Available online at www.sciencedirect.com
ScienceDirect
Procedia Computer Science 31 (2014) 910 – 917
Procedia Computer Science

2nd International Conference on Information Technology and Quantitative Management, ITQM 2014

Routing of supply vessels to with deliveries and pickups of multiple commodities

Eugen Sopot*, Irina Gribkovskaia

Molde University College, P.O. Box 2110, 6402 Molde Norway

Abstract

This paper considers *a single vehicle routing problem with pickups and deliveries of multiple commodities* where each customer requires both pickup and delivery of several types of goods from a single depot. This problem arises in offshore upstream logistics and is relevant for the oil and gas companies operating offshore. Offshore installations need to be supplied with several types of goods from an onshore base, and also some cargo need to be transported from the installations back to the base. Supply operations from and to the base are performed by supply vessels, which have separate compartments for different types of cargo. In this paper we present a mathematical formulation for the problem and describe a metaheuristic algorithm yielding non-Hamiltonian routes where customers may be visited once or twice. Computational tests show that the algorithm outperforms CPLEX optimization solver in speed on instances of medium size and generates high quality solutions for large-size instances compared to the Unified Tabu Search algorithm.

Keywords: vehicle routing; pickup and delivery; multiple commodities; metaheuristics; offshore logistics

1. Introduction

Offshore installations are supplied from onshore bases using a fleet of vessels. Assignment of installations to bases is normally fixed and decided by the authorities. Supply operations from a single base are performed according to a weekly vessel sailing plan where vessels' voyages are scheduled based on requirements from the

* Corresponding author. Tel.: +47 47 71 95 53.
E-mail address: eugen.sopot@himolde.no (E. Sopot), irina.gribkovskaia@himolde.no (I. Gribkovskaia).

installations. Usually, from 3 to 15 installations are serviced by a vessel during one voyage. Each installation has a demand for several types of cargo to be delivered and picked up. Goods to be delivered from a base could be materials, fuel, food and equipment, while wastes and empty containers are picked up and return back to the base. Supply vessels have separate compartments for each type of goods, for example deck for a deck cargo and tanks for liquids under it. Pickup and delivery demands of installations not being fulfilled may lead to delayed or halted operations on the installation and substantial losses. That is why routes for supply vessels should be carefully planned.

The problem of planning a route for supply vessel performing both pickups and deliveries of multiple types of commodities can be described as follows. We consider a single depot, which is an onshore base, and a set of customers representing offshore installations. Each customer has both pickup and delivery demands for several types of cargo. A supply vessel starts its route from the depot, visits customers in a sequence fulfilling all customers' demands, and returns back to the depot. The supply vessel has separate compartments for several types of cargo. It is assumed that the total delivery and the total pickup demands of all customers served on the route do not exceed vessel's capacity for each corresponding cargo type. All the delivery demands are loaded at the depot and transshipments of cargo between customers are not allowed. During a visit to a customer the vessel has to completely fulfill pickup and/or delivery demand of the customer for a particular type of cargo or not fulfill some of them at all, split deliveries and pickups are not allowed. Each customer may be visited once or twice on the route. During the first visit the vessel performs both deliveries and pickups for all types of cargo, if it is possible; otherwise it performs only deliveries while pickup demands are fulfilled during the second visit. The reason for this assumption will be explained below. Travel distances between all customers and the depot are given and transportation costs are linearly dependent on them. The objective is to find a least cost feasible route satisfying vessel's capacity constraints for all types of cargo along the route.

For the problem formulated above it is not always possible to find a feasible Hamiltonian solution where each customer is visited exactly once. An example of the problem instance with four customers and two commodities, which has no Hamiltonian solution, is given in Table 1. In this instance capacity of each vehicle compartment is 25 units, and the total delivery and pickup demands for each commodity are equal to 25. When vessel starts from the depot fully loaded, the only customer which can receive both pickup and delivery services in a single visit without vessel capacity violation is Customer 1, as the other customers have pickup demand larger than delivery demand for at least one type of commodity. After servicing Customer 1 the vessel has two units of a free space in each compartment. However, it cannot service any other customer during a single visit without capacity violation, because three units of a free space for Commodity 2 are required to service Customer 2 and Customer 3, and four units of a free space for Commodity 1 are required to service Customer 4.

Table 1. Example of problem instance with no Hamiltonian solution

Customer	Commodity 1		Commodity 2	
	delivery demand	pickup demand	delivery demand	pickup demand
1	5	3	6	4
2	7	6	7	10
3	9	8	1	4
4	4	8	11	7

However, if in the single vehicle routing problem with pickups and deliveries two visits at customers are allowed, it is always possible to find a feasible general solution[1], which encompasses other known solution shapes, such as Hamiltonian, double-path and lasso. In a double-path solution, all customers are visited by means of two Hamiltonian paths: one from the depot to some customer, and a second one from this customer to the depot. In this solution only the last customer on the first path has a simultaneous pickup and delivery, while all other customers are visited twice. In a lasso solution, the vehicle first performs deliveries to a subset *S* of customers, then visits all remaining customers to perform a combined delivery and pickup, and finally performs collections at the customers of set *S*.

In the problem with multiple commodities it is always possible to build a feasible solution with maximum two visits to each customer. Example of such feasible solution is a double-path solution where delivery demands for all commodities are satisfied simultaneously during the first visit (as there are no advantages in postponement of deliveries to the second visit) and all the pickup demands are satisfied during the second visit (as there is no need to satisfy a part of the pickup demands at the first visit). For the problem instance given in Table 1, an example of a feasible lasso solution with $S = \{1\}$ is shown in Figure 1. In this solution the vehicle first visits Customer 1 for deliveries, then visits three other customers for both pickups and deliveries, and on the way to the depot visits Customer 1 again for pickups.

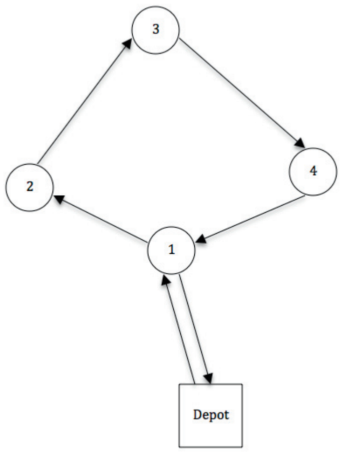

Fig. 1. Example of a lasso solution.

The problem studied in this paper can be classified as a single vehicle one-to-many-to-one pickup and delivery problem with multiple commodities and non-simultaneous services[2]. To the best of our knowledge, this problem has not been studied in the literature before. Several papers have been dedicated to the single commodity variant of the problem. A tabu search algorithm was developed for generation of lasso solutions for the travelling salesman problem with pickup and delivery[3]. A mathematical formulation and a Unified Tabu Search algorithm[4] yielding general solutions for the single vehicle routing problem with pickup and delivery were presented by Gribkovskaia et al[1]. A variant of the latter problem with selective pickups was studied and several heuristic algorithms were developed and tested[5]. Also, mathematical formulation and heuristic algorithms for a single vehicle problem with delivery and pickup and capacitated customers were presented[6].

2. Mathematical formulation

The single vehicle routing problem with pickups and deliveries of multiple commodities can be formulated as follows. Let $G = (V, A)$ be a complete graph with the set of vertices $= \{0, \dots, n\}$, where vertex 0 represents a depot and $1, \dots, n$ is the set of customers. The set of arcs is denoted by $A = \{(i,j): i,j \in V, i \neq j\}$. Each arc has a non-negative cost c_{ij}. We denote by $H = \{1, \dots, m\}$ a set of commodities, where m is a number of commodities. Each customer i has non-negative delivery demands d_{ih} and non-negative pickup demands p_{ih} for each commodity h. Vehicle has a compartment of capacity q_h for each commodity h.

For modelling purposes we assume that each customer i is represented by two vertices i and $+n$, where $p_{i+n,h} = p_{i,h}$. When all deliveries and all pickups are performed simultaneously at the single visit at customer i, vertex i is visited and vertex $i + n$ is not visited. Otherwise all deliveries are made at vertex i and all pickups at vertex $i + n$. We define costs $\bar{c}_{ij}$ on the extended arc set as follows: $\bar{c}_{ij} = c_{ij}$, if $i \leq n$ and $j \leq n$; $\bar{c}_{ij} = c_{i-n,j}$, if $i > n$ and $j \leq n$; $\bar{c}_{ij} = c_{i,j-n}$, if $i \leq n$ and $j > n$; and $\bar{c}_{ij} = c_{i-n,j-n}$, if $i > n$ and $j > n$.

Binary variable X_{ij} $(i,j = 0,..,2n; i \neq j; j \neq i+n \; if \; 1 \leq i \leq n; j \neq i-n \; if \; i > n)$ indicates if vertex j is visited directly after vertex i, in this case it is equal to 1 and to 0 otherwise. Binary variable Y_{jh} $(j = 1,..,n; h = 1,..,m)$ is equal to 1 if delivery and pickup for commodity h are performed simultaneously at customer j, and is zero otherwise. Binary variable Z_i $(i = 1,..,n)$ is equal to 1 if customer i is visited once and is 0 if twice. Variables V_{jh} $(i = 1,..,n; h = 1,..,m)$ represent an upper bound on the vehicle load for commodity h after vertex j has been visited. The mathematical formulation is given below:

$$minimize \sum_{i=0}^{2n} \sum_{j=0}^{2n} \bar{c}_{ij} X_{ij} \quad (1)$$

$$\sum_{j=0}^{2n} X_{ij} = 1, \forall i = 0, \dots, n \quad (2)$$

$$\sum_{i=0}^{2n} X_{ij} = 1, \forall j = 0, \dots, n \quad (3)$$

$$V_{0h} = \sum_{i=1}^{n} d_{ih}, h \in H \quad (4)$$

$$0 \leq V_{ih} \leq q_h, \forall i = 0, \dots, 2n; h \in H \quad (5)$$

$$V_{jh} \geq V_{ih} - d_{jh} + p_{jh} Y_{jh} - (1 - X_{ij}) q_h, \forall i = 0, \dots, 2n; \forall j = 1, \dots, n; h \in H \quad (6)$$

$$V_{jh} \geq V_{ih} + (1 - Y_{j-n,h}) p_{j-n,h} - (1 - X_{ij}) q_h, \forall i = 0, \dots, 2n; \forall j = n+1, \dots, 2n; h \in H \quad (7)$$

$$Z_i \leq \frac{\sum_{h \in H} Y_{ih}}{m}, \forall i = 1, \dots, n \quad (8)$$

$$\sum_{i=0}^{2n} X_{ij} = 1 - Z_{j-n}, \forall j = n+1, .., 2n \quad (9)$$

$$\sum_{j=0}^{2n} X_{ij} = 1 - Z_{i-n}, \forall i = n+1, \dots, 2n \quad (10)$$

$$X_{ij} \in \{0,1\}, i \neq j; i \neq j+n; i \neq j-n; \forall i,j = 0, \dots, 2n \quad (11)$$

$$Y_{ih} \in \{0,1\}, \forall i = 1, \dots, n; \; h \in H \quad (12)$$

$$Z_i \in \{0,1\}, \forall i = 1, \dots, n \quad (13)$$

$$V_{ih} \geq 0, \forall i = 1, \dots, 2n; h \in H \quad (14)$$

The objective function (1) expresses the minimization of the total travel cost. Constraints (2) and (3) guarantee that the first vertex associated with a customer is visited once, either for all deliveries, or for both all pickups and all deliveries. Constraints (4) define the load of the vehicle for each commodity upon leaving the depot. Constraints (5) guarantee that capacity of the vehicle for each commodity is not violated. Constraints (6) and (7) are load continuity constraints. In constraints (6), j is the first vertex associated with customer j: $V_{jh} \geq V_{ih} - d_{jh} + p_{jh}$ if pickup and delivery are performed simultaneously at j, and $V_{jh} \geq V_{ih} - d_{jh}$ otherwise. In constraints (7), j is the second vertex associated with customer j: $V_{jh} \geq V_{ih}$ if pickup and delivery are performed simultaneously at j and $V_{jh} \geq V_{ih} + p_{jh}$ otherwise.

Constraints (8), (9) and (10) express the fact that the second vertex associated with a customer is visited only if a combined pickup and delivery does not occur at the first vertex. Constraints (11), (12) and (13) are binary requirements for variables X, Y and Z respectively. Constraints (14) represent lower bounds on vehicle load of each commodity respectively.

3. Algorithm description

As the considered problem extends the well-known Traveling Salesman Problem, it is NP-hard. That is why metaheuristic algorithm is developed to solve it within a reasonable time. The idea of our algorithm is based on the assumption, that for every given solution, except the optimal one, there is a neighbourhood, where a solution better than the given one is reachable by only one move. To find this solution new neighbourhood structures are constructed at each iteration and evaluated, expecting that there is a move to a better solution.

A route is presented as a sequence of visits. A random double-path solution is taken as the initial because it is always possible to find a feasible double-path solution for this type of problem. The main part of the algorithm starts from generating a new neighborhood structure. For doing this current solution is randomly divided into several blocks of sequential visits. The number of blocks and the size of each block are set randomly at each iteration. Average number of blocks is controlled by a parameter denoted as r. During computational experiments this parameter was set to square root of n, where n is a number of visits in a current solution.

At the evaluation stage all possible neighbours of a current solution, which can be obtained by swapping blocks, are generated. After swapping blocks there is a possible situation when pickup visit of some customer node is placed in a route directly before the delivery visit of the same node. This is impossible in reality because vessel delivers first to empty a space in a compartment. Moreover, as the algorithm calculates load of a vessel by scanning visits one by one, this situation can force it to incorrectly consider a solution as infeasible. To avoid this where is a special procedure implemented. The procedure scans a solution, detects pairs of delivery and pickup visits of a node placed one after another and swaps them.

After that cost function including penalty for capacity violation is calculated for each obtained solution. Then Simulated Annealing acceptance criterion driven by acceptance probability factor denoted as z is applied to the best found neighbour. Thus, “best fit” approach is used, but as solution at this point is represented as a set of blocks “best fit” is not very computationally expensive. Finally, if solution obtained is the best found yet, it is stored.

Two additional procedures are sequentially applied to the solution. First procedure scans the current solution and randomly merges pairs of sequential visits of one node if they are found. Second one also scans the solution, and randomly split visits. Both procedures are driven by the parameter s called “split factor”, probability of visits to be split is equal to this parameter, while probability of pair of visits to be merged is equal to one minus this parameter. The purpose of this mechanism is to improve flexibility of search without duplicating all customer nodes and thus increasing the number of visits in a current solution. The main part of the algorithm is applied for a fixed number η of iterations, where $\eta=100n^2$.

The pseudo-code of our algorithm is given below:

```
**input:** feasible double-path solution $x$, split factor $s \in (0,1)$
$n$ is number of visits in $x$;
$x^b = x$;
$z = 1$;
**repeat**
    $c'$ is very large number;
    $r$ is random number close to $\sqrt{n}$;
    randomly divide $x$ into $r$ blocks of visits;
    **for** each $i$ from 1 to $r$ **do**
        **for** each $j$ from $i$ to $r$ **do**
            $x_{i,j}$ is solution obtained by swapping $i^{th}$ and $j^{th}$ blocks in $x$;
            **if** $c(x_{i,j}) < c'$ **then**
                $c' = c(x_{i,j})$;
                $a = i$;
                $b = j$;
            **end if;**
        **end for;**
```

```
end for;
if $zc(x) > c(x_{a,b})$ then
    $x = x_{a,b}$;
    $z = 1$;
else
    $z'$ is a random number close to zero;
    $z = z + z'$;
end if
if $c(x) < c(x^b)$ then
    $x^b = x$;
end if
for each visit $v$ do
    $s'$ is random number from 0 to 1;
    if $s' > s$ and visits $v$ and $v - 1$ belong to the same customer then
        merge $v$ and $v - 1$;
 end if
    end for
 for each visit $v$ do
     $s'$ is random number from 0 to 1;
     if $s' < s$ and both pick-up and delivery are performed during $v$ then
         split visit into two visits: first with delivery and second with pick-up;
        end if
    end for
 until $\eta$ iterations
 return $x^b$
```

4. Computational experiments

The algorithm described above is implemented in C using Xcode. All tests were run on a Mac computer with Intel i7 CPU and 16Gb of RAM. Two experiments on two sets of problem instances with two commodities were performed.

The purpose of the first experiment is to measure the performance of our algorithm against CPLEX on instances of practical size (7 to 11 customer nodes). Instances, which include less than 7 customer nodes, were not tested because it takes less than one second both for CPLEX and for our algorithm to solve them to optimality. Instances which include more than 11 customer nodes were not tested because it takes about 48 hours to obtain an optimal solution for a problem instance with 12 customer nodes using CPLEX. Instances for this experiment were generated randomly. Depot and customer nodes are placed in a square of 200x200. Pickup and delivery demands are generated according to the following rule: for each customer node the pickup demand for at least one of the commodities is larger than the delivery demand. The vessel capacity for each commodity is equal to the total delivery demand and/or the total pickup demand. Instances of this type have no Hamiltonian solution because it is always required to visit at least one customer twice: first to deliver cargo and get free space in a compartment and second to pick up cargo.

Table 2. Results of the first experiment

Instance size	Number of instances	CPLEX average time, sec	Our algorithm average time, sec
7	1000	1,5	1,5
8	100	3,9	2,1
9	100	10,7	4,1
10	50	229.2	6,2
11	10	4455,9	9,8

Results of the first experiment are given in Table 2. All solutions obtained by our algorithm are optimal. Tests show that our algorithm is able to yield optimal solutions for medium sized problem instances and outperforms CPLEX in terms of speed.

The purpose of the second experiment is to compare performance of our algorithm in terms of accuracy on larger instances with a version of Unified Tabu Search Algorithm (UTSA)[4], developed originally for the single commodity case of single vehicle routing problem with pickups and deliveries and modified to handle demand for multiple commodities[1].

Instances for this experiment were generated from CVRP instances taken from VRPLIB. For each customer its delivery demand and pickup demand are calculated as follows: delivery demand for the first commodity is set to the demand from an original CVRP instance, pickup demand for the first commodity is set to 80% of delivery demand for each even customer node and 120% for each odd one. Delivery and pickup demands of the second commodity are the reversed delivery and pickup demands of the first commodity. Size of the instances varies from 15 to 100 customer node. The instances are named as Xc_Y where X is a number of commodities and Y is a number of nodes including depot in a problem instance. Cost gap between results is calculated as $100 \cdot (U - O)/U$, where U is a cost of solution obtained by UTSA and O is a cost of solution obtained by our algorithm. Time gap is calculated as U^t/O^t, where U^t and O^t are execution times of UTSA and our algorithm on the same hardware respectively. Results of the second experiment are given in Table 3.

Table 3. Results of the second experiment

Instance	Customer nodes	Our algorithm, cost	UTSA, cost	Cost gap, %	Time gap, times
2c_016	15	220,74	220,99	**-0,11**	1,76
2c_021	20	255,71	271,69	**-5,88**	2,16
2c_022	21	283,09	286,68	**-1,25**	2,17
2c_023	22	489,99	484,11	1,21	1,84
2c_026	25	319,03	312,73	2,01	1,94
2c_031	40	323,69	329,11	**-1,65**	2,58
2c_041	40	379,61	376,48	0,83	1,51
2c_045	44	626,67	622,75	0,63	1,64
2c_048	47	33600,56	34302,34	**-2,05**	1,99
2c_051	50	441,96	443,08	**-0,25**	1,71
2c_072	71	208,57	212,42	**-1,81**	3,81
2c_076	75	577,97	569,36	1,51	4,33
2c_101	100	683,84	668,51	2,29	8,07
			Average:	**-0.35**	**2,73**

As shown in Table 3, our algorithm is comparable in the accuracy with UTSA and yields solutions on average 2.73 times faster.

5. Conclusion

In this paper we have considered the problem of routing supply vessels to offshore installations where delivery and pickup of multiple types of commodities is required and have proposed a metaheuristic algorithm for solving it. As it is demonstrated by computational experiments, the proposed algorithm exhibits ability to obtain optimal solutions for problem instances of a practical size within a reasonable time. Moreover, it is also able to solve larger instances with the competitive accuracy but faster than UTSA.

The proposed algorithm can be extended to solve more complex and feature-rich problems, which have important practical applications within oil and gas industry, including but not limited to problems with multiple vehicles, multiple depots, multiple time windows and capacitated customers.

References

1. Gribkovskaia I, Halskau O, Laporte G, Vlcek M. General solutions to the single vehicle routing problem with pickups and deliveries. *Eur J Oper Res* 2007;**180**:568–584.
2. Berbeglia G, Cordeau JF, Gribkovskaia I, Laporte G. Static pickup and delivery problems: A classification scheme and survey. *TOP* 2007;15: 1-31.
3. Hoff A, Lokketangen A. Creating lasso-solutions for the traveling salesman problem with pickup and delivery. *Cent Eur J Oper Res* 2006;**14**:125–140.
4. Cordeau JF, Laporte G, Mercier A. A unified tabu search heuristic for vehicle routing problems with time windows. *J Oper Res* 2001;**52(8)**:928-936.
5. Gribkovskaia I, Laporte G, Shyshou A. The single vehicle routing problem with deliveries and selective pickups. *Comp & Oper Res* 2008;35**(9)**:2908-2924.
6. Gribkovskaia I, Laporte G, Shlopak A. A tabu search heuristic for a routing problemarising in the servicing offshore oil and gas platforms. *J Oper Res Soc* 2007;**59**:1449-1559.