An Exact Solution for the TSP with Drone Based on a Decomposition Approach

Sebastián Vásquez, Gustavo Angulo, Mathias Klapp
School of Engineering, Pontificia Universidad Católica de Chile,
savasquez@uc.cl, gangulo@ing.puc.cl, maklapp@ing.puc.cl

1. Motivation

The Traveling Salesperson Problem with Drone (TSP–D), introduced by Murray and Chu (2015) and further studied in Agatz, Bouman, and Schmidt (2016), Bouman, Agatz, and Schmidt (2018), Yurek and Ozmutlu (2018) is a vehicle routing model in which a given set of customers must be visited once, either by a truck route starting and ending at a depot or by a drone direct dispatch from the truck en route. Immediately after the drone visits a customer, it must rendezvous with the truck at one of the remaining customer nodes to be visited in the truck's route. If the truck reaches the meeting point earlier, then it gets delayed until the drone arrives. The objective of the TSP–D is to minimize the truck route total duration, which is defined as the sum of its travel and waiting times.

In theory, the TSP–D is NP–Hard since it generalizes the TSP. In practice, it has also proven to be a difficult problem to solve since it combines route sequencing and time synchronization decisions. In the literature, exact methods for the TSP–D have been proposed by Bouman, Agatz, and Schmidt (2018), Roberti and Ruthmair (2019), Tang, van Hoeve, and Shaw (2019) and Yurek and Ozmutlu (2018). As we are aware, only Roberti and Ruthmair (2019) reports results solving instances having more than 18 nodes via Branch–and–Price.

Also, available compact formulations for the TSP–D in the literature constraint the feasible set of solutions to heuristically solve this problem. For example, Van Dijck, Bouman, and Spliet (2018) limits the number of nodes a truck can visit without the drone, but, as empirically suggested by Bouman, Agatz, and Schmidt (2018), this restriction can increase route total duration up to 15%.

Others, use enumeration approaches, such as constraint programming, dynamic programming or full enumeration. Some of these approaches have solved bigger problems, but at the cost of more complex solution implementations.

We propose a compact MIP formulation for the TSP–D, which is decomposed in two stages and solved via a Benders–type decomposition algorithm. To our knowledge, no research effort has

studied decomposition approaches for the TSP–D, which have been proven successful in solving related combinatorial optimization problems; see Rahmaniani et al. (2017).

2. Model formulation

The TSP–D model is defined over a digraph $G=(N,A)$, where N is the set of customer locations, A is the set of directed arcs linking all locations, and c_a and d_a are truck and drone travel times for each $a \in A$, respectively; vectors c and d are assumed to be metric and proportional to each other, *i.e.*, $c=\alpha \cdot d$. We assume that truck is no faster than the drone, *i.e.*, $\alpha \geq 1$. As in the literature (see Bouman, Agatz, and Schmidt (2018)), we assume negligible service and drone charging times, a drone with unit capacity and unlimited flight autonomy.

Our decomposition approach goes as follows. In the first–stage model (1), the objective is to plan a minimum duration truck route, encoded in a binary vector $x \in \mathbb{X}$. Each variable $x_a \in \{0,1\}$ selects whether or not we choose arc a, and $\mathbb{X}$ is the set of all feasible truck routes covering a subset S of customer locations. The total duration in the objective function is split into travel time $c^\top x$ and truck delay $Q(x)$.

$$\min\{c^\top x + Q(x) :\ x \in \mathbb{X}\}. \tag{1}$$

The value of $Q(x)$ is obtained in the second–stage model (2), which assumes x to be fixed, and is computed by solving a MIP. Decision vector $o \in \mathbb{Y}(x)$ encodes a drone route, where $\mathbb{Y}(x)$ is the set of all feasible drone routes covering $N \setminus S$. Each variable $o_{ikj} \in \{0,1\}$ selects whether or not the drone executes an operation (i,k,j), which is defined as a direct dispatch of the drone taking off from the truck in node i, visiting customer k and landing back to the truck in node j. The truck delay generated by operation (i,k,j), defined as $w_{ikj}(x) \triangleq \max\{d_{ik}+d_{kj}-\hat{c}_{ij}(x),0\}$, depends on the truck's sub–route duration between i and j, *i.e.*, $\hat{c}_{ij}(x)$. If no feasible solution exists, then $Q(x)=+\infty$.

$$Q(x) = \min\{w(x)^\top o :\ o \in \mathbb{Y}(x)\}. \tag{2}$$

We relax model (1) by replacing $Q(x)$ with an underestimate $\theta \geq 0$. An optimal relaxed solution (x^*,θ^*) can be leveraged to obtain a feasible solution $(x^*,Q(x^*))$ and produce a valid upper bound $c^\top x^* + Q(x^*)$ to the optimal value of model (1).

Once solved, the relaxed model can also be refined via optimality cuts introduced by Laporte and Louveaux (1993). An optimality cut is defined as

$$\theta \geq -Q(\hat{x}) \left[\sum_{a \in A:\hat{x}_a=1} (1-x_a) + \sum_{a \in A:\hat{x}_a=0} (1-x_a) \right] + Q(\hat{x}) \tag{3}$$

and, when it is added to our relaxation, cuts–off solution $(\hat{x},\hat{\theta})$ if $\hat{\theta} < Q(\hat{x})$ maintaining the relaxation of (1) valid.

We also manage to strengthen the optimality cut to each neighbor of a feasible route $\hat{x}$ defined as any sub–route skipping, at most, t consecutive customer nodes in $\hat{x}$. To do this, we define the *t–shortcut*, $\mathcal{S}_{\hat{x}}(t)$, as the set of arcs $(i,j) \in A$ that skip, at most, t nodes in $\hat{x}$ such that i is visited before j.

We prove that all feasible solutions (x,θ) to model (1) satisfy

$$\theta \geq -Q(\hat{x}) \left[\sum_{a \in A: \hat{x}_a = 1} (1 - x_a) - \sum_{a \in \mathcal{S}_{\hat{x}}(t)} x_a - \sum_{i \in N: \hat{\gamma}_i = 1} (1 - \gamma_i) \right] + Q(\hat{x}), \tag{4}$$

where $\gamma_i \triangleq \sum_{a \in \delta^+(i)} x_a$ is an auxiliary variable indicating if customer i is visited by the truck. So, (4) defines a valid lifting of the optimality cut.

Finally, we manage to build another relaxation to the TSP–D by relaxing the fact that the truck and drone are forced to wait for each other at rendezvous points. We use this result to build functional lower bounds to θ and add them to the relaxed first–stage model directly.

3. Computational results

We design a set of 20 computational instances with up to 25 customer nodes, which are uniformly distributed over a square region. Our solution method is tested for relative drone speeds $\alpha \in \{1, 2, 4\}$. Also, we use a Multicommodity Flow formulation to represent the truck route $\mathbb{X}$; see Claus (1984).

Table 1 presents average results for each instance configuration. Column #Opt displays the number of instances optimally solved within 2 hours, Runtime the average CPU time over all instances optimally solved and Gap the average optimality gap over all unsolved instances. Empirically, we suggest that our decomposition approach has a comparable performance to the existing literature without restricting the number of nodes a truck may travel alone *a priori*.

Table 1 Computational average results for TSP–D

n	$\alpha = 1$			$\alpha = 2$			$\alpha = 4$		
	#Opt	Runtime	Gap (%)	#Opt	Runtime	Gap (%)	#Opt	Runtime	Gap (%)
15	20/20	00:02:39	–	20/20	00:01:25	–	20/20	00:00:08	–
16	20/20	00:04:07	–	20/20	00:02:33	–	20/20	00:00:21	–
17	20/20	00:07:42	–	20/20	00:05:42	–	20/20	00:00:31	–
18	20/20	00:26:47	–	20/20	00:06:41	–	20/20	00:01:14	–
19	20/20	00:31:34	–	20/20	00:14:33	–	20/20	00:01:29	–
20	14/20	00:47:39	1.9	19/20	00:15:42	3.73	20/20	00:02:35	–
21	7/20	00:46:12	3.75	17/20	00:33:46	2.37	20/20	00:06:50	–
22	4/20	01:11:03	5.30	12/20	00:49:31	4.31	19/20	00:07:00	3.87
23	2/20	00:39:12	5.13	5/20	00:54:25	4.11	18/20	00:10:37	3.81
24	1/20	00:50:03	6.77	1/20	00:31:38	5.51	17/20	00:29:17	12.63
25	0/20	–	7.13	0/20	–	6.69	15/20	00:36:41	5.69

3.1. Extended experiments

We also address several TSP–D extensions such as allowing the drone to execute *loop* operations, *i.e.*, (i,k,i). This setting is also studied in Bouman, Agatz, and Schmidt (2018) solving up to 16 node problems.

Table 2 presents computational results for this setting. In this case, we do not set a time limit and $\alpha = 2$. Column Runtime displays the average CPU time, Explored nodes the average number of B&B tree nodes explored, and Cuts the average number of cuts added. Empirically, we conclude that our solution method is capable to solve larger problems than the existing literature.

Table 2 Computational average results for TSP–D with loops allowed

n	Runtime	Explored nodes	Cuts
15	00:03:17 (00:03:43)	56088 (45808)	2070
16	00:07:10 (00:09:22)	104435 (105457)	3654
17	00:31:13 (01:18:43)	294457 (564015)	7577
18	01:06:47 (03:18:03)	405034 (641947)	11933
19	01:56:53 (03:29:39)	626847 (722360)	18600

Obs: Standard deviation in parenthesis

References

Agatz N, Bouman P, Schmidt M, 2016 *Optimization approaches for the traveling salesman problem with drone. ERIM Report Series* .

Bouman P, Agatz N, Schmidt M, 2018 *Dynamic programming approaches for the traveling salesman problem with drone. Networks* 72(4):528–542.

Claus A, 1984 *A new formulation for the travelling salesman problem. SIAM Journal on Algebraic Discrete Methods* 5(1):21–25.

Laporte G, Louveaux FV, 1993 *The integer l-shaped method for stochastic integer programs with complete recourse. Operations research letters* 13(3):133–142.

Murray CC, Chu AG, 2015 *The flying sidekick traveling salesman problem: Optimization of drone-assisted parcel delivery. Transportation Research Part C: Emerging Technologies* 54:86–109.

Rahmaniani R, Crainic TG, Gendreau M, Rei W, 2017 *The benders decomposition algorithm: A literature review. European Journal of Operational Research* 259(3):801–817.

Roberti R, Ruthmair M, 2019 *Exact methods for the traveling salesman problem with drone. Optimization Online* .

Tang Z, van Hoeve WJ, Shaw P, 2019 *A study on the traveling salesman problem with a drone. International Conference on Integration of Constraint Programming, Artificial Intelligence, and Operations Research*, 557–564 (Springer).

Van Dijck E, Bouman P, Spliet R, 2018 *A Branch-and-Cut Algorithm for the Traveling Salesman Problem with Drone*. Master's thesis, Erasmus University of Rotterdam.

Yurek EE, Ozmutlu HC, 2018 *A decomposition-based iterative optimization algorithm for traveling salesman problem with drone. Transportation Research Part C: Emerging Technologies* 91:249–262.