Courier Dispatch Problem

EC.1. The Network Flow Formulation for the CDP.

The *Courier Dispatch Problem* (CDP) is a static optimization problem that attempts to find a new delivery plan to accommodate the arriving order o at decision epoch a_o. The state of the system at the new order arrival, $S_{a_o} = (\mathcal{O}_{a_o}, P_{a_o}, \Theta_{a_o})$, consists of O_{a_o}, the information of the active orders accepted in previous epochs but are not served by a_o yet, P_{a_o}, the list of products that need to be collected for the active orders, and Θ_{a_o}, the current delivery plan for couriers. Since the CDP is defined the same for every decision epoch, without loss of generality we omit subscript a_o from all parameters and variables from here on.

The CDP is presented on an activity graph $G = (\mathcal{V}, \mathcal{A})$, where $\mathcal{V} = \mathcal{V}^M \cup \mathcal{V}^O \cup \xi$ is the set of vertices representing the storage- and customer-locations, and an additional fictitious terminal vertex and $\mathcal{A}$ defines the arc set. Since couriers are not rerouted when traveling between vertices, $v_k \in \mathcal{V}^{\mathcal{K}} \subseteq \mathcal{V}$ refers to the location of the courier in the current delivery plan, from where courier k's forward trip can be altered. That is, courier k's immediate activities at the current epoch are frozen against the possible updates before she reaches her corresponding courier node v_k. Finally, the fictitious terminal vertex ensures that every courier trip has a given ending location.

Let $\mathcal{O}$ be the set of not yet delivered orders, P be the uncollected products of orders in $\mathcal{O}$ and $P_{\hat{o}}$ be the uncollected products of order $\hat{o} \in \mathcal{O}$. For each uncollected product $p \in P_{\hat{o}}$ of an order $\hat{o}$, we define a unique set of vertices, denoted by $\mathcal{V}^p$ that represents the storage locations in which product $p \in P_{\hat{o}}$ is available and can be picked. Consequently, we define a set $\mathcal{V}^P = \bigcup_{\hat{o}\in\mathcal{O}} \bigcup_{p\in P_{\hat{o}}} \mathcal{V}^p$, representing all unique vertices associated with each possible location where a product can be collected for an order in $\mathcal{O}$. Furthermore, the set $\mathcal{V}^{\mathcal{O}}$ consists of vertices associated with the customer locations of each undelivered order $\hat{o} \in \mathcal{O}$. Let mapping $l : \mathcal{O} \mapsto \mathcal{V}^{\mathcal{O}}$ link an order $\hat{o}$ to its associated node and the reverse function $l^{-1} : \mathcal{V}^{\mathcal{O}} \mapsto \mathcal{O}$ links the customer location to the order. Furthermore, we denote $z^k_{v_k}$ as the earliest moment of time that courier k can depart from the vertex v_k.

Note that we account for the shopping time through arcs, and therefore $t(i, j)$ denotes shopping time if node i and/or j are an element of $\mathcal{V}^P$. For the case of a fixed visiting

duration per store and a constant pick-up duration per product, we can write $t(i,j)$ as follows:

$$t(i,j) := \begin{cases} w_{ij}, & \text{if } i \overset{\text{L}}{\neq} j, \\ \triangle f_m(0), & \text{if } j = v_m \wedge i \neq v_m, \\ \triangle f_m(n), & \text{if } j = v_m \wedge i = v_m, \end{cases} \tag{EC.1}$$

where $\triangle f_m(n) := f_m(n) - f_m(n-1), n \geq 1$, represent the incremental shopping time for additional task picking (see Section 3.2). Notice that $\triangle f_m(1)$ considers the fixed cost for visiting store m. Also, v_m denotes the node associated with store m.

Let x_{ij}^k be the binary variable if courier $k \in \mathcal{K}$ traverses on arc $(i,j) \in \mathcal{A}$. Let z_i^k be the departure time of courier k from vertex i and let $\hat{q}_i^k$ be the load size in the number of product of the courier k before departing at node i. Note that we changed our load and capacity definition to a more general case of number of products instead of number of orders. Auxiliary variables need to be introduced to model the number of orders the courier is currently working on. Then, the mixed integer programming formulation of the CDP is written as follows:

$$\min \quad \sum_{k\in\mathcal{K}} \sum_{i,j\in\mathcal{A}} t(i,j) x_{ij}^k \tag{EC.2}$$

$$\sum_{k\in\mathcal{K}} \sum_{i\in\mathcal{V}} \sum_{j\in\mathcal{V}^p} x_{ij}^k = 1 \qquad p \in P_{\hat{o}},\ \ \hat{o} \in \mathcal{O} \tag{EC.3}$$

$$\sum_{k\in\mathcal{K}} \sum_{i\in\mathcal{V}} x_{ij}^k = 1 \qquad j \in \mathcal{V}^{\mathcal{O}} \tag{EC.4}$$

$$\sum_{k\in\mathcal{K}} \sum_{j\in\mathcal{V}} x_{ij}^k = \sum_{k\in\mathcal{K}} \sum_{j\in\mathcal{V}} x_{ji}^k \qquad j \in \mathcal{V}^{\mathcal{O}} \cup \mathcal{V}^{\mathcal{P}} \tag{EC.5}$$

$$\sum_{i\in\mathcal{V}} \sum_{j\in\mathcal{V}^{p_j}} x_{ij}^k - \sum_{i\in\mathcal{V}} \sum_{l\in\mathcal{V}^{p_l}} x_{il}^k = 0 \qquad p_j, p_l \in P_{\hat{o}},\ \ \hat{o} \in \mathcal{O},\ \ k \in \mathcal{K} \tag{EC.6}$$

$$\sum_{i\in\mathcal{V}} \sum_{j\in\mathcal{V}^p} x_{ij}^k - \sum_{i\in\mathcal{V}} x_{i,l(\hat{o})}^k = 0 \qquad p \in P_{\hat{o}},\ \ \hat{o} \in \mathcal{O},\ \ k \in \mathcal{K} \tag{EC.7}$$

$$\sum_{j\in\mathcal{V}} x_{v_k j}^k = 1 \qquad k \in \mathcal{K} \tag{EC.8}$$

$$\sum_{i\in\mathcal{V}} x_{i\xi}^k = 1 \qquad k \in \mathcal{K} \tag{EC.9}$$

$$z_j^k \geq z_i^k + t(i,j) - M(1 - x_{ij}^k) \qquad (i,j) \in \mathcal{A},\ \ k \in \mathcal{K} \tag{EC.10}$$

$$z^k_{l(\hat{o})} \leq d_{\hat{o}} + M(1 - \sum_{k\in\mathcal{K}}\sum_{i\in\mathcal{V}} x^k_{i,l(\hat{o})}) \qquad \hat{o}\in\mathcal{O} \qquad \text{(EC.11)}$$

$$\hat{q}^k_j = \hat{q}^k_i + \begin{cases} x^k_{ij}, & \text{if } i\in\mathcal{V}^P \\ -|P_{l^{-1}(i)}|x^k_{ij} & \text{if } i\in\mathcal{V}^{\mathcal{O}} \end{cases} \qquad (i,j)\in\mathcal{A},\ k\in\mathcal{K} \qquad \text{(EC.12)}$$

$$\hat{q}^k_i \leq \hat{q}_k \qquad i\in\mathcal{V},\ k\in\mathcal{K} \qquad \text{(EC.13)}$$

$$x^k_{ij}\in\{0,1\}, q^k_i\in\mathbb{Z}_+, z^k_i\geq 0 \qquad (i,j)\in\mathcal{A},\ k\in\mathcal{K} \qquad \text{(EC.14)}$$

The objective function in (EC.2) minimizes the total travel and picking (and shopping) time of shoppers. The travel time on arc i,j is given in a functional form as it also includes non-linear shopping time at a store. We refer Arslan et al. (2021) on how to linearize this function.

Constraints (EC.3) ensures that each product should be collected only from one of the stores selling the product. Constraints (EC.4) guarantee that the courier drops off the products at each order's address. Constraints (EC.5) ensure the flow conversation. Constraints (EC.6) and (EC.7) ensure that the same courier collects all products for the same order and eventually drops off them at the delivery address of the order. Constraints (EC.8) and (EC.9) ensure that a trip for each courier starts at the current location and ends at the terminal node. Constraint set (EC.10) and (EC.11) guarantee to acknowledge travel times between activities and orders are delivered before deadlines. Note that M denotes a big number and $d_{\hat{o}}$ the delivery deadline of order $\hat{o}$. Recall that $d_{\hat{o}}$ is the delivery deadline of order $\hat{o}$. The last set of constraints (EC.12) and (EC.13) ensures the maximum number of products that a courier can serve simultaneously is not violated.

EC.2. Solution Approach to the CDP

The forward courier trips for the CDP associated with state S_{a_o} that is of an exponential size and enumerating all of them may not be practical. Therefore, we devise a generation method, which starts to solve a restricted master problem (RMP) with a restricted set of forward trips, denoted by $\Theta'_{a_o}\subset\Theta_{a_o}$. The RMP's solution is optimal if there is not a forward trip in $\Theta_{a_o}\backslash\Theta'_{a_o}$ with negative reduced cost. To identify whether a forward trip with negative reduced cost in $\Theta_{a_o}\backslash\Theta'_{a_o}$, we solve the *pricing problem*. We add those forward trips into the restricted forward trip set and resolve the RMP. This procedure is continued until no forward trip with negative reduced cost remains.

The associated pricing problem of the CDP is the resource constraint shortest path problem (RCSRP). Let μ_o be the dual variable associated with Constraints (3b) and κ_k be the dual variable associated with Constraints (3c) obtained by solving the linear relaxation of the RMP. Then, the reduced cost of the forward trip θ for courier k, denoted by π^c_θ is:

$$\tau^c_\theta = \tau_\theta - \sum_{o \in \mathcal{O}_{a_o}(o)} \mu_o - \kappa_k.$$

The pricing problem is to find the forward trip with a minimum τ_θ for each courier $k \in \mathcal{K}$.

We adopt a simple enumeration algorithm to identify/generate forward trips with negative reduced costs in a greedy fashion. That is, for each courier $k \in \mathcal{K}$, we initially identify the first activity of the ongoing trip and mark it as the starting vertex and remain the committed activities. The following path of the courier is extended by adding new activities based on the cheapest insertion heuristic and considering the preceding relationship of the pick-ups and drop-offs and delivery deadlines of the orders. After finding at most $|\mathcal{K}| \times |\mathcal{O}_{a_o}|$ forward trips, we solve the RMP again and continue adding columns until no forward trips with negative reduced cost can be found by the pricing problem. Solving both linear relaxation of the RMP and the RMP will provide a lower- and upper-bound for the CDP. Finally, the optimal solution can be obtained by running the pricing problem once again and adding any forward trip for which the reduced cost are smaller than or equal to the difference between the lower- and upperbound. Solving the RMP on the this set of columns will provide the optimal integer solution for the CDP.

EC.3. Performance Statistics for the Rolling Horizon Approach

In this section, we analyze the performance of the rolling horizon approach used to solve the CDP as described in Section 4.2. For only the PSS instances studied in Section 5.3.2, Table EC.1 shows the performance statistics for varying courier capacities and different order due times. Column $\Delta(\%)$ shows the average and standard deviation of the percentile difference between the upper and lower bounds for the CDP obtained after no more forward trips with negative reduced cost can be found by the pricing algorithm. We can conclude that our solution approach can find bounds that are almost equal to each other. As such the percentile difference between the upper bound and

final (integer) solution found, as reported in column Δ^f (%) is again small. Only when the capacity is small and the order due times are long, these gaps become larger as in this case there will be more non-committed orders that can still be swapped between different courier forward trips. The last two columns, # of columns and Time (sec.) present the average and standard deviation of the number of columns generated per decision epoch and the computational time for generating the columns and solving the model on an Intel Core i7 with 16 GB of RAM. From here we can see that minimal effort is required to resolve the CDP at each epoch. This is due that our approach can reuse the current forward trip from the previous epoch which allows us to quickly find the new optimal solution when a new order arrives.

Table EC.1 Performance statistics for the CDP for PSS instances.

Capacity (#)	Due mean (min.)	Δ (%)	Δ^f (%)	# of columns	Time (sec.)
1	60	0.01 (±0.02)	0.01 (±0.01)	2.08 (±0.29)	0.00 (±0.00)
2	60	0.19 (±0.12)	0.01 (±0.02)	2.52 (±0.55)	0.00 (±0.00)
3	60	0.15 (±0.03)	0.02 (±0.03)	2.43 (±0.27)	0.00 (±0.00)
4	60	0.11 (±0.07)	0.01 (±0.01)	2.75 (±0.67)	0.00 (±0.00)
1	90	0.12 (±0.04)	0.10 (±0.10)	11.83 (±1.56)	0.01 (±0.00)
2	90	0.20 (±0.11)	0.03 (±0.05)	4.82 (±0.52)	0.01 (±0.00)
3	90	0.21 (±0.07)	0.05 (±0.07)	5.12 (±0.89)	0.01 (±0.00)
4	90	0.25 (±0.13)	0.04 (±0.05)	5.41 (±0.95)	0.01 (±0.00)
1	120	0.26 (±0.11)	0.17 (±0.09)	28.79 (±3.93)	0.05 (±0.01)
2	120	0.22 (±0.09)	0.04 (±0.02)	6.76 (±1.27)	0.01 (±0.00)
3	120	0.25 (±0.11)	0.07 (±0.05)	6.68 (±1.71)	0.01 (±0.00)
4	120	0.23 (±0.10)	0.03 (±0.04)	6.78 (±1.25)	0.01 (±0.00)