DM872 assignment 2

sagra16
(Valdemar Grange - 081097-2033)

June 2020

It should preemptively be noted that the `ipynb` notebook supplied along side with the code shows results and the way the code was developed. It should also be noted that in the assignments where the data-set was not mentioned, I generated the combinations myself. I in addition to taking a subset of the provided data, generated the combinations myself for assignments where techniques such as column generation needs "new columns" to add.

1 Task 1

The (capacitated vehicle routing problem) CVRP can be formulated as a set-partitioning problem as follows.

$$\begin{aligned} \text{minimize} \quad & \sum_{r\in\Omega} c_r\theta_r \\ \text{subject to} \quad & \sum_{r\in\Omega} a_{i,r}\theta_r = 1, \quad \forall i \in N \\ & \sum_{r\in\Omega} \theta_r \leq m \\ & \\ & \theta_r \in \{0,1\}, \qquad \forall r \in \Omega \end{aligned}$$

Compared to the tasked problem we need a couple of things to complete the model. We implicitly only allow routes that satisfy the capacity limit by only allowing such sets in the problem.

The "additional" aspect is the time windows, which we can model quite easily by the following constraint.

$$\theta_r\, a_{i,r}\, q_{i,r} \leq l_i, \qquad \forall i \in N, \forall r \in \Omega$$

We let $q_{i,r}$ denote the time it takes to reach customer i in route r, where we include accounting for waiting for each customer in the route to be ready (eg the time window beginning). This can simply be implemented by observing `makeQ` from the source code. Observe that we can realize that this constraint can be applied before solving the model, meaning that effectively we can filter out solutions that do not satisfy this constraint. The optimal solution yields an objective value of 41.

2 Task 2

By introducing linear relaxation we can find a good, but not necessarily optimal solution. We perform linear relaxation by relaxing the binary constraint, such that our model becomes.

$$\begin{aligned} \text{minimize} \quad & \sum_{r\in\Omega} c_r\theta_r \\ \text{subject to} \quad & \sum_{r\in\Omega} a_{i,r}\theta_r = 1, \quad \forall i \in N \\ & \sum_{r\in\Omega} \theta_r \leq m \\ & \\ & \theta_r \geq 0, \qquad \forall r \in \Omega \end{aligned}$$

Running this on all combinations of the possible routes. The result from the above model yields $m = 3$ which is not a violation of the original problem, furthermore an objective value of 41, which is in fact the same as the IP model. Luckily we did reach an integer solution which must mean that our polytope's best value is also the integer solution! I tried to not preprocess the time-window constraint, which did not yield an IP solution, such that it needed rounding but luckily the LP solution was integer (because of unimodularity?). If we inspect the produced routes:

$$\begin{bmatrix} [0 & 1 & 0 & 1 & 0 & 0 & 0 & 0] \\ [1 & 0 & 1 & 0 & 0 & 0 & 0 & 1] \\ [0 & 0 & 0 & 0 & 1 & 1 & 1 & 0] \end{bmatrix}$$

Which in fact is exactly the same as the IP solution:

$$\begin{bmatrix} [0 & 1 & 0 & 1 & 0 & 0 & 0 & 0] \\ [1 & 0 & 1 & 0 & 0 & 0 & 0 & 1] \\ [0 & 0 & 0 & 0 & 1 & 1 & 1 & 0] \end{bmatrix}$$

I also ran this on the supplied data, which did not yield an integer solution.

θ_0	0.33
θ_1	0.66
θ_4	0.66
θ_5	0.33
θ_7	0.33
θ_9	0.33
θ_{10}	0.33

A Lagrangian relaxation might fit better in some more complicated case where LP relaxation would not yield a good enough solution, since lagrangian is both tighter and does not pose so many questions about solution correctness. I will relax the amount of present vehicles.

$$\begin{aligned} & f(\lambda) = \\ & \text{minimize} \quad \sum_{r\in\Omega} c_r\theta_r - \lambda\left(\sum_{r\in\Omega}\theta_r - m\right) \\ & \text{subject to} \quad \sum_{r\in\Omega} a_{i,r}\theta_r = 1, \qquad \forall i \in N \\ & \qquad \theta_r \in \{0,1\}, \qquad \forall r \in \Omega \end{aligned}$$

To solve the problem finding the best λ for the lagrangian relaxation, I will use the Held and Karp method.

$$\theta = \mu\frac{z_{LR}(\lambda^k) - z}{\sum_i \lambda_i^2}$$

The solution found from the gradient decent with 10 iterations and $\mu = 2$ was $\lambda = 0$, the code can be viewed for my solution. The actual solution reached an objective of 41 with the same three rows as with the IP and LP relaxed models. Finding the specific sequence of customers to visit is simply a shortest path problem on each vehicle route. Of course the shortest-path algorithm should include the time-window constraint as the cost might not be the same as the travel-time.

This solution is not really interesting and just proves again the point that the vehicle limit constraint doesn't matter that much, since both the LP relaxation picks it and the lagrangian relaxation of the vehicle constraint. We can also see this since the gradient decent picks $\lambda = 0$, which indicates that it doesn't need this term to find a good feasible solution. In fact when I remove the vehicle constraint altogether the solution is still the same.

Therefore I also present a model where the other constraint of all customers being active as the relaxed constraint.

$$\begin{aligned}
\text{minimize} \quad & \sum_{r\in\Omega} c_r\theta_r + \sum_{i\in N} \lambda\Big(\sum_{r\in\Omega} a_{i,r}\theta_r - 1\Big) \\
\text{subject to} \quad & \sum_{r\in\Omega} \theta_r \le m \\
& \theta_r \in \{0,1\}, \qquad\qquad \forall r \in \Omega
\end{aligned}$$

Clearly the solver has a much less constrained solution space to pick from. The heuristic solution yields -10560 as the objective value, and a $\lambda = 105$ after 10 iterations and $\mu = 2$. Clearly the solution is completely invalid, naturally this must be because the solver is now instructed in simply finding a solution that minimizes cost and minimizes the amount of active customers. From increasing the iterations it seems like it will just keep growing until infinity.

On either my generated dataset and the supplied it keeps growing the constraint to infinity.

3 Task 3

Like in task 2, the master problem is actually just the linear relaxation of the original problem.

$$\begin{aligned}
\text{minimize} \quad & \sum_{r\in\Omega} c_r\theta_r \\
\text{subject to} \quad & \sum_{r\in\Omega} a_{i,r}\theta_r = 1, \quad \forall i \in N \\
& \sum \theta_r \le m \\
& \theta_r \ge 0, \qquad\qquad \forall r \in \Omega
\end{aligned}$$

The restricted master problem is a subset of the master problem where we do not yet consider a majority of the available problem. We denote this arbitrary subset by $\hat{\Omega} \subset \Omega$.

$$\begin{aligned}
\text{minimize} \quad & \sum_{r \in \hat{\Omega}} c_r \theta_r \\
\text{subject to} \quad & \sum_{r \in \hat{\Omega}} a_{i,r} \theta_r = 1, \quad \forall i \in N \\
& \sum \theta_r \leq m \\
& \theta_r \geq 0, \qquad\qquad \forall r \in \hat{\Omega}
\end{aligned}$$

We also need to develop a pricing model for evaluating the sub-problem. The sub-problem should minimize the reduced cost, where the negative reduction means that the column we are inspecting will enter the basis. In fact if a non-negative reduced cost is found, the solution is optimal. We must construct the dual to our problem, to determine the reduced cost. The dual variables can be observed by inspecting the primal tableau. In the problem we have two constraints, such that we have two sets of dual variables. One of the constraints is a singleton (the $\leq m$ constraint), which I will denote λ_0. The second set of constraints is over $i \in N$, so I will denote them $\lambda_{1,r}$. Our reduced cost of a variable θ_r can be realized by $\hat{c}_r = c_r - \lambda_T A_r$. In fact, constraints will be a valid pathing and with the reduction in cost it becomes a shortest path problem. Every vertex must exactly consume one edge and produce one, or rather every vertex must be the sink of exactly one edge and the source of one. Additionally we use big-M to model capacity constraints, and include the time window constraint. Let A be the set of edges, V be the set of vertices, Q the the max capacity and $x_{i,j}$ be the activation variable of path i, j. Finally let $y_i \in \mathbb{R}$ be the load on the vehicle when leaving vertex $i \in V$.

To express the "path taken" we must accumulate and check all time window constraints for violation.

$$\sum_{0,j \in A} x_{0,j} t_{0,j} + \sum_{j,i \in A} x_{j,i} t_{j,i} \cdots \sum_{s,z \in A} x_{s,z} t_{s,z} \leq l_z \sum_{s,z \in A} x_{s,z} \quad \forall z \in V$$

As one might notice, this is not easily expressible, as the amount of sum's should be equal to the length of the path from 0 to z.

1. One solution is to make the shortest-path algorithm aware of the domain of the routes, in a sense pre-generating all possible permutations of path's and excluding bad ones.

2. Another could be to check the constraint after the problem has been solved, if the constraint is violated then solve it again with the previously chosen path excluded.

3. A solution could also be to introduce an additional variable like y, but for time window's as this can also be viewed as a resource.

4. There also exists a polynomial time algorithm to solve this based on labelling in chapter two of [1], but seems way out of scope of this assignment.

I will use item 3, as it seems the most appropriate (but not simple, that would certainly be item 2). To do this I will introduce a new variable t_i, which will be the time spent travelling to position t_i. t_i should be bounded by $0 \leq t_i \leq l_i$, where l_i is the latest possible time to arrive. This method uses big-M, denoted by M, and is formulated such that the solver must assign the time-spent correctly to the variable. Additionally we still need to account for waiting before the customer "opens" in case that the vehicle is too early, which is $max(e_i, t_i)$. We can model this by.

$$max(e_j, time_{i,j} + t_i) + Mx_{i,j} \leq t_j + M \quad \forall i, j \in A : j \neq 0$$

Once again there is a problem, max is not linear thus we must find a way around this. Luckily we are minimizing and want to find the maximum, so we can simply introduce a variable to the model which specifies the time of delivery j given we are coming from i.

$$\alpha_{i,j} \geq e_j \wedge \alpha_{i,j} \geq time_{i,j} + t_i \quad \forall i, j \in A : j \neq 0$$

The solver will pick the smallest value to assign to α that is, the maximum of the two, without compromising the solution quality. Keep in mind that "serving time" can also easily be modelled here if wished by simply adding it to α. The time constraint can now be expressed more formally as.

$$\begin{array}{ll}
\alpha_{i,j} \geq e_j \wedge \alpha_{i,j} \geq time_{i,j} + t_i & \forall i, j \in A : j \neq 0 \\
\alpha_{i,j} + Mx_{i,j} \leq t_j + M & \forall i, j \in A : j \neq 0 \\
0 \leq t_i \leq l_i & \forall i \in V \\
 & \\
\alpha \in \mathbb{R} &
\end{array}$$

The problem is as follows.

$$\begin{array}{lll}
\text{minimize} & \sum\limits_{i,j \in A} (c_{i,j} - \lambda_{1,i}) x_{i,j} - \lambda_0 & \\
\text{subject to} & \sum\limits_{i,k \in A} x_{i,k} = \sum\limits_{k,j \in A} x_{k,j} & \forall k \in N \\
 & \sum\limits_{j \in V \setminus \{i\}} x_{i,j} = 1 & \forall i \in V \\
 & & \\
 & \alpha_{i,j} \geq e_j \wedge \alpha_{i,j} \geq time_{i,j} + t_i & \forall i, j \in A : j \neq 0 \\
 & \alpha_{i,j} + Mx_{i,j} \leq t_j + M & \forall i, j \in A : j \neq 0 \\
 & 0 \leq t_i \leq l_i & \forall i \in V \setminus \{0\} \\
 & & \\
 & y_i + q_j + Mx_{i,j} \leq y_j + M & \forall i, j \in A : j \neq 0 \\
 & 0 \leq y_i \leq Q & \forall i \in N \\
 & & \\
 & x_{i,j} \in \{0, 1\} & \forall i, j \in A \\
 & \alpha_{i,j} \in \mathbb{R} & \forall i, j \in A
\end{array}$$

Notice that we cannot preemptively remove path's to be checked based on time windows, like before. This is because the shortest path algorithm works different in that the permutation of visiting order dictates when we arrive, thus we must include the time-window constraint.

4 Task 4

I will begin using a subset of original dataset and then add columns by looking at missing columns from the original dataset. In practice you would have the solution determine the column to add, but as this task is to be carried out by hand I will be not be creative and just use the initially proposed columns, and terminate once there are no more columns with reduced cost. Initially $\hat{\Omega} = \{\theta_3, \theta_4, \theta_{11}\}$.

$18\ \theta_3$	$+15\ \theta_4$	$+12\ \theta_{11}$	
	$+\theta_4$		$= 1$
θ_3			$= 1$
	θ_4		$= 1$
θ_3			$= 1$
θ_3			$= 1$
		θ_{11}	$= 1$
		θ_{11}	$= 1$
		θ_{11}	$= 1$
θ_3	$+\theta_4$	θ_{11}	≤ 3
θ_3	$,\theta_4$	θ_{11}	≥ 0

Then we solve and look for a possible column to introduce based on the price of the previously introduce problem. We have initially.

$$\begin{gathered}
\theta_3 = 1.00 \\
\theta_4 = 1.00 \\
\theta_{11} = 1.00 \\
cost = 45.00 \\
\lambda_{1,r} = (15.00, 18.00, 0.00, 0.00, 0.00, 12.00, 0.00, 0.00) \\
\lambda_0 = 0.0
\end{gathered}$$

We can introduce the variable θ_7, with the reduced cost $\hat{c} = 18 - 10 = -8$

λ \ c_r	18	15	12	
15.0	0	1	0	$= 1$
18.0	1	0	0	$= 1$
0.0	0	1	0	$= 1$
0.0	1	0	0	$= 1$
0.0	1	0	0	$= 1$
12.0	0	0	1	$= 1$
0.0	0	0	1	$= 1$
0.0	0	0	1	$= 1$
0.0	1	1	1	≤ 3
θ	1.0	1.0	1.0	

Figure 1: Table before adding the column

λ \ c_r	18	15	10	12	
15.0	0	1	0	0	$= 1$
0.0	1	0	1	0	$= 1$
0.0	0	1	0	0	$= 1$
0.0	1	0	1	0	$= 1$
18.0	1	0	0	0	$= 1$
12.0	0	0	0	1	$= 1$
0.0	0	0	0	1	$= 1$
0.0	0	0	0	1	$= 1$
0.0	1	1	1	1	≤ 3
θ	1.0	1.0	0.0	1.0	

Figure 2: Table after adding column

Figure 3: Notice that we use the **previous** table's λ's (dual variables, that represent the constraints) since the new column's reduced cost is relative to the **current** solution, the interesting cells have been marked in red

The remaining tables will be in subsection 7.1. Next I will introduce θ_1, with $\hat{c}$ in respect to the previous table in Figure 2.

$$\hat{c} = -17.0$$
$$cost = 45.00$$
$$\lambda_{1,r} = (15.00, 0.00, 0.00, 0.00, 18.00, 12.00, 0.00, 0.00)$$
$$\lambda_0 = 0.0$$

Next θ_2, like above:

$$\hat{c} = -5.0$$
$$cost = 45.00$$
$$\lambda_{1,r} = (15.00, 0.00, 0.00, 0.00, 18.00, 0.00, 18.00, -6.00)$$
$$\lambda_0 = 0.0$$

Next θ_9:

$$\hat{c} = -11.0$$
$$cost = 45.00$$
$$\lambda_{1,r} = (0.00, 0.00, 15.00, 0.00, 18.00, 0.00, 18.00, -6.00)$$
$$\lambda_0 = 0.0$$

Next θ_{10}:

$$\hat{c} = -16.0$$
$$cost = 45.00$$
$$\lambda_{1,r} = (0.00, 15.00, 20.00, 0.00, 8.00, 9.00, 8.00, 0.00)$$
$$\lambda_0 = -5.0$$

No more of the variables make sense to add since there are no more variables which yield a negative reduced cost (in the limited example). The final thetas.

$$\theta_1 = 0.00$$
$$\theta_2 = 0.00$$
$$\theta_3 = 1.00$$
$$\theta_4 = 1.00$$
$$\theta_7 = 0.00$$
$$\theta_9 = 0.00$$
$$\theta_{10} = 0.00$$
$$\theta_{11} = 1.00$$
$$cost = 45.00$$
$$\lambda_{1,r} = (4.00, 0.00, 11.00, 0.00, 18.00, 19.00, 18.00, -25.00)$$
$$\lambda_0 = 0.0$$

The duals can be read easier if Figure 4 is inspected, notice that the solution is but a subset of the entire dataset, also violations don't matter as they don't contribute to the optimal solution.

5 Task 5

I will first give a quick description of what the code does and then give a short example. First the `column_generation_RMP` function is invoked, which keeps iterating, since an internal return will stop the loop. In every iteration I solve the master problem, and find the duals. I then solve the pricing problem with these duals on every addable column until there are no more columns remaining, adding each negative objective value result to a list. After going through all these sub-problems, I then go through the generated sub-problems and pick the one with the most reduced cost. If no sub-problems with reduced cost was found OR there are no more columns to add, I return the restricted master solution, as it is the optimal. Keep in mind much of the code is simply accounting for things such as "off by 1" and loop base-cases.

Since the lagrangian solution was very unfriendly, I will use the result of the LP relaxed solution of the initially supplied dataset. The solutions from this were not integer, so I will randomly round and have the initial routes $1, 3, 4, 7$. Initially we have the subset $\hat{\Omega} \subset \Omega$.

r_r	$\hat{\Omega}_r$	c_r
r_1	[0 0 0 0 1 1 0 1]	12.0
r_3	[0 1 0 1 1 0 0 0]	18.0
r_4	[1 0 1 0 0 0 0 0]	15.0
r_7	[0 1 0 1 0 0 0 0]	10.0

And the solution values after the first iteration, the first r rows are λ_1's and the final ≤ 3 row is λ_0.

λ \ c_r	12	18	5	10	
15.0	0	0	1	0	$= 1$
17.0	0	1	0	1	$= 1$
0.0	0	0	1	0	$= 1$
0.0	0	1	0	1	$= 1$
0.0	1	1	0	0	$= 1$
0.0	1	0	0	0	$= 1$
13.0	0	0	0	0	$= 1$
0.0	1	0	0	0	$= 1$
0.0	1	1	1	1	≤ 3
θ	1	0	1	1	

First I introduce the route $[1, 1, 0, 0, 0, 0, 1, 0]$, which yields a reduction in cost of -17.0. Moreover the algorithm will keep solving the shortest path sub-problem and pick the best reduced cost to approach the point of primal and dual feasibility.

6 Task 6

References

[1] B. Petersen and D. Pisinger, *Shortest paths and vehicle routing.* PhD thesis, PhD thesis, 2011.

7 Appendix

7.1 CG Tables

λ \ c_r	18	15	12	
15.0	0	1	0	$= 1$
18.0	1	0	0	$= 1$
0.0	0	1	0	$= 1$
0.0	1	0	0	$= 1$
0.0	1	0	0	$= 1$
12.0	0	0	1	$= 1$
0.0	0	0	1	$= 1$
0.0	0	0	1	$= 1$
0.0	1	1	1	≤ 3
θ	1.0	1.0	1.0	

λ \ c_r	18	15	10	12	
15.0	0	1	0	0	$= 1$
0.0	1	0	1	0	$= 1$
0.0	0	1	0	0	$= 1$
0.0	1	0	1	0	$= 1$
18.0	1	0	0	0	$= 1$
12.0	0	0	0	1	$= 1$
0.0	0	0	0	1	$= 1$
0.0	0	0	0	1	$= 1$
0.0	1	1	1	1	≤ 3
θ	1.0	1.0	0.0	1.0	

λ \ c_r	12	18	15	10	12	
15.0	0	0	1	0	0	$= 1$
0.0	0	1	0	1	0	$= 1$
0.0	0	0	1	0	0	$= 1$
0.0	0	1	0	1	0	$= 1$
18.0	1	1	0	0	0	$= 1$
0.0	1	0	0	0	1	$= 1$
18.0	0	0	0	0	1	$= 1$
-6.0	1	0	0	0	1	$= 1$
0.0	1	1	1	1	1	≤ 3
θ	0.0	1.0	1.0	0.0	1.0	

λ \ c_r	12	22	18	15	10	12	
0.0	0	1	0	1	0	0	$= 1$
0.0	0	0	1	0	1	0	$= 1$
15.0	0	0	0	1	0	0	$= 1$
0.0	0	0	1	0	1	0	$= 1$
18.0	1	1	1	0	0	0	$= 1$
0.0	1	0	0	0	0	1	$= 1$
18.0	0	0	0	0	0	1	$= 1$
-6.0	1	0	0	0	0	1	$= 1$
0.0	1	1	1	1	1	1	≤ 3
θ	0.0	0.0	1.0	1.0	0.0	1.0	

λ \ c_r	12	22	18	15	10	11	12	
0.0	0	1	0	1	0	0	0	$= 1$
15.0	0	0	1	0	1	0	0	$= 1$
20.0	0	0	0	1	0	0	0	$= 1$
0.0	0	0	1	0	1	0	0	$= 1$
8.0	1	1	1	0	0	1	0	$= 1$
9.0	1	0	0	0	0	0	1	$= 1$
8.0	0	0	0	0	0	1	1	$= 1$
0.0	1	0	0	0	0	1	1	$= 1$
-5.0	1	1	1	1	1	1	1	≤ 3
θ	0.0	0.0	1.0	1.0	0.0	0.0	1.0	

λ \ c_r	12	22	18	15	10	11	13	12	
4.0	0	1	0	1	0	0	0	0	$= 1$
0.0	0	0	1	0	1	0	1	0	$= 1$
11.0	0	0	0	1	0	0	1	0	$= 1$
0.0	0	0	1	0	1	0	1	0	$= 1$
18.0	1	1	1	0	0	1	0	0	$= 1$
19.0	1	0	0	0	0	0	0	1	$= 1$
18.0	0	0	0	0	0	1	0	1	$= 1$
-25.0	1	0	0	0	0	1	0	1	$= 1$
0.0	1	1	1	1	1	1	1	1	≤ 3
θ	0.0	0.0	1.0	1.0	0.0	0.0	0.0	1.0	

	$+\lambda_{1,0}$	$\lambda_{1,1}$	$+\lambda_{1,2}$	$+\lambda_{1,3}$	$+\lambda_{1,4}$	$+\lambda_{1,5}$	$+\lambda_{1,6}$	$+\lambda_{1,7}$	$+\lambda_0$	
θ_0				0.00		19.00	18.00		0.0	≤ 15.0
θ_1					18.00	19.00		-25.00	0.0	≤ 12.0
θ_2	4.00				18.00				0.0	≤ 22.0
θ_3		0.00		0.00	18.00				0.0	≤ 18.0
θ_4	4.00		11.00						0.0	≤ 15.0
θ_5	4.00	0.00					18.00		0.0	≤ 22.0
θ_6		0.00	11.00					-25.00	0.0	≤ 18.0
θ_7		0.00		0.00					0.0	≤ 10.0
θ_8			11.00		18.00		18.00		0.0	≤ 15.0
θ_9					18.00		18.00	-25.00	0.0	≤ 11.0
θ_{10}		0.00	11.00	0.00					0.0	≤ 13.0
θ_{11}						19.00	18.00	-25.00	0.0	≤ 12.0
									0.0	≤ 0.0

Figure 4