Online Appendix
Dynamic Demand Management for Parcel Lockers

Daniela Sailer[1], Robert Klein[1], Claudius Steinhardt[2]
daniela.sailer@wiwi.uni-augsburg.de, robert.klein@wiwi.uni-augsburg.de,
claudius.steinhardt@unibw.de

[1] Chair of Analytics & Optimization, University of Augsburg
[2] Chair of Business Analytics & Management Science, University of the Bundeswehr Munich

A Notation

$\delta \in \mathcal{D} = \{1, \dots, D\}$	Compartment size (indexed in ascending order)
$d \in \mathcal{D}$	Parcel size
Q_δ	Number of compartments per size δ
$\tau = 1, \dots, \infty$	Day
$t \in \mathcal{T} = \{1, \dots, T\}$	Point in time
$t \in \mathcal{T}^+ = \mathcal{T} \cup \{T+1\}$	Point in time including allocation decision in $T+1$
$\mathfrak{F}^a$	Request arrival process
$c \in \mathcal{C} = \{1, \dots, C\}$	Customer type
$e \in \mathcal{E} = \{1, \dots, E\}$	Lead time
$\psi = (b, q)$	Pickup time
$b \in \mathcal{B} = \{1, \dots, B\}$	Day of pickup (measured in number of days after allocation)
B	Maximum storage time
$q \in \mathcal{T}$	Point in time of pickup
$\mathfrak{F}^p$	Pickup probability distribution
k	Decision epoch
$S_k = (\tau_k, t_k, L_k, O_k, r_k) \in \mathcal{S}$	Pre-decision state in decision epoch k
τ_k	Day in decision epoch k
t_k	Point in time in decision epoch k
$h \in \mathcal{H} = \{1, \dots, B-1\}$	Dwell time
$L_k = (l^k_{\delta ch})_{\delta \in \mathcal{D}, c \in \mathcal{C}, h \in \mathcal{H}}$	Locker occupancy in pre-decision state in decision epoch k
$f \in \mathcal{F} = \{1, \dots, F\}$	Remaining fulfillment time
$O_k = (o^k_{dcf})_{d \in \mathcal{D}, c \in \mathcal{C}, f \in \mathcal{F}}$	Pending orders in pre-decision state in decision epoch k
$r \in \mathcal{R} = \{1, \dots, C \cdot D \cdot E\}$	Request type with customer type c_r, parcel size d_r, lead time e_r
r_k	Request type in decision epoch k

S_0	Initial state
$X_k \in \mathcal{X}(S_k)$	Decision in decision epoch k
$g_k \in \mathcal{G}(S_k) \subseteq \{0,1\}$	Demand control decision in decision epoch k
$y_{\delta c f}$	Number of compartments of size δ with tentatively allocated parcel of customer type c in allocation decision f
$\tilde{O}_k$	Modified pending orders in decision epoch k
$A_k = (a^k_{d\delta c})_{d\in\mathcal{D},\delta\in\{d,\dots,D\},c\in\mathcal{C}} \in \mathcal{A}(S_k)$	Allocation decision in decision epoch k
$S^x_k = (\tau_k, t_k, L^x_k, O^x_k) \in \mathcal{S}^x$	Post-decision state in decision epoch k
L^x_k	Locker occupancy in post-decision state in decision epoch k
O^x_k	Pending orders in post-decision state in decision epoch k
$S^{Mx}(S_k, X_k)$	Transition function for transition from pre-decision state S_k with decision X_k to post-decision state S^x_k
$W_{k+1} = (\tau_{k+1}, t_{k+1}, r_{k+1}, P_{k+1}) \in \mathcal{W}(S^x_k)$	Exogenous information
$S^{MW}(S^x_k, W_{k+1})$	Transition function for transition from post-decision state S^x_k with exogenous information W_{k+1} to pre-decision state S_{k+1}
$\mathbb{P}(W_{k+1}\|S^x_k)$	Probability of exogenous information W_{k+1} in post-decision-state S^x_k
$P_{k+1} = (p^{k+1}_{\delta c h})_{\delta\in\mathcal{D},c\in\mathcal{C},h\in\mathcal{H}}$	Pickups during transition from post-decision state S^x_k to pre-decision state S_{k+1}
$R(S_k, X_k)$	Reward function for accepted requests
$\pi \in \Pi$	Policy
$X^\pi_k(S_k)$	Decision in pre-decision state S_k when following policy π
π^*	Optimal policy
$\bar{R}^*$	Reward rate of the optimal policy
$V(S)$	Value function in pre-decision state $S \in \mathcal{S}$
$V^x(S)$	Value function in post-decision state $S \in \mathcal{S}^x$
$\Delta V^x(S^x_k)$	Opportunity cost in decision epoch k
$Q^v = (Q^v_\delta)_{\delta\in\mathcal{D}}$	Virtual capacity
m	Failed delivery penalty
$M(S_k, X_k)$	Reward function for failed deliveries
$\bar{R}^*(Q^v)$	Reward rate of the optimal policy for virtual capacity Q^v
$\boldsymbol{\phi}(S^x_k)$	Feature vector for post-decision state S^x_k
$\boldsymbol{\theta}$	Vector of trained weights for value function approximation
λ	Length of capacity window
$w_{\delta\lambda}$	Number of capacity windows in tentative allocation plan in a compartment of size δ with length λ
$v_{\delta\lambda}$	Objective function coefficients for cost function approximation
$\hat{V}^x(S\|\boldsymbol{\theta})$	Approximated value function
$u = 1, \dots, U$	Scenarios for features
$\hat{\mathcal{F}} = \{1, \dots, F + B - 1\}$	Planning horizon for tentative allocation plans of features

$\hat{w}^u_{\delta\lambda}$	Number of capacity windows per compartment size δ and length λ in scenario u
$\hat{w}_{\delta\lambda}$	Capacity window features for value function approximation
$\tau^{\max}$	Number of simulated days for training
n	Simulated decision epoch during training
ε_n	Exploration scheme
$\mathbb{M}$	Replay memory
κ	Size of the replay memory
χ	Size of the samples for experience replay updates
η	Start of experience replay updates
ζ	Frequency of experience replay updates
S^x_0	Initial post-decision state
$(\boldsymbol{\phi}^0_n, \boldsymbol{\phi}^{\text{reject}}_n, \boldsymbol{\phi}^{\text{accept}}_n, R^{\text{accept}}_n)$	Experience in simulated decision epoch n
$\boldsymbol{\phi}^0_n$	Feature vector in simulated decision epoch $n-1$
$\boldsymbol{\phi}^{\text{reject}}_n$	Feature vector after rejecting in simulated decision epoch n
$\boldsymbol{\phi}^{\text{accept}}_n$	Feature vector after accepting in simulated decision epoch n
R^{accept}_n	Reward of accepting in simulated decision epoch n
$\bar{\mathbb{M}}$	Simple random sample drawn from $\mathbb{M}$
ν_i	Prediction target for experience replay
$\bar{R}$	Reward rate estimate
$\mathbb{E}[b]$	Expected value for day of pickup b
$\text{SD}[b]$	Standard deviation for day of pickup b
$\Omega_\delta(S_k)$	Aggregate representation of pre-decision state S_k
$V^\pi(S)$	Value function in state S when following policy π
$s_{\delta f}$	Number of empty compartments of size δ in allocation decision f
$w_{\delta\lambda f}$	Number of capacity windows in compartment size δ and with length λ that start with allocation decision f
β	Sampled value for day of pickup b
$\hat{o}^{ukx}_{\delta cf\beta}$	Pending orders with sampled day of pickup β in scenario u
$\hat{l}^{ukx}_{\delta ch\beta}$	Locker occupancy with sampled day of pickup β in scenario u
$\hat{s}^u_{\delta f}$	Number of unoccupied compartments of size δ in allocation decision f in scenario u
$\hat{w}^u_{\delta\lambda f}$	Number of capacity windows in compartment size δ and with length λ that start with allocation decision f in scenario u
$\hat{y}^u_{\delta cf\beta}$	Number of compartments of size δ with tentatively allocated parcel of customer type c in allocation decision f with sampled day of pickup β in scenario u
α^θ	Stepsize for temporal difference learning updates
$\alpha^{\bar{R}}, \alpha^{\bar{R}}_n$	Stepsizes for reward rate estimate
γ	Ridge regression regularization parameter
ρ	Auxiliary variable for reward rate estimate

Δ	Temporal difference error
$\bar{\boldsymbol{\phi}}_i^0$	Standardized features
$\boldsymbol{\theta}^{ER^*}$	Standardized weight vector
φ^{DLP}	Horizon of deterministic linear program
$\vartheta, \varphi \in \{1, \ldots, \varphi^{\mathrm{DLP}}\}$	Day in the deterministic linear program
$\hat{o}_{dc\varphi}$	Expected number of future requests with parcel size d, customer type c, and delivery date φ
$\bar{p}_{ch\varphi}$	Probability that a parcel of customer type c with dwell time h on day τ_k is still in the locker on day φ
$\hat{p}_{c\vartheta\varphi}$	Probability that a parcel of customer type c allocated on day ϑ is still in the locker on day φ
$\tilde{y}_{\delta c\varphi}$	Number of compartments of size δ with tentatively allocated parcel of customer type c on day φ in the deterministic linear program
$\mathcal{F}'$	Planning horizon of the full-information benchmark

B Proof for monotonicity of opportunity cost

This section is dedicated to the proof that a request's opportunity cost is nondecreasing with increasing parcel size. In preparation, we briefly restate the opportunity cost definition from Section 3.2.7:

$$\Delta V(S_k) = V(S^{Mx}(S_k, g_k = 0)) - V(S^{Mx}(S_k, g_k = 1)) \tag{16}$$

For the proof, we consider two arbitrary demand control pre-decision states $\dot{S}_k$ and $\bar{S}_k$ in a specific decision epoch k that only differ in the request's parcel size. All other components of the states are identical: $\dot{\tau}_k = \bar{\tau}_k, \dot{t}_k = \bar{t}_k, \dot{L}_k = \bar{L}_k, \dot{O}_k = \bar{O}_k$. For the sake of brevity, we denote the request's parcel size by $\dot{d}$ and $\bar{d}$ with $\dot{d} < \bar{d}$ for the respective states $\dot{S}_k$ and $\bar{S}_k$. The other two request attributes, customer type and lead time, do not differ between the two states and are represented by $\ddot{c}$ and $\ddot{f}$. Drawing on the definition given by (16), the monotonicity property translates to:

$$\Delta V(\bar{S}_k) \geq \Delta V(\dot{S}_k) \tag{17}$$

$$V(S^{Mx}(\bar{S}_k, g_k = 0)) - V(S^{Mx}(\bar{S}_k, g_k = 1)) \geq V(S^{Mx}(\dot{S}_k, g_k = 0)) - V(S^{Mx}(\dot{S}_k, g_k = 1)) \tag{18}$$

Note that the values in case of rejection ($g_k = 0$) are equal as we reach exactly the same post-decision state in both cases. Consequently, to prove (18) and thus (17), it suffices to show that:

$$V(S^{Mx}(\dot{S}_k, g_k = 1)) \geq V(S^{Mx}(\bar{S}_k, g_k = 1)) \tag{19}$$

To prove (19), we first observe that the post-decision states $\dot{S}_k^x$ and $\bar{S}_k^x$ which result from accepting the request in $\dot{S}_k$ and $\bar{S}_k$ respectively only differ with respect to the pending orders. In light of this, we can directly construct $\bar{S}_k^x$ from $\dot{S}_k^x$ by adjusting the orders according to $\bar{o}^{kx}_{\dot{d}\ddot{c}\ddot{f}} =$

$\dot{o}^{kx}_{\dot{d}\ddot{c}\ddot{f}} - 1$ and $\bar{o}^{kx}_{\bar{d}\ddot{c}\ddot{f}} = \dot{o}^{kx}_{\bar{d}\ddot{c}\ddot{f}} + 1$ with $\bar{o}^{kx}_{dcf} = \dot{o}^{kx}_{dcf}$ in all other cases. Note that the exogenous information W_{k+1} and its probability distribution are not affected. Consequently, the transition from the post-decision states $\dot{S}^x_k$ and $\bar{S}^x_k$ to the subsequent pre-decision states $\dot{S}_{k+1}$ and $\bar{S}_{k+1}$ is unaltered and, for each realization of exogenous information, the two pre-decision states again only differ with regard to the pending orders.

Importantly, this difference in pending orders affects the decision spaces $\mathcal{X}(\dot{S}_{k+1})$ and $\mathcal{X}(\bar{S}_{k+1})$ in decision epoch $k+1$. As a key result, every decision in $\mathcal{X}(\bar{S}_{k+1})$ is also a feasible decision for $\mathcal{X}(\dot{S}_{k+1})$, but not vice versa:

$$\mathcal{X}(\bar{S}_{k+1}) \subset \mathcal{X}(\dot{S}_{k+1}) \tag{20}$$

For both demand control and allocation, the impact of the different pending orders in the two pre-decision states is limited to constraints (3) and (4) of the demand control decision space:

- Comparing $\mathcal{X}(\dot{S}_{k+1})$ and $\mathcal{X}(\bar{S}_{k+1})$, constraint (3) only differs for $c = \ddot{c}$, $f = \ddot{f}$, $\delta \in \{\dot{d}, \dots, \bar{d}-1\}$. We can construct the adapted constraint for $\mathcal{X}(\bar{S}_{k+1})$ from the corresponding one for $\mathcal{X}(\dot{S}_{k+1})$:

$$\sum_{j=1}^{\delta} y_{j\ddot{c}\ddot{f}} \leq \sum_{j=1}^{\delta} \dot{o}^{k}_{j\ddot{c}\ddot{f}} - 1 \qquad\qquad \delta \in \{\dot{d}, \dots, \bar{d}-1\} \tag{21}$$

  For $\delta \in \{\bar{\delta}, \dots, D-1\}$, the reduction in $\bar{o}^{kx}_{\dot{d}\ddot{c}\ddot{f}}$ and the increase in $\bar{o}^{kx}_{\bar{d}\ddot{c}\ddot{f}}$ for $\bar{S}_{k+1}$ compared to $\dot{S}_{k+1}$ cancel each other out in the summation, yielding the exact same constraint as for $\mathcal{X}(\dot{S}_{k+1})$.

- Regarding (4), we observe that the total number of orders for $\dot{S}_{k+1}$ and $\bar{S}_{k+1}$ is identical.

As a result from these two observations, we conclude that the decision spaces only differ with regard to a reduced right-hand side in constraint (3) for $\mathcal{X}(\bar{S}_{k+1})$ as illustrated in (21). Consequently, every feasible decision out of $\mathcal{X}(\bar{S}_{k+1})$ is automatically also a feasible decision for the less restricted decision space $\mathcal{X}(\dot{S}_{k+1})$, but not vice versa.

Repeatedly applying this argument, we can feasibly replicate the sequence of decisions dictated by the optimal policy π^* starting from $\bar{S}^x_k$ and implement it from $\dot{S}^x_k$ onward, yielding a new policy $\bar{\pi}$. By following $\bar{\pi}$ from $\dot{S}^x_k$, we behave as if we were applying π^* starting from $\bar{S}^x_k$. The expected sequence of rewards is encapsulated in the value function and must be identical in both cases. Letting $V^{\pi}(S)$ denote the value if we apply policy π from S onward, we obtain $V^{\bar{\pi}}(S^{Mx}(\dot{S}_k, g_k = 1)) = V^{\pi^*}(S^{Mx}(\bar{S}_k, g_k = 1))$. By contrast, the value $V^{\pi^*}(S^{Mx}(\dot{S}_k, g_k = 1))$ resulting from applying the optimal policy π^* from $\dot{S}^x_k$ onward assumes that we select the optimal, value-maximizing decision in every subsequent decision epoch. As a general property, $V^{\pi^*}(S) \geq V^{\pi}(S)$ holds for all states and policies (Mahadevan 1996). Therefore, $V(S^{Mx}(\dot{S}_k, g_k = 1)) = V^{\pi^*}(S^{Mx}(\dot{S}_k, g_k = 1)) \geq V^{\bar{\pi}}(S^{Mx}(\dot{S}_k, g_k = 1)) = V(S^{Mx}(\bar{S}_k, g_k = 1))$, which proves (19) and thereby (17).

C Cost function approximation: Implementation details

In this section, we shed light on the details for the cost function approximation (CFA) by formalizing the CFA decision space, elaborating on the objectives, and demonstrating an efficient implementation of the lexicographic optimization.

C.1 CFA decision space

To model the CFA decision space, we require auxiliary decision variables $s_{\delta f}$ to denote the number of unoccupied compartments of size δ in allocation decision f and $w_{\delta\lambda f}$ as the number of capacity windows in a compartment of size δ and with length λ that start with allocation decision f. We formalize the CFA decision space as follows:

$$\sum_{c\in\mathcal{C}}\sum_{j=1}^{f} y_{\delta cj} + \sum_{c\in\mathcal{C}}\sum_{h=1}^{B-f} l^k_{\delta ch} + s_{\delta f} = Q_\delta \qquad \delta\in\mathcal{D},\ f=1,\dots,\min\{F,B-1\} \tag{22}$$

$$\sum_{c\in\mathcal{C}}\sum_{j=0}^{B-1} y_{\delta c,f-j} + s_{\delta f} = Q_\delta \qquad \delta\in\mathcal{D},\ B\le f\le F \tag{23}$$

$$\sum_{j=1}^{\delta} y_{jcf} \le \sum_{d=1}^{\delta} o^k_{dcf} \qquad \delta\in\mathcal{D}\setminus\{D\},\ c\in\mathcal{C},\ f\in\mathcal{F} \tag{24}$$

$$\sum_{\delta\in\mathcal{D}} y_{\delta cf} = \sum_{d\in\mathcal{D}} o^k_{dcf} \qquad c\in\mathcal{C},\ f\in\mathcal{F} \tag{25}$$

$$y_{\delta c1} = \sum_{d=1}^{\delta} a^k_{d\delta c} \qquad \delta\in\mathcal{D},\ c\in\mathcal{C} \tag{26}$$

$$o^k_{dc1} = \sum_{\delta=d}^{D} a^k_{d\delta c} \qquad d\in\mathcal{D},\ c\in\mathcal{C} \tag{27}$$

$$\sum_{j=1}^{f}\sum_{\lambda=f-j+1}^{F-j+1} w_{\delta\lambda j} = s_{\delta f} \qquad \delta\in\mathcal{D},\ f\in\mathcal{F} \tag{28}$$

$$\sum_{j=1}^{f-1} w_{\delta j,f-j} \le \sum_{c\in\mathcal{C}} y_{\delta cf} \qquad \delta\in\mathcal{D},\ f\in\{2,\dots,F\} \tag{29}$$

$$w_{\delta\lambda} = \sum_{f=1}^{F-\lambda+1} w_{\delta\lambda f} \qquad \delta\in\mathcal{D},\ \lambda\in\mathcal{F} \tag{30}$$

$$a^k_{d\delta c}, s_{\delta f},\ w_{\delta\lambda},\ w_{\delta\lambda f},\ y_{\delta cf}\in\mathbb{N}_0 \qquad \delta\in\mathcal{D},\ d\in\{1,\dots,\delta\},\ c\in\mathcal{C}, f\in\mathcal{F},\ \lambda\in\{1,\dots,F-f+1\} \tag{31}$$

Constraints (22)-(27) serve the same purpose as constraints (1)-(7) and model the basic allocation decision space. Constraints (28)-(30) determine the number of capacity windows. More specifically, constraints (28) ensure that the number of capacity windows that include f matches

the available capacity in f. Constraints (29) require a capacity window ending before F to be followed by an occupied compartment. Otherwise, a capacity window of length seven could, e.g., also be cast as seven capacity windows of length one. Constraints (30) aggregate $w_{\delta\lambda f}$ to $w_{\delta\lambda}$.

C.2 CFA objectives

We propose two objectives for the CFA:

1. We maximize the number of capacity windows in hierarchical order of the compartment size (the largest size having the highest priority):

$$\max \sum_{\delta\in\mathcal{D}} \sum_{\lambda\in\mathcal{F}} \delta\lambda w_{\delta\lambda} \tag{32}$$

   Basically, the term $\sum_{\lambda\in\mathcal{F}} \lambda w_{\delta\lambda}$ captures the available capacity per compartment size. The vacant capacity is then weighted by the compartment size index δ to ensure that we prioritize it accordingly.

2. To maximize the number of capacity windows in hierarchical order of the window length (with the longest one having the highest priority), we formulate the following objective:

$$\max \sum_{\delta\in\mathcal{D}} \sum_{\lambda\in\mathcal{F}} (2\lambda - 1) w_{\delta\lambda} \tag{33}$$

While the first objective is trivial, the second one requires more elaboration. In the following, we prove that by setting the coefficients in the objective to $2\lambda - 1$, we maximize the number of windows in lexicographic order of their length with $\lambda = F$ getting the highest priority. Essentially, the proof is based on showing that splitting up a single capacity window $\bar{W}$ in a compartment of size $\bar{\delta}$ and with length $\bar{\lambda} \geq 2$ into an arbitrary combination of capacity windows of shorter length leads to a deterioration of the objective value given by (33).

To formalize the notion of splitting up $\bar{W}$, we let $\bar{w}_{\delta\lambda}$ denote the number of capacity windows in a compartment of size $\delta \in \mathcal{D}$ and length $\lambda \in \mathcal{F}$ that $\bar{W}$ is divided into. A decomposition of $\bar{W}$ is permissible if it adheres to the following:

$$\sum_{\delta\in\mathcal{D}} \sum_{\lambda\in\mathcal{F}} \lambda \bar{w}_{\delta\lambda} = \bar{\lambda} \tag{34}$$

$$\sum_{\delta\in\mathcal{D}} \sum_{\lambda\in\mathcal{F}} \bar{w}_{\delta\lambda} \geq 2 \tag{35}$$

$$\bar{w}_{\delta\lambda} \in \mathbb{N}_0 \qquad \delta \in \mathcal{D},\ \lambda \in \mathcal{F} \tag{36}$$

Constraints (34) state that the sum of the length of the shorter windows must equal the length of the original window $\bar{W}$ given by $\bar{\lambda}$. Constraints (35) require that a decomposition must consist of at least two windows.

We now show that any decomposition fulfilling constraints (34)-(36) leads to a smaller contribution to the objective function than the original capacity window $\bar{W}$:

$$2\bar{\lambda} - 1 > \sum_{\delta \in \mathcal{D}} \sum_{\lambda \in \mathcal{F}} (2\lambda - 1)\bar{w}_{\delta\lambda} \tag{37}$$

$$(2 \sum_{\delta \in \mathcal{D}} \sum_{\lambda \in \mathcal{F}} \lambda \bar{w}_{\delta\lambda}) - 1 > \sum_{\delta \in \mathcal{D}} \sum_{\lambda \in \mathcal{F}} 2\lambda \bar{w}_{\delta\lambda} - \sum_{\delta \in \mathcal{D}} \sum_{\lambda \in \mathcal{F}} \bar{w}_{\delta\lambda} \tag{38}$$

$$\sum_{\delta \in \mathcal{D}} \sum_{\lambda \in \mathcal{F}} \bar{w}_{\delta\lambda} > 1 \tag{39}$$

Inequality (37) states our hypothesis. Plugging in constraint (34) and slightly rearranging the terms, we obtain (38) and (39). The latter is always true because of (35). This proves that (37) holds for all permissible decompositions.

Given that the total available capacity in the tentative allocation plan is fixed, (33) means that a longer window will always dominate an arbitrary combination of shorter windows. As a result, the objective prioritizes maximizing the number of windows with maximum length F as a primary objective, followed by maximizing the number of windows with length $F - 1$ as a secondary objective, and so forth, thereby inducing the desired allocation scheme.

C.3 Lexicographic optimization of (32) and (33) and CFA parameterizations

In this section, we show how the lexicographic optimization of (32) and (33) can be efficiently implemented by adequately weighting the two objective functions. The argumentation is based on upper bounds of the objective values of (32) and (33), which we state in a first step:

$$1 + \sum_{\delta \in \mathcal{D}} \delta F Q_\delta \tag{40}$$

$$1 + (2F - 1) \cdot \sum_{\delta \in \mathcal{D}} Q_\delta \tag{41}$$

In both cases, the highest attainable objective value arises if the tentative allocation plan contains no orders. In (40), this means that the total number of compartments given by $\sum_{\delta \in \mathcal{D}} Q_\delta$ is unoccupied over the entire horizon F. For (41), we follow a similar argument and additionally use the fact that longer capacity windows dominate shorter windows (Appendix C.2). To obtain bounds strictly larger than the highest attainable objective value, we add a constant of 1.

Given that the objective function coefficients in (32) and (33) as well as the decision variables are integer, the smallest possible change in the objectives is 1 unit. If we weight the secondary objective with the reciprocal of its upper bound, the contribution of the secondary objective in

the weighted objective function is guaranteed to be strictly less than 1. As a result, we obtain the following two objectives for maximizing (32) and (33) in lexicographic order:

$$\max \sum_{\delta\in\mathcal{D}}\sum_{\lambda\in\mathcal{F}} \delta\lambda w_{\delta\lambda} + \sum_{\delta\in\mathcal{D}}\sum_{\lambda\in\mathcal{F}} \frac{(2\lambda-1)}{1+(2F-1)\cdot\sum_{\delta\in\mathcal{D}} Q_\delta} w_{\delta\lambda} \tag{42}$$

$$\max \sum_{\delta\in\mathcal{D}}\sum_{\lambda\in\mathcal{F}} (2\lambda-1) w_{\delta\lambda} + \sum_{\delta\in\mathcal{D}}\sum_{\lambda\in\mathcal{F}} \frac{\delta\lambda}{1+F\cdot\sum_{\delta\in\mathcal{D}} \delta Q_\delta} w_{\delta\lambda} \tag{43}$$

Weighted objective (42) treats (32) as the primary and (33) as the secondary objective, (43) vice versa. They correspond to the following parameterizations of (14) respectively:

$$v_{\delta\lambda} = \delta\lambda + \frac{(2\lambda-1)}{1+(2F-1)\cdot\sum_{\delta\in\mathcal{D}} Q_\delta} \tag{44}$$

$$v_{\delta\lambda} = 2\lambda - 1 + \frac{\delta\lambda}{1+F\cdot\sum_{\delta\in\mathcal{D}} \delta Q_\delta} w_{\delta\lambda} \tag{45}$$

Note that depending on the instance size, numerical problems might arise for the secondary objective. In that case, one would have to resort to sequentially optimizing the individual objectives (32) and (33).

D Value function approximation: Implementation details

In this section, we elaborate on the implementation of the VFA. We provide details on the feature calculation, the training procedure, and the structure-enforcing constraints during experience replay.

D.1 Feature calculation

To generate scenarios of pickup behavior for the VFA features, it is sufficient to sample b, i.e., the number of days after allocation until the pickup occurs, because the tentative plan considers each day on an aggregate level, rendering the specific point in time of the pickup irrelevant. Letting β denote the sampled values for b, we transform the pending orders o^{kx}_{dcf} and orders in the locker $l^{kx}_{\delta ch}$ into $\hat{o}^{ukx}_{dcf\beta}$ and $\hat{l}^{ukx}_{\delta ch\beta}$.

The IP to calculate the VFA features is an extended version of the CFA decision space presented in Appendix C.1 to accommodate the sampled pickup behavior. To determine the number of capacity windows $\hat{w}^u_{\delta\lambda}$ with length λ and compartment size δ in the tentative allocation plan with sampled pickups in scenario u, we introduce auxiliary decision variables: $\hat{s}^u_{\delta f}$ denotes the number of unoccupied compartments of size δ in allocation decision f and scenario u, $\hat{w}^u_{\delta\lambda f}$ represents the number of capacity windows in a compartment of size δ and with length λ that start with allocation decision f in scenario u, and $\hat{y}^u_{\delta cf\beta}$ equals the number of compartments of size δ

to which we tentatively allocate a parcel belonging to a customer of type c in f with the pickup occurring β days after the allocation in scenario u.

$$\sum_{c\in\mathcal{C}}\sum_{j=1}^{f}\sum_{\beta=f-j+1}^{B}\hat{y}^u_{\delta cj\beta}+\sum_{c\in\mathcal{C}}\sum_{h=1}^{B-f}\sum_{\beta=f+h}^{B}\hat{l}^{ukx}_{\delta ch\beta}+\hat{s}^u_{\delta f}=Q_\delta \quad \delta\in\mathcal{D},\ f=1,\dots,\min\{F,B-1\} \tag{46}$$

$$\sum_{c\in\mathcal{C}}\sum_{j=0}^{B-1}\sum_{\beta=j+1}^{B}\hat{y}^u_{\delta c,f-j,\beta}+\hat{s}^u_{\delta f}=Q_\delta \quad \delta\in\mathcal{D},\ B\le f\le F \tag{47}$$

$$\sum_{c\in\mathcal{C}}\sum_{j=0}^{F+B-1-f}\sum_{\beta=f-F+j+1}^{B}\hat{y}^u_{\delta c,F-j,\beta}+\hat{s}^u_{\delta f}=Q_\delta \quad \delta\in\mathcal{D},\ F+1\le f\le F+B-1 \tag{48}$$

$$\sum_{j=1}^{\delta}\hat{y}^u_{jcf\beta}\le\sum_{d=1}^{\delta}\hat{o}^{ukx}_{dcf\beta} \quad \delta\in\mathcal{D}\setminus\{D\},\ c\in\mathcal{C},\ f\in\mathcal{F},\ \beta\in\mathcal{B} \tag{49}$$

$$\sum_{\delta\in\mathcal{D}}\hat{y}^u_{\delta cf\beta}=\sum_{d\in\mathcal{D}}\hat{o}^{ukx}_{dcf\beta} \quad c\in\mathcal{C},\ f\in\mathcal{F},\ \beta\in\mathcal{B} \tag{50}$$

$$\sum_{j=1}^{f}\sum_{\lambda=f-j+1}^{F+B-j}\hat{w}^u_{\delta\lambda j}=\hat{s}^u_{\delta f} \quad \delta\in\mathcal{D},\ f\in\hat{\mathcal{F}} \tag{51}$$

$$\sum_{j=1}^{f-1}\hat{w}^u_{\delta j,f-j}\le\sum_{c\in\mathcal{C}}\sum_{\beta=1}^{B}\hat{y}^u_{\delta cf\beta} \quad \delta\in\mathcal{D},\ f\in\{2,\dots,F\} \tag{52}$$

$$\hat{w}^u_{\delta\lambda}=\sum_{f=1}^{F+B-\lambda}\hat{w}^u_{\delta\lambda f} \quad \delta\in\mathcal{D},\ \lambda\in\hat{\mathcal{F}} \tag{53}$$

$$\sum_{\delta\in\mathcal{D}}\sum_{f=F+1}^{F+B-1}\sum_{j=1}^{f-1}\hat{w}^u_{\delta j,f-j}=0 \tag{54}$$

$$\hat{s}^u_{\delta f},\ \hat{w}^u_{\delta\lambda},\ \hat{w}^u_{\delta\lambda f},\ \hat{y}^u_{\delta cf\beta}\in\mathbb{N}_0 \quad \delta\in\mathcal{D},\ c\in\mathcal{C},\ f\in\hat{\mathcal{F}},\ \lambda\in\{1,\dots,F+B-f\},\ \beta\in\mathcal{B} \tag{55}$$

Similar to (22)-(25), constraints (46)-(50) ensure that we generate a feasible tentative allocation plan. Constraint (48) depicts the extended time horizon of the tentative plan due to the sampled pickup times. Mirroring (28)-(30), constraints (51)-(53) guarantee the correct computation of the capacity windows. We add constraint (54) to prohibit capacity windows terminating in $F,\dots,F+B-2$. In other words, capacity windows must either be followed by an occupied compartment or end with the final allocation decision in $f=F+B-1$.

Due to the longer planning horizon, we need to replace F in the denominators of (42) and (43) with $F+B-1$.

D.2 Training procedure

Algorithm 1 is an extended version of Algorithm 1 with additional hyperparameters: It includes stepsizes α^{θ} and $\alpha^{\bar{R}}$ together with $\alpha_n^{\bar{R}}$ for updating the parameter weights $\boldsymbol{\theta}$ and reward rate $\bar{R}$ respectively. For the ER updates, we use a regularization term γ and standardized features $\bar{\boldsymbol{\phi}}_i^0$ in the ridge regression. We obtain $\boldsymbol{\theta}^{ER*} = \arg\min_{\boldsymbol{\theta}^{ER}} \sum_{i\in\bar{\mathbb{M}}}[\nu_i - \boldsymbol{\theta}^{ER^T}\bar{\boldsymbol{\phi}}_i^0] + \gamma \sum_{\delta\in\mathcal{D}}\sum_{\lambda\in\hat{\mathcal{F}}}(\theta_{\delta\lambda}^{ER})^2$ and transform $\boldsymbol{\theta}^{ER*}$ into $\boldsymbol{\theta}$ by reversing the standardization.

Algorithm 1 Training of VFA parameter weights $\boldsymbol{\theta}$

Parameters: $\alpha^{\bar{R}}, \alpha^{\theta}, \gamma, \varepsilon_n, \zeta, \eta, \kappa, \tau^{\max}, \chi, v_{\delta\lambda}$ (for allocation decisions and computation of $\boldsymbol{\phi}(S_k^x)$)
Initialization: $\boldsymbol{\theta} = \mathbf{0}, \rho = 0, \mathbb{M} = \{\}, n = 0, \bar{R} = 0, S_0^x$

for $\tau = 1, \ldots, \tau^{\max}$ **do**
  repeat
    $n \leftarrow n + 1$
    sample exogenous information W_n to obtain $S_n = S^{MW}(S_{n-1}^x, W_n)$
    if $t_n \leq T$ **then**
      make ε-greedy demand control decision g_n to obtain $S_n^x = S^{Mx}(S_n, g_n)$
      compute TD error $\Delta = R(S_n, g_n) - \bar{R} + \hat{V}(S_n^x|\boldsymbol{\theta}_{n-1}) - \hat{V}(S_{n-1}^x|\boldsymbol{\theta}_{n-1})$
      update reward rate $\rho \leftarrow \rho + \alpha^{\bar{R}}(1-\rho),\ \alpha_n^{\bar{R}} \leftarrow \frac{\alpha^{\bar{R}}}{\rho},\ \bar{R} \leftarrow \bar{R} + \alpha_n^{\bar{R}}\Delta$
      perform TD update of parameter weights $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + \alpha^{\theta}\Delta\boldsymbol{\phi}(S_{n-1}^x)$
      update memory $\mathbb{M} \leftarrow \mathbb{M} \cup \{n\}$ and store experience $(\boldsymbol{\phi}_n^0, \boldsymbol{\phi}_n^{\text{reject}}, \boldsymbol{\phi}_n^{\text{accept}}, R_n^{\text{accept}})$
      if $|\mathbb{M}| > \kappa$ **then**
        remove oldest experience from memory $\mathbb{M}$
      end if
    end if
  until $t_n = T + 1$
  make allocation decision A_n to obtain $S_n^x = S^{Mx}(S_n, A_n)$
  if $(\tau_n > \eta) \wedge (\tau_n \bmod \zeta = 0)$ **then**
    uniformly draw sample $\bar{\mathbb{M}}$ from $\mathbb{M}$ with $|\bar{\mathbb{M}}| = \min\{\chi, |\mathbb{M}|\}$
    compute ER update targets $\nu_i \leftarrow \max\{R_i^{\text{accept}} - \bar{R} + \boldsymbol{\theta}^T\boldsymbol{\phi}_i^{\text{accept}}; -\bar{R} + \boldsymbol{\theta}^T\boldsymbol{\phi}_i^{\text{reject}}\}\ \forall i \in \bar{\mathbb{M}}$
    perform ER update on $\boldsymbol{\theta}$ via ridge regression using $\bar{\mathbb{M}}$ and γ
  end if
end for

D.3 Structure-enforcement during experience replay

To enforce structure in the value function, we impose the following constraints on the experience replay update:

$$\theta_{\delta\lambda} \geq \theta_{i\lambda} \qquad \delta \in \mathcal{D},\ i \in \{1, \ldots, \delta - 1\},\ \lambda \in \hat{\mathcal{F}} \tag{56}$$

$$\theta_{\delta\lambda} \geq \theta_{\delta j} \qquad \delta \in \mathcal{D},\ \lambda \in \hat{\mathcal{F}},\ j \in \{1, \ldots, \lambda - 1\} \tag{57}$$

$$\theta_{\delta\lambda} \geq 0 \qquad \delta \in \mathcal{D},\ \lambda \in \hat{\mathcal{F}} \tag{58}$$

Constraints (56) require the valuation of a capacity window of a given length λ to be nondecreasing with increasing compartment size δ. Similarly, constraints (57) ensure that the valuation of a capacity window with given compartment size δ is nondecreasing with increasing window length λ. Lastly, constraints (58) postulate nonnegativity.

E Policies and upper bound: Implementation details

In the following, we provide details for some of the policies and the full-information benchmark.

E.1 Bottom-up allocation

For the allocation according to BU, we combine the CFA decision space (Appendix C.1) with the following objective:

$$\max \sum_{\delta \in \mathcal{D}} \delta s_{\delta, f=1}$$

BU exclusively considers the next allocation decision ($f = 1$) and maximizes the available capacity in hierarchical order of the compartment size with the largest compartment size getting the highest priority.

E.2 Deterministic linear program

The DLP is a fundamental concept in the revenue management literature and, for a problem similar to the DPLDMP, also applied in practice by retailer Amazon (Sethuraman et al. 2024). Our DLP approach is based on certainty equivalent control (Bertsimas and Popescu 2003) and works as follows: To make a demand control decision in pre-decision state $S_k = (\tau_k, t_k, L_k, O_k, r_k)$, we solve two instances of the linear program formalized by (59)-(64). One of these instances represents acceptance of the current request r_k, the other rejection. The difference in the optimal objective values then serves as an estimate for the opportunity cost, which allows us to apply the decision rule in (12).

The DLP covers a horizon of φ^{DLP} days, which is the only hyperparameter of this approach. Note that days are modeled on an aggregate level, i.e., we neglect individual points in time. We use $\vartheta, \varphi \in \{1, \ldots, \varphi^{\text{DLP}}\}$ to denote days in the DLP with $\vartheta, \varphi = 1$ corresponding to the current day τ_k. The existing pending orders are represented by $\tilde{O}_k$. We model the acceptance of the current request r_k by modifying O_k to $\tilde{O}_k$ with $\tilde{o}^k_{dcf} = o^k_{dcf} + \mathbf{1}(d = d_{r_k}, c = c_{r_k}, f = f_{r_k})$ and $\mathbf{1}(\cdot)$ symbolizing the indicator function. For the second instance, which results from rejecting r_k, we set $\tilde{O}_k = O_k$.

The probabilistic information on request arrivals and pickups is incorporated through the following three parameters:

- $\hat{o}_{dc\varphi}$ equals the expected number of future requests with parcel size d, customer type c, and delivery date φ.
- $\bar{p}_{ch\varphi}$ denotes the conditional probability that a parcel belonging to a customer of type c with dwell time h on day τ_k is still in the locker for at least one point in time on day φ. In this way, we include the pickups for orders currently occupying the locker L_k.

- To incorporate pickups of pending orders $\tilde{O}_k$ and future accepted requests, we let $\hat{p}_{c\vartheta\varphi}$ represent the probability that a parcel of customer type c allocated on day ϑ is still in the locker for at least one point in time on day φ.

Lastly, we define decision variable $\tilde{y}_{\delta c\varphi}$ as the number of compartments of size δ to which we tentatively allocate a parcel belonging to a customer of type c on day φ. We formulate the DLP as follows:

$$\max \sum_{c\in\mathcal{C}}\sum_{\delta\in\mathcal{D}}\left(\sum_{\varphi=1}^{\varphi^{\mathrm{DLP}}}\tilde{y}_{\delta c\varphi}-\sum_{f\in\mathcal{F}}\tilde{o}_{\delta cf}\right) \tag{59}$$

$$\sum_{c\in\mathcal{C}}\sum_{j=1}^{\varphi}\hat{p}_{cj\varphi}\tilde{y}_{\delta cj}+\sum_{c\in\mathcal{C}}\sum_{h\in\mathcal{B}}\bar{p}_{ch\varphi}l^{k}_{\delta ch}\leq Q_{\delta} \qquad \delta\in\mathcal{D},\ \varphi\in\{1,\dots,\varphi^{\mathrm{DLP}}\} \tag{60}$$

$$\sum_{d=1}^{\delta}\tilde{y}_{jc\varphi}\leq\sum_{j=1}^{\delta}(\tilde{o}^{k}_{dc\varphi}+\hat{o}_{dc\varphi}) \qquad \delta\in\mathcal{D},\ c\in\mathcal{C},\ \varphi\in\mathcal{F} \tag{61}$$

$$\sum_{j=1}^{\delta}\tilde{y}_{jc\varphi}\leq\sum_{j=1}^{\delta}\hat{o}_{jc\varphi} \qquad \delta\in\mathcal{D},\ c\in\mathcal{C},\ F<\varphi\leq\varphi^{\mathrm{DLP}} \tag{62}$$

$$\sum_{\delta\in\mathcal{D}}\tilde{y}_{\delta cf}\geq\sum_{d\in\mathcal{D}}\tilde{o}^{k}_{dcf} \qquad c\in\mathcal{C},\ f\in\mathcal{F} \tag{63}$$

$$\tilde{y}_{\delta c\varphi}\geq 0 \qquad \delta\in\mathcal{D},\ c\in\mathcal{C},\ \varphi\in\{1,\dots,\varphi^{\mathrm{DLP}}\} \tag{64}$$

The objective (59) maximizes the number of future accepted (and thus allocated) requests. Note that $\tilde{y}_{\delta c\varphi}$ also encompasses the allocations for existing pending orders in $\tilde{O}_k$, which is why we must subtract these orders in the objective. Constraints (60) ensure that the number of compartments is not exceeded, taking into account the current occupancy of the locker, the allocation of pending and future orders over time as well as the pickups. Constraints (61) and (62) guarantee that the orders are assigned to compatible compartments and that the number of allocated parcels does not exceed the number of pending orders and the expected demand-to-come. Constraints (63) state that all orders in $\tilde{O}_k$ must be allocated.

E.3 Full-information benchmark

The full-information benchmark builds on the integer program for the VFA features from Appendix D.1. It covers the entire planning horizon of each instance, which we denote by $\mathcal{F}'$. The full-information benchmark does not undergo a warm-up phase like the other policies, i.e., it starts each instance directly with an empty locker and no pending orders. Moreover, we do not evaluate the state at the end of the horizon and fully focuses on maximizing the number of served requests within $\mathcal{F}'$. All request arrivals and the corresponding pickup behavior are known upfront and represented by a single scenario u with $\hat{o}^{ukx}_{dcf\beta}$, similar to Appendix D.1, but also including

all future arriving requests. We formalize the IP for the full-information benchmark model as follows:

$$\max \sum_{c \in \mathcal{C}} \sum_{f \in \mathcal{F}'} \sum_{\beta \in \mathcal{B}} \hat{y}^u_{\delta c f \beta}$$

subject to constraints (46)-(50) and (55) from Appendix D.1, but substituting $\mathcal{F}$ with $\mathcal{F}'$ and replacing equality constraint (50) with a less-than-inequality. We solve the IP to optimality with Gurobi.

F Hyperparameters

This section explains the process for tuning the hyperparameter values used during our experiments.

For the VFA, we employ the following calibration: To generate the initial state S^x_0, we start with an empty locker and no pending orders and simulate 5 days with the myopic demand control approach FC. The training procedure then runs for $\tau^{\max} = 2000$ days. The remaining hyperparameters are tuned with BO for each setting and VFA variant individually. In case of overbooking, we tune the VFA hyperparameters for $m = \infty$ and utilize them for all failed delivery penalty values. We perform 50 iterations of BO after generating 10 random samples for initialization. In each iteration, we train the VFA and then evaluate it based on 10 customer streams consisting of 10 days each (with a prior warm-up phase of 5 days) for the large instances and 20 days (with a ramp-up of 10 days) for the small instances. After manual pre-tests to avoid divergence, we use the following uniformly distributed priors for our search space: For the update stepsizes, we set $\alpha^\theta = \alpha^{\bar{R}} \in [0.0005, 0.005]$. Regarding exploration, we start with $\varepsilon_1 = 1$ and linearly decay it to $\varepsilon \in [0.05, 0.3]$ until day $\tau \in \{1200, \dots, 1900\}$, after which it is kept constant for the remainder of the training procedure. The ER parameters for V-CER and V-ER include $\kappa \in \{5000, 10000\}$ for the size of the replay memory and $\chi \in \{2000, 4000\}$ for the sample size along with regularization term $\gamma \in [0, 10]$, start $\eta \in \{100, \dots, 300\}$, and frequency $\zeta \in \{5, \dots, 20\}$. To calculate the features, we use $U \in \{5, 10\}$ scenarios. Given that the VFA training is based on simulation and hence subject to uncertainty, we performed pre-tests where we trained every VFA variant five times and examined the average results. The coefficient of variation in the objective value was below 3% for each of our VFA variants, meaning that the learning process produces stable results.

For the DLP, we test a horizon φ^{DLP} of 5, 10, 15, and 20 days and report the results for the best-performing one. To compute the optimal policy, we first perform an aperiodicity transformation on our sequential decision problem with $\tau^{\text{VI}} = 0.8$ and use $\varepsilon^{\text{VI}} = 0.0001$ as the threshold criterion for terminating the algorithm (Puterman 1994).

References

Dimitris Bertsimas and Ioana Popescu. Revenue management in a dynamic network environment. *Transportation Science*, 37(3):257–277, 2003. doi: 10.1287/trsc.37.3.257.16047.

S. Mahadevan. Average reward reinforcement learning: Foundations, algorithms, and empirical results. *Machine Learning*, 22:159–195, 1996. doi: 10.1007/BF00114727.

Martin L. Puterman. *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. John Wiley & Sons, Hoboken, NJ, 1994.

Samyukta Sethuraman, Ankur Bansal, Setareh Mardan, Mauricio G.C. Resende, and Timothy L. Jacobs. Amazon locker capacity management. *INFORMS Journal on Applied Analytics*, 54(6):455–470, 2024. doi: 10.1287/inte.2023.0005.