HEURISTIC SOLUTION OF THE SINGLE VEHICLE PICKUP AND DELIVERY PROBLEM

E.M. Bronstein
Department of Computer Science and Robotics
Ufa State Aviation Technical University
Ufa, Russia
e-mail: bro-efim@yandex.ru

R.V. Gindullin
Institute of Economics, Finances and Business
Bashkir State University
Ufa, Russia
e-mail: gramiz@mail.ru

Abstract[1]

The Pickup and delivery problem (PDP) with single vehicle (SPDP) with capacity constraints is considered. The problem requires constructing the shortest cyclic route for delivery of homogeneous cargo from all the producers to specific customers with one capacitated vehicle. Such problem, for example, could be used for transporting passengers (like taxi station). Enumeration procedure for finding approximate solution is developed. Efficiency of developed procedure is empirically analysed and compared to the efficiency of exact methods. Quality of the solution and elapsed time of that heuristic with different numbers of iterations, which don't improve solution's length by using that procedure, is analysed.

1. Introduction

First variant of vehicle routing problem (VRP) was the truck dispatching problem [1]. Many modifications of the VRP were examined since then. Information about VRP research is accumulating in [2].

A variant of VRP, called the pickup and delivery problem (PDP), is examined in the article. More precisely, PDP with single vehicle of limited capacity (SPDP). That problem requires full graph, nods of which are clients (both consumers and providers), arcs are corresponding routes, and single vehicle of limited capacity, that has to pick-up cargo of different weights from providers, with the consideration, that the cargo from a specific producer must be delivered to a specific customer. Shortest route, that satisfy all the constraints, must be constructed. This is NP-hard problem. SPDP is also called Pickup-Delivery Traveling Salesman Problem and Traveling Salesman Problem with Pickups and Deliveries [3].

SPDP without vehicle capacity constraint and cargo weights was solved with exact approach in [4]. Example with 15 nods was solved with the combination of branch-and-cuts and greed search algorithms.

[5] and [6] used different approaches with perturbations in cycles to solve SPDP without vehicle capacity constraint and cargo weights. Approaches were used on group of 108 examples from TSPLIB, including the example with 441 nods.

Similar problems were also considered:

Pickup and delivery problem with time windows: variant of the problem, with added time constraints. Branch-and-bounds and branch-and-cut (with column generation) algorithms were successfully used in [7]. In [8] this approach was used for the problem with several vehicles. Models with polynomial number of constraints are called compact.

The Pickup and Delivery Traveling Salesman Problem or *Traveling Salesman Problem with Pickups and Deliveries*: there is single vehicle of limited capacity, cargo is picked up from multiple producers and delivered to multiple consumers, with no constraints of order of visitations. Every nod is visited once. In [9], branch-and-cut method was used to solve this problem.

Dial-a-ride problem: unlike PDP, the goal of DARP is the minimization of client's waiting time. Review of algorithms designed for DARP is presented in [10].

In [11], several formalizations of capacitated SPDP were designed, including adaptations of algorithms [12, 13] that were used for quadratic assignment problem [14]. In this article, results of [11] are used to compare effectiveness of proposed heuristic approach.

2. Problem definition

Assume, that $P=\{1,\ldots,n\}$ – nodes of cargo pickups and $D=\{1+n,\ldots,2n\}$ – nodes of cargo deliveries. For the node i from P, the amount of cargo that needs to be picked up is equal q_i, and then, for the node $i+n$ from D, amount of cargo that needs to be delivered is equal to the negative amount ($-q_i$). Set of all the nodes is $V = P \cup D \cup \{0\}$, where $\{0\}$ is the depot. Vehicle with the capacity S must visit all the nodes once and deliver cargo from the nodes i to the nodes $i+n$. Route must start and end in the depot. Distances between all pairs of nodes (c_{ij}) are known. Problem is NP-hard, because if we assume $S=q_1=\ldots=q_n=1$, then the problem is the equivalent of the asymmetric traveling salesman problem with $n+1$ nodes.

Proceedings of the 6th All-Russian Scientific Conference "Information Technologies for Intelligent Decision Making Support", May 28-31, Ufa - Stavropol, Russia, 2018

Some properties of the problem are formulated.

Minimal allowed vehicle capacity is equal $\max\{q_i\}$. Clearly, no route could be made if $S<\max\{q_i\}$. With $S=\max\{q_i\}$ there are at least $n!$ allowed routes, for example: $0 - 1 - (1+n) - 2 - (2+n) - \ldots - n - 2n - 0$.

If vehicle capacity is unlimited, then the number of the allowed routes is equal $(2n)!/2^n$. Clearly, number of all permutations is equal $(2n)!$, and each allowed permutation leads to 2^n permutations by switching places of nodes i and $i+n$ for all $i=1,2,\ldots,n$. Same result was achieved in [4] by more complex reasoning.

With $S\geq\max\{q_i\}$ any allowed section of a route could be continued. For example, if vehicle is empty and there are unvisited clients, then vehicle can move to some node i to pick up its cargo. If there is some cargo in vehicle, that means, some node i is visited, but not the node $i+n$ and vehicle can deliver that cargo to that node.

Problem formalizations are listed in [11]

3. 4-opt-o procedure

Literature describes different enumeration algorithms, and most famous of them are 2-opt and 3-opt for traveling salesman problem. The idea of 2-opt and 3-opt algorithms is that 2 or three arcs, that cross each other, are removed, and instead replaced by arcs that don't cross each other [15, 16].

4-opt algorithms, in it's different variations, deletes 4 arcs from the cycle by certain rules [5, 6, 17].

Cycle is called proper, if, during the movement from the depot, for each i, node i is placed earlier than node $i+n$, i.e. delivery of cargo in this cycle is possible.

Proposed procedure, analogous to [17], that reforms proper cycle in such way, that:

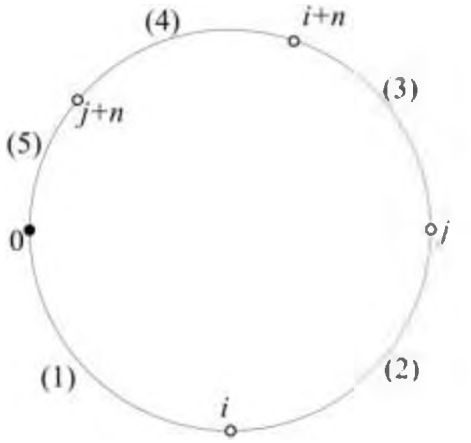
1. proper cycle is constructed;

2. after using the procedure, from every proper cycle it is possible to get any proper cycle. This will guarantee algorithm convergence to the optimal cycle.

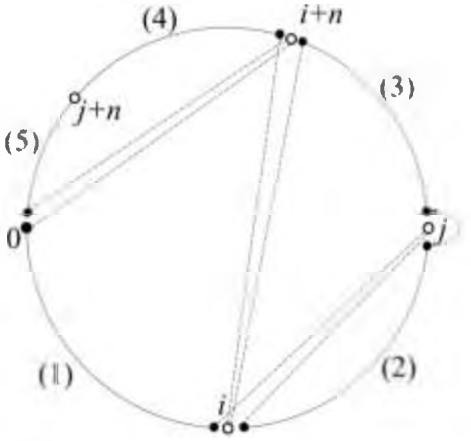
4-opt-o procedure is organized as follows:

Randomly select two pair of nodes (i, $i+n$, j, $j+n$; hereafter $i\neq j$), and delete all arcs that connect these nodes in the cycle. There will be left 5, 4, 3, 2 or 1 arch left from previous cycle. Let's enumerate these arch in order of visitation. Arch 1 always contains the depot.

Assume that 5 arcs left. This means, that there's no two of selected nodes, that are adjacent to each other. Proper cycle is created, if order of visitation of arcs 1, 2, 3, 4, 5 is remains. The nodes i, $i+n$, j, $j+n$ is placed in order of visitation in such way, that i (and j, accordingly) is placed before $i+n$ (and $j+n$, accordingly). For example, from source cycle $1–i–2–j–3–(i+n)–4–(j+n)$-5 (Pic. 1.a) it is possible to get cycles $1–j–2–3–i–4–(j+n)–5–(i+n)$ (Pic. 1.b), $1–j–i–2–3–(j+n)–4–(i+n)–5$, etc.



a)



b)

Picture 1 – Example of permutation of nodes and arcs

Maximal number of possible arcs in cycle is five. If the number of arcs is lower than five, the arc with biggest number of nodes is divided by half. This procedure repeated, until there's five arcs.

Number of proper cycles (excluding vehicle capacity constraint). In the sequence 1, 2, 3, 4, 5 let's add nodes i, $i+n$, j, $j+n$ (not necessarily assuring creation of a proper cycle). For example, it is possible to create sequences: 1, 2, 3, 4, i; 1, $i+n$, 2, 3, 4, i, 5; 1, $i+n$, 2, j, 3, 4, i, 5; 1, $j+n$, i, 2, j, 3, 4, 5, $i+n$. Total number of such combinations is $5\cdot6\cdot7\cdot8=1680$ (since arc 1 is always placed the first). They could be divided on quadruplets, with only differences between them is order of pairs i, $i+n$ and j, $j+n$. In each quadruplet, the only proper cycle is created in only one sequence (1, j, i, 2, $j+n$, 3, 4, $i+n$). As a result, the total number of proper cycles, that could be created, is 420. Shortest proper cycle, that doesn't break vehicle capacity constraints, is selected.

It should be noted, that 4-opt-o operation is reversible.

Proposition. From any proper cycle, it is possible to get any proper cycle by using 4-opt-o operation.

Proof.

1. With 4-opt-o operation it is possible to permute two adjacent nodes, that belongs to different pairs. For example, nodes $i+n$ and j are adjacent. If nodes i, $i+n$, j, $j+n$ are selected, then permutation of nodes $i+n$ and j with remaining nodes in the same position is possible.

2. Main proposition will be proved by induction. For two pairs of nodes the proposition is obvious. Assume, that it is true for cycles with k pair of nodes, and there's two proper cycles, P^1_{k+1} and P^2_{k+1}, that have (k+1) pairs of nodes. After removing nodes $(k+1)$, $(k+1)+n$, there will be proper cycles P^1_k, P^2_k, that contain k pair of nodes. By assumption of induction, there's a sequence of 4-opt-o operations, Q, that transforms P^1_k into P^2_k. Then, add nodes $(k+1)$, $(k+1)+n$ to each cycle in Q right after nodes k, $k+n$. obviously, all the cycles are proper. Denote initial and final cycles as $P^{(1)}_{k+1}$, $P^{(2)}_{k+1}$. Using transformation from paragraph 1, transform P^1_{k+1} into $P^{(1)}_{k+1}$. To done this, it is necessary to move nodes $(k+1)$, $(k+1)+n$, while saving placement of other nodes. Consider different placements of $(k+1)$, $(k+1)+n$ in cycles P^1_{k+1} (same denomination) and cycles $P^{(1)}_{k+1}$ (denote as $(k+1)^{(1)+}$, $(k+1)^{(1)-}$).

$(k+1)$, $(k+1)^{(1)+}$, $(k+1)+n$, $(k+1)^{(1)-}$. First, move $(k+1)+n$ in place of $(k+1)^{(1)+}$, then move $(k+1)$ in place of $(k+1)^{(1)-}$.

$(k+1)$, $(k+1)^{(1)+}$, $(k+1)^{(1)-}$, $(k+1)+n$. Same as the previous one.

Cases, when the first placed node is $(k+1)^{(1)+}$, also considered.

$P^{(2)}_{k+1}$ transformed into P^2_{k+1} same way. In the end, there will be a sequence of operations, that transforms P^1_{k+1} into P^2_{k+1}, which was to be proven.

4-opt-o procedure is applied, until solution is stabilized, i.e. if, during set number of iterations, there will be no cycle length reduction, then the process is stopped.

4. Computational experiments

To conduct numerical experiments, PC with *Intel Core i5-4670* 3.4*GHz*, 16*Gb RAM* and *Windows* 10 64*bit* was used. 4-opt-o procedure was realized in NetBeans IDE 8.1 (Java(TM) SE Runtime Environment 1.8.0_45-b14). To example with n pairs of nodes, initial proper cycle (0, 1, $1+n$, 2, $2+n$, …, n, $2n$) is considered. Stopping rule - if, during set k iterations, there will be no path length reduction, then the process is stopped.

For comparative analysis of the models, standard library of symmetrical examples, TSPLIB [18], was used.

Nodes coordinates are taken accordingly to data from example library. Hereafter, for examples with even number of nodes, last node is removed, first node is depot, first half of remaining nodes are producers (i), other half – consumers ($i+n$), all cargo weight is equal to 1, i.e. $q_i = 1$, $q_{i+n} = -1$, $i \in [1, n]$.

First series of experiments compares effectiveness of the procedure in the particular case (vehicle capacity is equal 1). Stopping rule – 100 iterations without solution improvement. Ten experiments were conducted for procedure 4-opt-o, with each launch having its own random generator seed number. Path length is compared to exact solution from [11]. Results are shown in Table 1.

Table 1 – Comparison of algorithms for particular case

n	Example	Exact solution		4-opt-o (10 experiments)				Error, %
		Path length	Time, sec	Average path length	Coefficient of variation for path length, V	Average number of iterations	Average time, sec	
7	ulysses16	135.4	0.083	135.4	0.004%	112.1	0.3029	0.003%
14	bayg29	19260	0.202	20216	2.75%	179.3	0.8453	4.965%
25	eil51	1078	0.230	1136	1.96%	284.2	4.096	5.427%
49	rat99	10984	0.660	10991	0.024%	926	42.05	0.063%
68	gr137	7900	1.650	7906	0.023%	812.6	62.94	0.085%
114	gr229	18303	8.7	18340	0.041%	1515.7	306.95	0.206%
199	rd400	116608	79.78	135555	1.69%	2070.8	1192.2	16.248%

It is clear from the results, that 4-opt-o takes significantly more time to solve particular case problem, than exact model. As a result, solving this type of problems with this approach is not effective.

In examples rat99, gr137 and gr229 path lengths, derived from 4-opt-o, are very close to optimal solutions, since coordinates of nodes in the examples are placed practically in optimal sequence for traveling salesman problem.

In other examples, where nodes are placed randomly, it is obvious, that with increasing complexity of the example, difference between exact solution and 4-opt-o solution will increase, up to 16.248% for n=199. Range of increase of path length, in this experiment, is from 13.83% to 20.22%.

Next series of experiments compares effectiveness of algorithms for general case of the problem. Coordinates and cargo weights are the same as previous ones, but vehicle capacity is equal n (S=n). Experiments are conducted to compare effectiveness of 4-opt-o procedure with the exact solution from [11]. Ten experiments were conducted for procedure 4-opt-o, with each launch having its own random generator seed number. Results are shown in Table 2.

Table 2 – Comparison of algorithms for general case

n	Example	Exact solution		4-opt-o (10 experiments)				Error, %
		Path length	Time, sec	Average path length	Coefficient of variation for path length, V	Average number of iterations	Average time, sec	
7	ulysses16	73,35	0,91	73,893	0,469%	117,3	0,5106	0,743%
10	ulysses22	85,03	168,83	87,552	4,65%	172,2	0,9407	2,961%

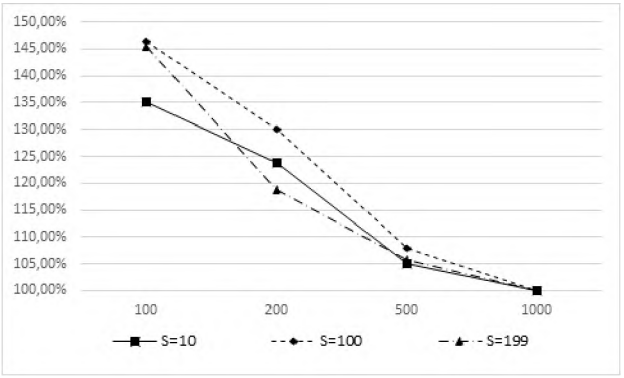
Unlike the particular case, usage of 4-opt-o procedure for general case is more advantageous in solution time, and gave results, that are close to optimal path length.

Next series of experiments are conducted to evaluate the behavior of 4-opt-o procedure, depending on vehicle capacity S and stopping rule. For each variant of the problem ten experiments were conducted, with each launch having its own random generator seed number. Average path length for k=1000 is accepted as 100% for each example. Results are shown in Table 3.

Table 3 – Comparison of 4-opt-o effectiveness with different stopping rule

S	k	Minimal path length, %	Maximal path length, %	Average path length, %	Coeffi-cient of variation for path length, V	Time, sec	Number of iterations
ch130.tsp, n=64							
64	32	125%	147%	136%	5.4%	50.1	722.6
	64	104%	127%	118%	6.8%	110	1557.6
	130	101%	119%	110%	5.2%	176	2459
	260	98%	112%	104%	4.1%	298	4319.8
	520	92%	105%	100%	3.7%	459	6284.4
	1000	95%	107%	100%	4.01%	587	8279.5
gil262.tsp.tsp, n=130							
130	64	133%	154%	143%	4.6%	830	2929.9
	130	111%	129%	120%	5.6%	1972	6856.6
	260	103%	123%	110%	5.2%	3173	12038
	520	97%	114%	102%	5.02%	5285	19308.1
	1000	91%	110%	100%	5.4%	7520	30012
rd400.tsp, n=199							
10	100	123%	146%	135%	5.3%	2733	4751.9
	200	113%	138%	124%	7.2%	5302	8772.8
	500	96%	114%	105%	4.6%	13467	22585.8
	1000	93%	104%	100%	3.8%	20911	36387.7
100	100	136%	155%	146%	4.7%	4167	7217.2
	200	121%	146%	130%	5.6%	7447	12630.2
	500	100%	117%	108%	4.3%	21097	34807.8
	1000	97%	105%	100%	2.5%	34479	57307.2
199	100	123%	171%	146%	9.8%	4922	7444.9
	200	110%	127%	119%	4.8%	8389	13996.1
	500	100%	112%	106%	4.5%	19801	32386.3
	1000	88%	108%	100%	5.9%	31795	51608.6

Graph of average path length in relation to on stopping rule for various vehicle capacities in example rd400 is shown in Picture 2.

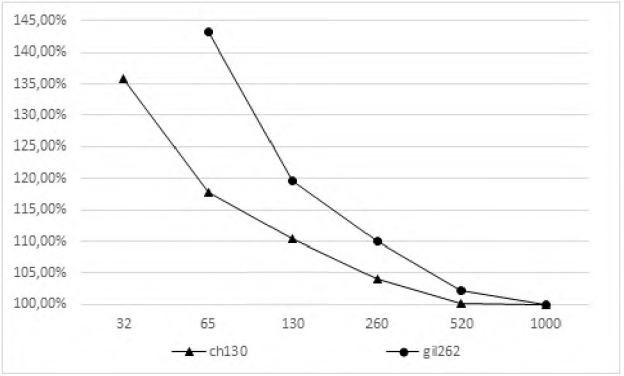


Picture 2 – Relation of average path length and number of iterations in stopping rule for rd400

k=500 has the increase of average path length of 5-7%, compared to k=1000, regardless of vehicle capacity (S=10, S=100 and S=199). Conclusion is, that regardless of vehicle capacity, increase of number of iterations from 500 to 1000 in stopping rule will not lead to significant increase of the quality of the solution.

In every variant of the problem (S=10, S=100 and S=199), difference in average path length between k=200 and k=1000 is insignificant (18-30%), while time consumption varies significantly. Solution time of the problem with k=1000 is about four times higher, than with k=200. This means, that it's not practical to use high number of iterations in stopping rule.

Similar graph for examples gil262 and ch130 is shown in Picture 3.



Picture 3 – Relation of average path length and number of iterations in stopping rule for gil262 and ch130

High number of iterations (k=500 and k=1000) does not lead to significant reduction of average path length, as shown in the figure. Considering, that doubling number of iterations in stopping rule leads to doubling of time consumption (on average), most reasonable approach would be using stopping rule with number of iterations equal to number of pairs of nodes (k=n).

4. Conclusion

New enumeration algorithm 4-opt-o is proposed.

Two series of computational experiments were conducted to evaluate quality of 4-opt-o procedure compared to exact solutions, in particular and general case. For particular case, usage of 4-opt-o procedure is impractical.

In general case, 4-opt-o is significantly faster, than exact solutions, while quality of the solution is comparable.

Series of computational experiments were conducted to evaluate behaviour of 4-opt-o procedure with different stopping rules. Most acceptable rule was the one, where the number of iterations was equal to number of pairs of nodes in the problem.

References

1. Danzig G., Ramser J. The Truck Dispatching Problem // Management Science, 1959, Vol. 6, Issue 1, P. 80–91.
2. Vehicle Routing Problem, NEO Research Group. Available at: http://neo.lcc.uma.es/vrp/ (accessed February 13th, 2018).
3. Parragh S., Doerner K., Hartl R. A survey on pickup and delivery problems. Part II: Transportations between customers and depot // Journal fur Betriebswirtschaft, 2008, Issue 58, P.21-51.
4. Ruland K.S., Rodin E.Y. The pickup and delivery problem: Faces and branch-and-cut algorithm // Computers and Mathematics with Applications, Volume 33, Issue 12, June 1997, P. 1-13.
5. Renaud J., Boctor F.F., Ouenniche J. A heuristic for the pickup and delivery traveling salesman problem // Computers and Operations Research, V. 27, 2000, Issue 9, P. 905-916
6. Renaud J., Boctor F.F., Laporte G. Perturbation heuristics for the pickup and delivery traveling salesman problem // Computers and Operations Research, V. 29, 2002, Issue 9, P. 1129-1141
7. Dumas Y., Desrosiers J., Soumis F. The pickup and delivery problem with time windows. European Journal of Operation Research, V.54, Issue 1, 1991, P. 7–22
8. Furtadoa M., Munaria P., Morabito R. Pickup and delivery problem with time windows: a new compact two-index formulation // Technical Report. Production Engineering Department, Federal University of São Carlos, Rod. Washington Luís, km 235 – SP-310, São Carlos – SP – CEP: 13565-905 – Brazil, July, 2015. www.optimization-online.org/DB_HTML/2015/07/5022.html (accessed February 13th, 2018)
9. Hernández-Pérez H., Salazar-González J.J. A branch-and cut algorithm for the traveling salesman problem with pickup and delivery // Discrete Applied Mathematics, Volume 145, Issue 1, 2004, P. 126-139
10. Cordeau J.-F., Laporte G. The dial-a-ride problem: models and algorithms // Annals of Operations Research, Volume 153, Number 1, 2007, P. 29—46
11. Bronshtein E.M., Gindullina E.V., Gindullin R.V. Formalizatsii zadach pogruzki i dostavki // Vestnik Yuzhno-Ural'skogo universiteta. Seriya: Matematika. Mekhanika. Fizika., Tom 9, Nomer 1, 2017, C. 13–21 (Russian)
12. Burkard R.E., Cela E., Pardalos P.M., Pitsoulis L.S. The Quadratic Assignment Problem // Handbook of Combinatorial Optimization: Vol. 1-3, Springer US, 1999, P. 1713–1809
13. Glover F. Improved linear integer programming formulations of nonlinear integer problems // Management Science, 1975, Vol. 22, P. 455–460
14. Lawler E.L. The quadratic assignment problem // Management Science. Vol. 9. 1963. P. 586-599
15. Croes G.A. A Method for Solving Traveling-Salesman Problems // Operations Research, 1958, Issue 6, P.791-812
16. Lin S. Computer solutions of the traveling salesman problem. Bell System Technical Journal. Volume 44, Issue 10, P.2245-2269
17. Hooijenga D. Perturbation heuristics for the pickup and delivery traveling salesman problem [Internet] // Econometrie. 2015. Available from: http://hdl.handle.net/2105/30373
18. MP-TESTDATA - The TSPLIB Symmetric Traveling Salesman Problem Instances. Available at: http://elib.zib.de/pub/mp-testdata/tsp/tsplib/tsp/ (accessed February 13th, 2018)