP), where approximation of the general problem is *NP*-hard as proved by Gonzales and Sahni 1976 [9], but where instances respecting the triangle inequality can be approximated in polynomial time using, for example, the algorithm by Christofides 1976 [10]. This is also the case for the CARP, where approximation is hard for the problem in general, but an approximation algorithm can be constructed for the case where the triangle inequality is respected as we shall see next. However the following result by Golden and Wong 1981 [1] shows that even in this case there is a limit as to how tight the CARP can be approximated if $P \neq NP$.

Theorem 3. *Finding a $\frac{3}{2}$-approximation of the CARP with C satisfying the triangle inequality is NP-hard.*

There is a one-to-one correspondence between the class of CARP problems and the class of General Capacitated Routing Problems (GCRP). This results from the fact that demand nodes can be transformed into demand edges and

*Address correspondence to this author at the Department of Business Studies, Aarhus School of Business, University of Aarhus, Fuglesangs Alle 4, DK-8210 Aarhus V, Denmark; E-mail: sanw@asb.dk

1874-2432/08

vice versa, as proved by Golden and Wong 1981 [1] and Assad *et al*. 1987 [11], respectively. An approximation algorithm, Shortest Optimal Tour Partitioning (SOTP), has previously been suggested for the GCRP by Jansen 1993 [12]. This algorithm is based on a similar algorithm for the Capacitated Vehicle Routing Problem (CVRP) by Beasley 1983 [13] and has an approximation factor of $7/2 - 3/W$, where W is the vehicle capacity. Due to the above mentioned equivalence relation, the SOTP algorithm has the same approximation factor when applied to the CARP. Many other routing problems have been considered from an approximation point of view. See for example [14-16].

The idea of A-ALG is first to construct a giant tour that passes through all demand edges, and then use dynamic programming to optimally partition this tour into smaller tours that respect the vehicle capacity. The details of the algorithm are given in Section 2.

In Section 3 we show that A-ALG has an approximation factor of $7/2 - 3/W$, and prove that it always performs at least as well as SOTP, which is the only existing approximation algorithm for the problem.

Finally, in Section 4 we show that, in practice, AALG performs significantly better than SOTP since for the benchmark instances, the result obtained by A-ALG is strictly better than the one obtained by SOTP. Furthermore, we show that A-ALG is highly competitive to the existing problem-specific heuristics.

2. THE ALGORITHM

The first step in the algorithm is to construct a giant tour, i.e. a tour that passes through all demand edges in the cheapest possible way. This problem is known as the Rural Postman Problem (RPP), and was proved to be *NP*-hard by Rinnooy Kan and Lenstra 1976 [17]. Frederickson 1979 [18] stated that a $\frac{3}{2}$-approximation algorithm for RPP "is obtainable by using an algorithm similar to the traveling salesman algorithm by Christofides" (He refers to Christofides 1976 [10]). Jansen 1992 [19] presented a $\frac{3}{2}$-approximation algorithm for the General Routing Problem (GRP). Following these results, a giant tour of cost at most 3/2 times the optimal is obtained for the RPP when the cost structure respects the triangle inequality, in the following way.

If the graph induced by the required edges is connected, the RPP can be solved in polynomial time by the CPP algorithm (Adding the edges of a minimum cost perfect matching of the odd degree nodes and making an Euler Tour). If this is not the case, let E_R be the set of required edges and let $G_R = (N_R, E_R)$ be the graph induced by E_R. If the Depot node is not included in N_R, the Depot node is split into two nodes which are connected with a zero demand required edge. This edge will be contained in a separate component. The algorithm is outlined as follows:

1. Calculate shortest paths between the connected components. Let k be the number of connected components in G_R. Let $\overline{G}$ be a complete graph with k nodes, one for each component in G_R. The cost of an edge (i, j) in $\overline{G}$ equals the shortest path length between any node in component C_i and any node in component C_j in G_R. I.e. $c_{ij} = \min_{v \in C_i, w \in C_j} SPL(v, w)$, where $SPL(\cdot, \cdot)$ is the shortest path cost between the nodes indicated.
2. Construct a minimum spanning tree between the nodes in $\overline{G}$ and let MST be the set of edges on the tree.
3. Identify the set, Θ, of odd degree nodes with respect to $E_R \cup MST$.
4. Construct a minimum cost perfect matching between the nodes in Θ and let M be the set of matching edges.
5. Construct an Euler tour in $G' = (N, E_R \cup MST \cup M)$. The cost of this tour is $c(E_R) + c(MST) + c(M)$.
6. Shorten the tour in the following way: Let the constructed Euler Tour, containing s edges, be $v_1 v_2 v_3 \ldots v_S v_1$. Some of these edges, (v_i, v_{i+1}), are in E_R and may not be removed. All other edges on the tour may be replaced without violating the feasibility of the tour. Now, scan through the tour until two consecutive edges (v_i, v_{i+1}) and (v_{i+1}, v_{i+2}) that are not in E_R are identified, and replace these two edges with one edge (v_i, v_{i+2}) with cost equal to the shortest path in G between v_i and v_{i+2}. Because of the triangle inequality, we have $c_{i,i+2} \leq c_{i,i+1} + c_{i+1,i+2}$. When this is done for the whole tour, at least one of any two adjacent edges is an E_R-edge.

Theorem 4. *The above algorithm is a* $\frac{3}{2}$*-approximation algorithm for the RPP.*

The proof mimics the similar proof for the GRP which can be found in Jansen 1992 [19] and will not be repeated here.

For the part of the algorithm that partitions the giant tour into vehicle tours, we consider the demand edges in the order in which they appear on the giant tour, and use two tables, $\tilde{T}$ and T, each of size $m \times m$, where m is the number of demand edges. Here $\tilde{T}(i, j)$ will contain the cost of servicing demand edges i through j in the given order optimally using a single uncapacitated vehicle, and $T(i, j)$ is the cost of servicing demand edges i through j in the given order optimally, when capacity restrictions are respected. The values in table $\tilde{T}$ are needed to calculate those of table T, there the optimal partitioning is found.

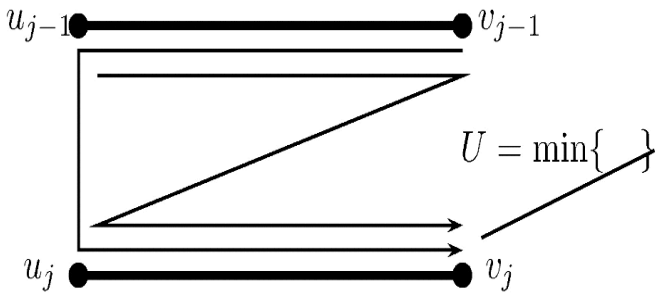

Fig. (1). Minimizing for each direction.

Let the end points of edge i be denoted by u_i and v_i respectively. To calculate the values in table $\widetilde{T}$, we use the following observations: For each $j > i$, servicing demand edge j can be done in two directions, from v_j to u_j or from u_j to v_j . Assuming that we service edge j in direction from u_j to v_j, edge $j-1$ can be serviced in two directions. Repeatedly minimizing over the two directions for edge $j-1$ gives us the cost for servicing edges i through j, where edge j is serviced in the specified direction, which is illustrated in Fig. (**1**). This value is stored in variable U in the algorithm. The similar value with edge j serviced in the opposite direction is stored in variable V. Finally the algorithm minimizes over the two directions and iterates. To summarize, we have the following.

For $i = 1$ to m do:

$$\widetilde{T}(i,i) = SPL(0,u_i) + SPL(v_i,0) + c_i$$
$$U' = SPL(0,u_i) + c_i$$
$$V' = SPL(0,v_i) + c_i$$
for $j = i+1$ to m do
$$U = \min\{U' + SPL(v_{j-1},u_j), V' + SPL(u_{j-1},u_j)\} + c_j$$
$$V = \min\{U' + SPL(v_{j-1},v_j), V' + SPL(u_{j-1},v_j)\} + c_j$$
$$\widetilde{T}(i,j) = \min\{U + SPL(v_j,0), V + SPL(u_j,0)\}$$
$$U' = U, V' = V$$

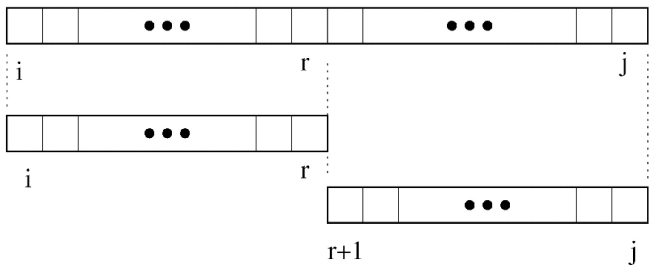

Fig. (2). Partitioning the tour.

To calculate the values in table T, we note that servicing edges i through j in that order using capacitated vehicles must be done using the following rule. If the edges can be serviced by a single vehicle, i.e. if their joined demand is small enough, due to the triangle inequality, it must be best to do so. In this case the cost can be obtained directly from table $\widetilde{T}$. If this is not the case, there must be some index, r with $i \le r < j$ such that edges r and $r + 1$ are serviced by different vehicles as illustrated in Fig. (**2**). In this case the cost can be obtained as $T(i, r) + T(r+1, j)$. Minimizing over all possible values of r gives the result. In conclusion, table T is filled in the order from $i = m$ down to 1, and from $j = i$ to m according to:

$$T(i,j) = \begin{cases} \widetilde{T}(i,j) & \text{if } \sum_{k=i}^{j} q_k \le W \\ \min_{r:i\le r<j}\{T(i,r) + T(r+1,j)\} & \text{else} \end{cases}$$

The value of an optimal partitioning of the giant tour is now given by $T(1, m)$.

3. THEORETICAL RESULTS

The goal of this section is to give some theoretical results about A-ALG and in particular to show that A-ALG is an approximation algorithm with approximation factor $7/2-3/W$. To this goal, we prove that A-ALG outperforms SOTP, which, to out knowledge, is the only existing approximation algorithm for the CARP.

As in A-ALG, the idea in SOTP is to construct a giant tour and to partition this tour into single vehicle tours. In SOTP, this is done by solving a shortest path problem in a directed graph where the costs of the arcs equals the cost of servicing a subset of the edges in the order and direction given by the giant tour.

As for our algorithm, the giant tour is constructed using the 3/2-approximation algorithm summarized in Section 2. Now, let m be the number of demand edges and let $e^{(1)}$, $e^{(2)}$, $e^{(3)}$, . . ., $e^{(m)}$ be these edges in the order by which they occur on the giant tour, where there may be a shortest path edge between any $e^{(k)}$ and $e^{(k+1)}$.

Let G_D be a directed graph with $m + 1$ vertices denoted by 0, 1, 2, . . .m. The cost d_{rs} of an arc (r, s) in G_D equals the cost of a tour starting in the depot node, servicing edges $e^{(r+1)}$ . . . $e^{(s)}$ in the order and direction of the giant tour, and returning to the depot. The cost d_{rs} is only defined for legal tours, i.e. when $r < s$ and $\sum_{k=r+1}^{s} q(e^{(k)}) \le W$, where $q(e^{(k)})$ denotes the demand of edge $e^{(k)}$. If we let $v_1^{(k)}$ $v_2^{(k)}$ denote the source (target) of $e^{(k)}$, as we meet them when passing along the giant tour, then we have the following costs:

$$d_{rs} = \sum_{k=r+1}^{s-1}(c(e^{(k)}) + SPL(v_2^{(k)}, v_1^{(k+1)})) + SPL(0, v_1^{(r+1)}) + c(e^{(s)}) + SPL(v_2^{(s)}, 0)$$

Next a shortest path problem is solved from node 0 to node m in G_D. Since the cost of an arc $(i \to j)$ in G_D equals the cost of servicing edges $e^{(i+1)}$ through $e^{(j)}$ in that order, a path from node 0 to node m must give the cost of servicing all the edges in the order specified. Each arc on the path corresponds to one vehicle tour. Since all legal tours are included in G_D, a shortest path from 0 to m will give us an optimal splitting of the initial tour with the corresponding minimum cost.

Jansen 1993 [12] proves the following approximation factor for the algorithm in terms of the GCRP.

Theorem 5. *If the initial tour is an α-approximate RPP tour, then SOTP for CARP has performance ratio* $\frac{c(SOTP)}{c^*(OPT)} \le 2 + \alpha(1 - 2/W)$.

Using the $\frac{3}{2}$-approximation algorithm for RPP the performance ratio of the SOTP algorithm for the CARP is $\frac{7}{2} - \frac{3}{W}$.

It is not hard to see that the two algorithms, SOTP and A-ALG are very similar. They start by constructing the same giant tour, and proceed by optimally partitioning this tour into vehicle tours. The proof that A-ALG outperforms SOTP is based on the fact that whenever the algorithms partition

the RPP tour into vehicle tours, A-ALG is allowed to split in any way that SOTP is, but because AALG is allowed to service some of the edges in the opposite direction, A-ALG has many more possibilities when constructing the solution. In other words, the solution space for SOTP is strictly contained in the solution space for A-ALG. This means that even if the RPP tour is split in the same places by the two algorithms, A-ALG may be able to make the resulting solution cheaper, and we have the following theorem.

Theorem 6. *If the same initial RPP tour is used then A-ALG outperforms SOTP, i.e.* $c(A\text{-}ALG) \leq c(SOTP)$.

It is not hard to see that the same result holds, if A-ALG is used for the General Capacitated Routing Problem (GCRP) as well. The ability for A-ALG to obtain strictly better results than SOTP is based on the option of servicing edges in the opposite direction. Therefore, the presence of demand nodes in GCRP does not favor one of the two algorithms above the other. An immediate consequence of Theorem 6 along with Theorems 4 and 5 is

Corollary 7. *A-ALG is a* $(7/2 - 3/W)$*- approximation algorithm both for the CARP and for the GCRP, where C satisfies the triangle inequality.*

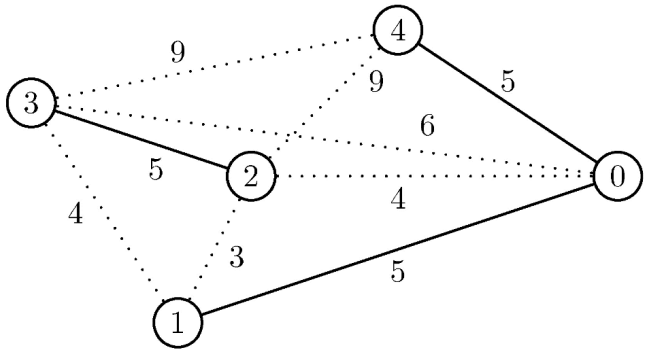

Fig. (3). Example.

The graph has two connected components, and hence $MST = \{(1, 2)\}$, $\Theta = \{3, 4\}$, and $M = \{(3, 4)\}$. The constructed Euler tour becomes 0-1 2- 3 4-0, where a dash indicates service. The total cost of this tour is 27. Partitioning the tour according to the two algorithms gives us the following solutions. For SOTP two tours are constructed: 0-1 2-3 0 and 0 4-0 with total cost of 29. For A-ALG the tours become 0-1 3-2 0 and 0 4-0, with total cost of 28.

4. COMPUTATIONAL RESULTS AND CONCLUSION

In the previous section we proved that A-ALG performs at least as well as SOTP and we gave an example in which A-ALG is strictly better. In this section we show that A-ALG performs significantly better than SOTP in practice and is indeed competitive to a set of well-known problem-specific heuristics for the CARP.

We have tested A-ALG on the four standard sets of benchmark instances (the GDB, KSHS, Val, and Eglese instances) and on two additional sets of instances. In total the algorithms are tested on 143 instances. The results of a comparison between A-ALG and SOTP are shown in Table **1**, whereas Table **2** summarizes the results of a comparison between A-ALG and a set of problems-specific heuristics for the problem. Data for the instances and detailed computational results are available at http://www.hha.dk/~sanw.

The results show that A-ALG performs strictly better than SOTP for all 143 instances. In 25 instances, A-ALG obtains the best known solution and in 8 instances A-ALG reaches a proved optimal solution. The results obtained regarding average performance also favour A-ALG. The average result obtained with SOTP was 22.4% above the best known lower bound, whereas the average result obtained with A-ALG was 15.4% above the best known lower bound[1].

Besides giving the number of instances in each set, Table **1** provides a comparison between SOTP and A-ALG. Column three gives the average percentage of the improvement obtained by using A-ALG instead of SOTP, and the next column gives the standard deviation of this improvement. Column 5 (6) provides the smallest (largest) percentage-wise

Table 1. Comparison between SOTP and A-ALG for the 143 Instances

Set	Number of Instances	Average % Better	Std. Deviation of % Better	Min % Better	Max % Better	SOTP % Above	LBA-ALG % Above LB
GDB	23	5.7	3.9	1.2	14.4	15.2	8.5
KSHS	6	8.3	3.9	4.2	14.5	20.9	10.8
Val	34	2.9	1.8	0.4	7.9	16.3	12.8
Eglese	24	6.8	6.5	0.6	20.4	23.6	14.7
A	32	4.8	2.4	0.8	10.4	24.0	18.0
B	24	8.0	2.4	2.6	13.1	34.8	24.2
Total	143	5.5	4.0	0.4	20.4	22.4	15.4

We have proved the promised approximation factor for A-ALG and showed that it always performs at least as well as SOTP. In Fig. (**3**) we give an example of a graph where A-ALG performs strictly better than SOTP. Numbers on the edges indicate cost. Let the three solid edges be demand edges each with demand of 5 and let the vehicle capacity be 10.

[1]It should be noted that for many instances in set A and set B, the optimal solution is not known.

Table 2. Comparison of the Problem-Specific Heuristics for the 143 Test Instances

	Classical Heuristics				Resent Heuristics			
	Augment Insert	**Augment Merge**	**Path Scanning**	**SOTP**	**Mod Path Scanning**	**DB O. Scan**	**ND Heu**	**A ALG**
Average Rank	7.8	4.2	4.2	5.2	4.1	4.9	2.4	2.2
Average % above LB	21.2	24.2	22.4	20.2	26.7	15.4	15.4	62.2

improvement observed. The final two columns give the average percent above the best known lower bound of the results obtained with each of the two algorithms.

In Table **2** we provide a summary of a comparison of A-ALG to a set of well-known problem-specific heuristics for the CARP. In addition to SOTP, these include three widely used classical problem-specific heuristics (Augment-Insert, Augment-Merge, and Path Scanning) and three recent algorithms presented in Wøhlk 2005 [20] (Modified Path Scanning, Double Outer-Scan, and Node Duplication Heuristic). The first row gives the average rank of the results obtained for the 143 instances, where rank 1 is given to the algorithm which obtained the best result. The second row gives the average percentage above the best known lower bound obtained with each algorithm. These results indicate that AALG is highly competitive to a large set of problem-specific heuristics for the problem. It should be noted that all of the algorithms mentioned are often outperformed by meta heuristic approaches.

We conclude that the approximation algorithm, A-ALG is interesting both from a theoretical point of view with a proven approximation factor that is at most equal to the best one previously known, and from a practical point of view since, as justified above, A-ALG is highly competitive to the set of existing problem-specific heuristics.

REFERENCES

[1] Golden BL, Wong RT. Capacitated arc routing problems. Networks 1981; 11: 305-315.

[2] Hertz A, Laporte G, Mittaz M. A tabu search heuristic for the capacitated arc routing problem. Oper Res 2000; 48(1): 129-135.

[3] Lacomme P, Prins C, Ramdana-Chérif W. Competitive genetic algorithms for the capacitated arc routing problem and its extensions. Lect Notes Comput Sci 2001; 2037: 473-483.

[4] Lacomme P, Prins C, Ramdane-Chérif W. Competitive memetic algorithms for arc routing problems. Ann Oper Res 2004; 131: 159-185.

[5] Hertz A, Mittaz M. A variable neighborhood descent algorithm for the undirected capacitated arc routing problem. Transpor Sci 2001; 35(4): 425-434.

[6] Assad AA, Golden BL. Arc routing methods and applications. In: Ball MO, Magnanti TL, Monma CL, Nemhauser GL, editors. Handbooks in Operations Research and Management Science, Vol. 8-Network Routing. Elsevier; 1995.

[7] DrorM, Ed. Arc Routing-Theory, Solutions and Applications. Kluwer Academic Publishers; 2000.

[8] Wøhlk S. A decade of capacitated arc routing. In: Golden BL, Raghavan S, Wasil EA, Eds. The Vehicle Routing Problem: Latest Advances and New Challenges. Springer; 2008. .

[9] Gonzales T, Sahni S. P-complete approximation problems. J Assoc Comput Machinery 1976; 23(3): 555-565.

[10] Christofides N. Worst-Case Analysis of a new heuristic for the travelling salesman problem. report 388, Graduate School of Industrial Administration, CMU, 1976.

[11] Assad AA, Golden BL, Pearn WL. Transforming arc routing into node routing problems. Comput Oper Res 1987; 14(4): 285-288.

[12] Jansen K. Bounds for the genaral capacitated routing problem. Networks 1993; 23: 165-173.

[13] Beasley JE. Route first-cluster second methods for vehicle routing. OMEGA Int J Manag Sci 1983; 11(4): 403-408.

[14] Arkin EM, Hassin R, Levin A. Approximations for minimum and min-max vehicle routing problems. J Algorithms 2006; 59.

[15] Chalasani P, Motwani R. Approximating capacitated routing and delivery problems. SIAM J Comput 1999; 28.

[16] Katoh N, Yano T. An approximation algorithm for the pickup and delivery vehicle routing problem on trees. Discrete App Math 2006; 154.

[17] Kan AHGR, Lenstra JK. On general routing problems. Networks 1976; 6: 273-280.

[18] Frederickson GN. Approximation algorithms for some postman problems. J Assoc Comput Machinery 1979; 26(3): 538-554.

[19] Jansen K. An approximation algorithm for the general routing problem. Inf Process Lett 1992; 41: 333-339.

[20] Wøhlk S. Contributions to Arc Routing. Ph.D. thesis. University of Southern Denmark, 2005.

Received: October 28, 2007 Revised: December 5, 2007 Accepted: January 25, 2008