Drone Arc Routing Problems

Ángel Corberán

(James Campbell[1], Isaac Plana[2] and José M. Sanchis[3])

1. University of Missouri-St. Louis 2. Universitat de València
3. Universidad Politécnica de Valencia

Cádiz, January 30 - February 1, 2019

Outline

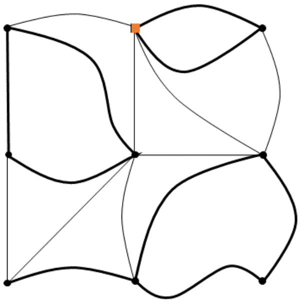
Outline

1. Arc Routing Problems (ARPs)
2. Drones and ARPs (1 vehicle)
   - Drone RPP
   - How to solve the Drone RPP?
   - Another type of drone
3. Drones and ARPs (several vehicles)
   - An example
4. Future work

ARPs

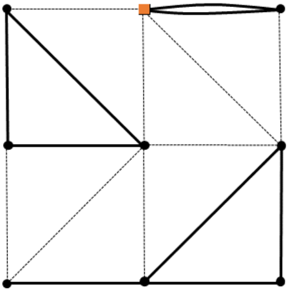
- Given a (street, road, electric line, etc.) network,
- given a set of lines that must be covered, each one with an associated cost,

$\Rightarrow$ find a tour that, leaving and coming back to the depot, traverses (servicing) all these lines with total minimum cost.



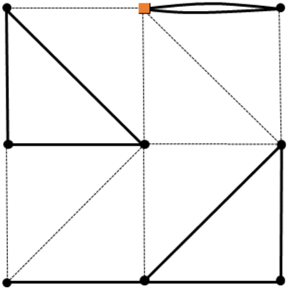
Postman ARPs

- Postman ARPs are usually solved in a graph, where each line is represented by an edge/arc with an associated cost.



The Rural Postman Problem (RPP)

- Orloff (1974)
- NP-hard
  Lenstra & Rinnooy Kan (1976)
  (easy transformation from the TSP)
- $G^R = (V, E_R)$ usually non-connected
- Its difficulty increase with the number of R-components
- Polynomially solvable if G^R is connected

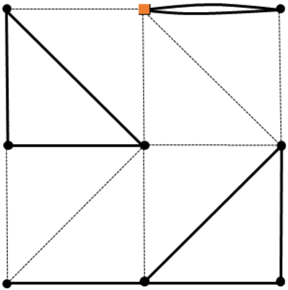


Postman ARPs

- In Postman ARPs, it is assumed that:

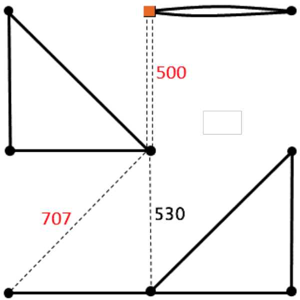
(1) Each edge has to be completely traversed (from one of its vertices to the other one).

(2) From the final vertex of one edge to the initial vertex of another edge, the postman can only go through the edges of the graph.



Postman ARPs

- Optimal Postman (RPP) solution
- z = “required cost” + 2237

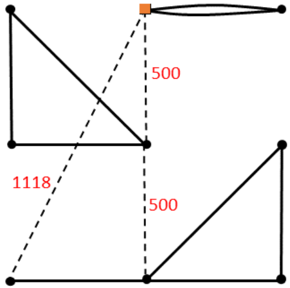


Outline

Drones and ARPs (1 vehicle)

- In Drones ARPs, it is assumed that:

(1) After servicing an edge, the drone can travel directly from any of its vertices to any endvertex of another edge without following the edges of the graph.



Optimal solution for the drone

z = "req. cost" + 2118

Drones and ARPs (1 vehicle)

- For this assumption 1 we have two theoretical results:

Drones and ARPs (1 vehicle)

Proposition

If graph $G_R = (V, E_R)$ is connected, then the cost of the optimal "Postman" route is at most twice the cost of the optimal "Drone" route, and this value is attainable.
(assuming that the cost of deadheading an edge is the same as the cost of traversing and servicing it).

Proposition

If graph $G_R = (V, E_R)$ is not connected, then the ratio between the cost of the optimal "Postman" route and the cost of the optimal "Drone" route can be as large as desired.

Drones and ARPs (1 vehicle)

- In Drone ARPs, it is assumed that:

(1) After servicing an edge, the drone can travel directly from any of its vertices to any endvertex of another edge without following the edges of the graph.

(2) Drones may start and end the service of an edge at any point of it.

Drones and ARPs (1 vehicle)

- In Drone ARPs, it is assumed that:

(1) After servicing an edge, the drone can travel directly from any of its vertices to any endvertex of another edge without following the edges of the graph.

(2) Drones may start and end the service of an edge at any point of it.

Assumption 2 implies that the shape of the lines must now be taken into account. (This is another difference with respect to postman ARPs, where the geometric shape of the edge is irrelevant, as long as its cost is given).

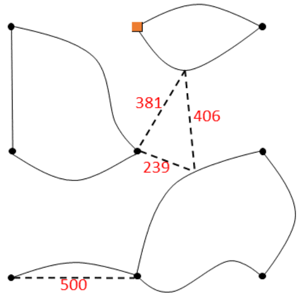
Drones and ARPs (1 vehicle)

- In Drone ARPs, it is assumed that:

(1) After servicing an edge, the drone can travel directly from any of its vertices to any endvertex of another edge without following the edges of the graph.

(2) Drones may start and end the service of an edge at any point of it.

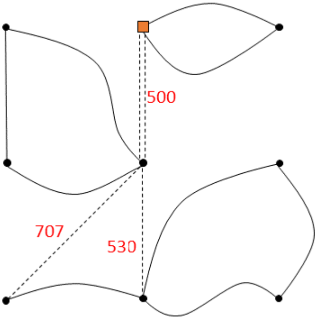
Assumption 2 implies that the shape of the lines must now be taken into account. (This is another difference with respect to postman ARPs, where the geometric shape of the edge is irrelevant, as long as its cost is given).

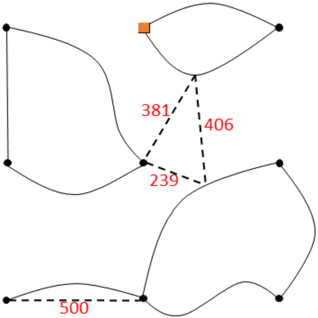


Optimal "Drone" solution
z = "Req. cost" + 1526

Drones and ARPs (1 vehicle)

"Postman" solution	"Req. cost" + 2237
"Drone" solution with assumption 1	"Req. cost" + 2118
"Drone" solution with assumptions 1+2	"Req. cost" + 1526



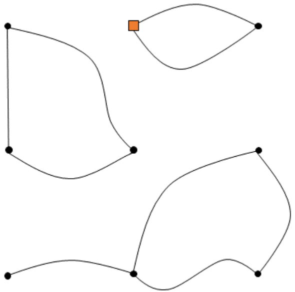


Drone RPP

We can now define the Drone RPP as follows:

- given a set of lines, each one with an associated service cost,
- a point called the depot, and
- assuming that the cost of deadheading between any two points is the Euclidean distance,

$\Rightarrow$ find the minimum cost tour starting and ending at the depot that services all the given lines.



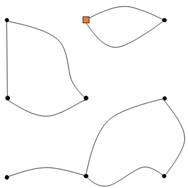
Drone RPP

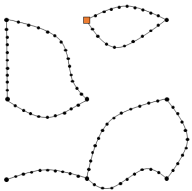
The drawback of the above definition is the difficulty of modeling a curved line.

In a real-world network, a curved line joining two points v, w can be approximated by a polygonal chain, formed by point v, an ordered set of intermediate points, and point w.

The better defined we want the curve to be, the greater the number of intermediate points we will need.

Each segment in a polygonal chain has associated a service cost, so that the sum of all the costs associated to all the segments of the polygonal chain equals the service cost of the original curve.

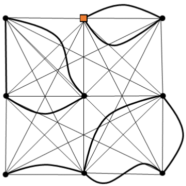




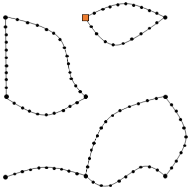
Solution of the Drone RPP

The Drone RPP can be solved as an undirected RPP on the following graph:

- Vertices: all the endpoints and all the intermediate points.
- Required edges: the segments in all the polygonal chains.
- The nonrequired edges form a complete graph with costs equal to the Euclidean distances.



$|V| = 9, \ |E_R| = 9, \ |E_{NR}| = 36$



$|V| = 98, \ |E_R| = 98, \ |E_{NR}| = 4753$

Solution of the Drone RPP

Hence, a Drone RPP instance can be extremely large.
An instance with 84 polygonal chains with 20 intermediate points per chain on average, has 1680 vertices, 1681 required edges and may have up to 1,410,360 nonrequired edges. However,

- the Drone RPP instance has the same number of R-connected components as the initial problem (not a very large number)
- The (potentially huge) number of nonrequired edges can be reduced (Christofides et al., 1981, Garfinkel and Webb, 1999).
- We can generate smaller RPP instances by considering only a subset of intermediate points from each polygonal chain. These RPP instances, called RPP(k), are easier to solve and provide upper bounds that can help to solve the Drone RPP instance. Moreover, their solutions are also feasible solutions for the Drone RPP instance.

Solution of the Drone RPP

The global algorithm

- Generate and solve RPP(0). Let z_0 be its optimal cost or the cost of the best feasible solution found.
- Generate RPP(1) and solve it with z_0 as an upper bound. Let z_1 be its optimal cost or the cost of the best feasible solution found.
- Generate RPP(3) and solve it with $\min\{z_0, z_1\}$ as an upper bound. Let z_3 be its optimal cost or the cost of the best feasible solution found.
- Solve the Drone RPP with $\min\{z_0, z_1, z_3\}$ as an upper bound. If solved, the solution is optimal.
- The best solution obtained (if any) is either the optimal solution, if obtained in (4), or a feasible solution (maybe optimal) otherwise.

The RPP(k) instances are solved using the branch-and-cut algorithm proposed in C., Plana & Sanchis (2007) for a generalization of the RPP. It is capable of solving to optimality RPP instances with up to 1000 vertices, 4000 edges, and 500 R-connected components.

Solution of the Drone RPP

Table: Characteristics of the “difficult” Drone RPP instances

Instancia	vért orig	líneas orig	vért total	segmentos	NR aristas potencial	NR aristas final	R comp
DronRPP56	22	24	477	479	113526	63445	4
DronRPP58	34	29	585	580	170820	61924	10
DronRPP510	44	47	939	942	440391	132866	9
DronRPP66	26	23	463	460	106953	63332	7
DronRPP68	34	32	639	637	203841	98230	9
DronRPP610	49	42	847	840	358281	112081	15
DronRPP77	39	46	912	919	415416	226185	6
DronRPP79	58	58	1134	1134	642411	207287	10
DronRPP710	58	54	1087	1083	590241	261592	11
DronRPP88	55	46	927	918	429201	152033	14
DronRPP89	61	55	1100	1094	604450	223080	14
DronRPP810	69	67	1343	1341	901153	335794	12
DronRPP99	64	61	1217	1241	739936	292996	10
DronRPP910	81	86	1715	1720	1469755	471435	12
DronRPP1010	83	92	1832	1841	1677196	674725	11

Solution of the Drone RPP

Table: Results on the “difficult” Drone RPP instances (* means an UB)

	RPP(0)		RPP(1)		RPP(3)		Drone RPP	
Instan.	opt.	time	imp.(%)	time	imp.(%)	time	imp.(%)	time
D56	5620.9	0.1	2.51	0.1	3.36	0.2	3.37	4.5
D58	6637.9	0.2	0.48	0.3	0.49	1.0	0.53	730.9
D510	9639.3	0.2	0.17	0.4	0.29	1.1	0.32	73.3
D66	4871.3	0.1	0.01	0.3	0.17	1.2	0.21	661.7
D68	7025.0	0.6	1.49	0.9	1.63	3.5	1.78	2164
D610	9544.9	0.3	0.00	1.1	0.06	9.2	*1.31	3600
D77	9700.9	0.4	0.32	1.6	0.35	1.9	*0.45	3600
D79	12088.1	0.5	0.07	0.5	0.07	2.1	0.09	1907
D710	11516.5	0.3	0.76	0.7	0.81	2.9	0.87	929.5
D88	10162.0	0.2	0.83	0.6	0.90	2.5	*1.36	3600
D89	11478.2	12.4	1.00	24.1	*1.43	3600	*2.10	3600
D810	13842.5	0.2	0.40	0.9	0.41	4.9	*1.20	3600
D99	13081.5	0.7	1.22	0.8	1.22	7.4	*1.75	3600
D910	17491.6	20.1	0.54	87.8	*0.61	3600	*1.91	3600
D1010	18336.6	0.6	0.05	5.8	0.13	336.2	*0.97	3600

Solution of the Drone RPP

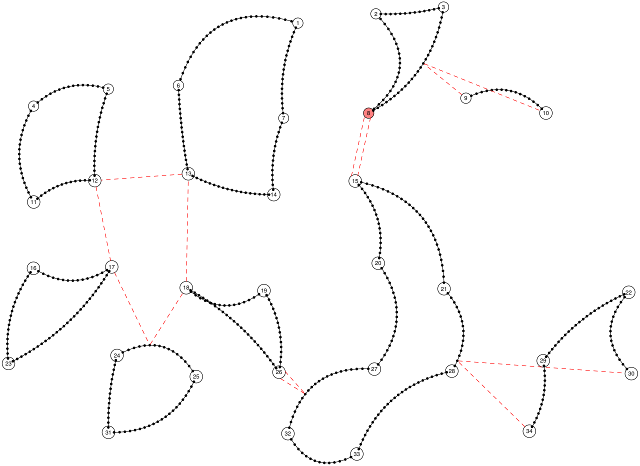


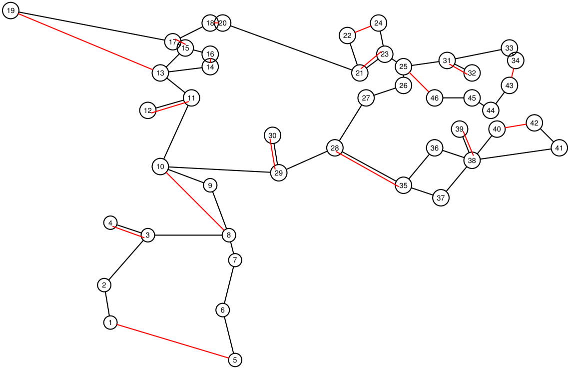
Figure: Optimal solution of DroneRPP68 instance

Solution of the Drone RPP

In many real utility networks (such as gas and electricity), there can be many tree-like structures creating:
more opportunities for drones to fly off the network, and providing greater benefits vs a vehicle that is restricted to the network.
A map of part of Spain‘s gas pipeline network:

Solution of the Drone RPP

The Drone RPP instance consists of a large component of 42 edges and 17 terminal nodes, and a small component of 3 edges. The optimal solution includes the drone flying 6 nonrequired edges within the large component, two nonrequired edges to connect the two components, and deadheading on 8 required edges.



The solution for a vehicle restricted only to the network would consist of duplicating most of the edges in the instance, which is 41% longer than the Drone RPP solution.

Another drone type

Cutting plotter

Cutting plotter

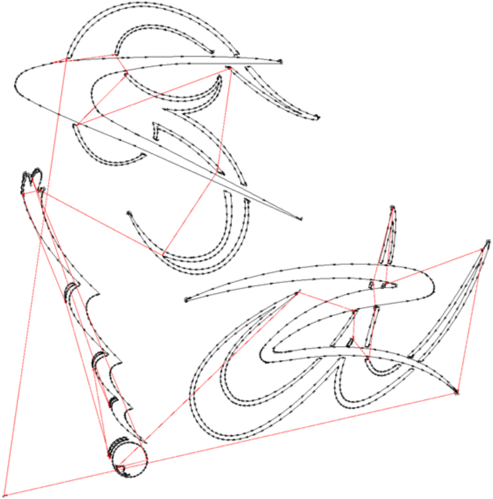
The drawing consists of some “vectors” that have to be cut with the plotter blade “down".

Black arrows: lines to service (cut)

Red arrows: nonrequired edges (movements with the blade up)

Up time: 82564.29

Down time: 204545.60

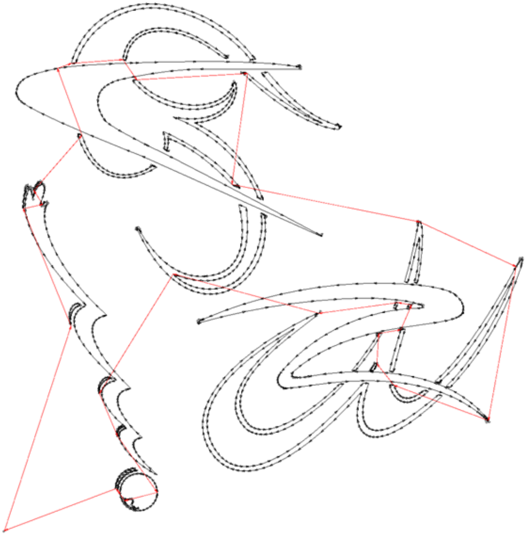


Another drone type

Cutting plotter

Up time: 48520.55

Down time: 204545.60



Outline

Drones and ARPs (several vehicles)

We make now some comments on the case in which the limited autonomy of the drones implies one vehicle alone cannot service all the required edges. Then, several drones may be needed (or several routes of the same drone).

L = Time limit or maximum length of a single route.

In this case, in addition to Assumptions 1 and 2, we can also assume that:

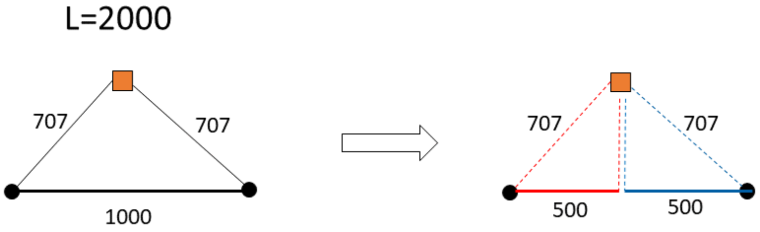
(3) The service of an edge can be shared by several vehicles.

Drones and ARPs (several vehicles)

We can assume that:

(3) The service of an edge can be shared by several vehicles.

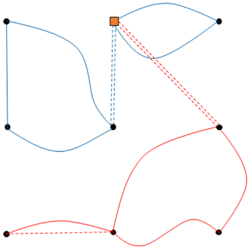
An example:



This assumption may be mandatory to obtain feasible solutions, as above.

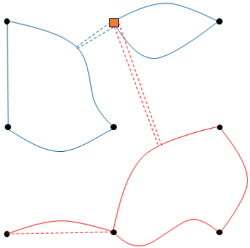
Drones and ARPs (several vehicles)

Although in some instances it may not be mandatory, this assumption may be convenient to obtain better feasible solutions:



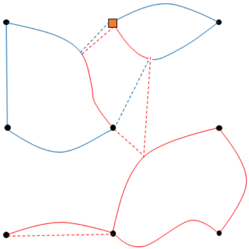
Solution with assumption (1) only

z = 3481 + 3201 = 6682



Solution with assumptions (1)+(2)

z = 3349 + 2743 = 6092



Solution with assumptions (1)+(2)+(3)

Z = 3610 + 2308 = 5918

With the Assumptions 1 to 3, we can define the problem for several vehicles as follows:

Definition (Length constrained k-drones rural postman problem, LCkDRPP)

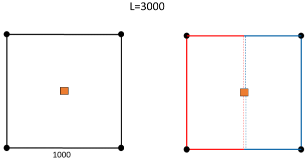
Given a set of lines, each one with an associated service cost, and a point called the depot, assuming that the cost of deadheading between any two points is the Euclidean distance, and given a constant L, find a set of tours starting and ending at the depot and with lengths no greater than L such that they jointly traverse all the given lines completely with minimum total cost.

Drones and ARPs (several vehicles)

In the LCkDRPP, the service of a given edge can be shared by two or more different vehicles.

This feature also appears in other known routing problems such as the Split Delivery Vehicle Routing Problem (SDVRP). In the SDVRP, the demand of a customer can be split and serviced by several vehicles. It is known that, if the costs satisfy the triangle inequality, there is an optimal solution to the SDVRP where no two routes have more than one customer in common

In the LCkDRPP two vehicles can share more than one customer (edge):



Drones and ARPs (several vehicles)

The LCkDRPP is much more difficult than the single vehicle version for several reasons.
In addition to having several vehicles, the length constraints are difficult to handle.
Furthermore, the number of nonrequired edges cannot be reduced as much as in the Drone RPP, because, for example, the nonrequired edges joining two vertices, not both with odd degree, of the same R-connected component cannot be removed (Garfinkel and Webb).

The LCkDRPP is an area of ongoing research. How could it be solved?

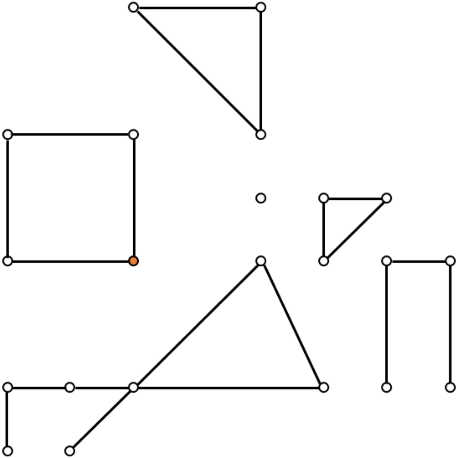
(1) Discretizing the instance.

(2) Solving a k-RPP with maximum length constraints.

Drones and ARPs (several vehicles)

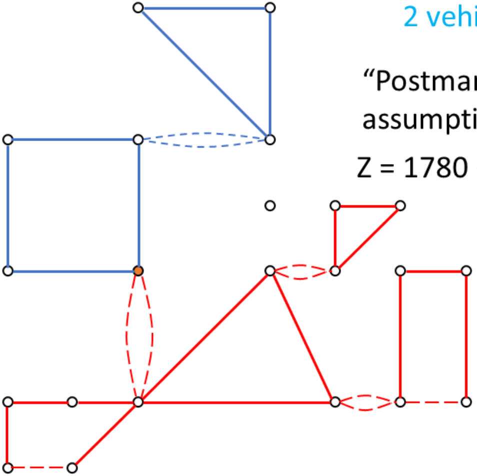
An example:

Drones and ARPs (several vehicles)



2 vehicles, L=1900

Drones and ARPs (several vehicles)

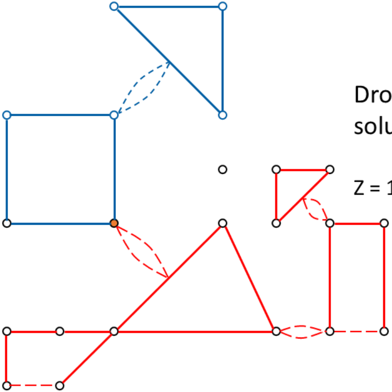


2 vehicles, L=1900

"Postman" & Drone ARP with assumption 1 solution:

Z = 1780 + 700 (940 + 1540)

Drones and ARPs (several vehicles)



2 vehicles, L=1900

Drone ARP with assumptions 1+2+3 solution:

Z = 1780 + 553.56 (881.42 + 1452.14)

Outline

Future work

- Look for some valid inequalities for the LCkDRPP.
- Design and implement some heuristics and a branch-and-cut algorithm.
- Study more general problems considering several depots for the drones and combining the location of the depots for the drones and their routes.

Drones and ARPs

Thank you for your attention !