



Order Fulfillment Under Pick Failure in Omnichannel Ship-From-Store Programs

Sagnik Das,^a R. Ravi,^{a,*} Srinath Sridhar^b

^aTepper School of Business, Carnegie Mellon University, Pittsburgh, Pennsylvania 15213; ^bOnera, Inc., San Francisco, California 94107

*Corresponding author

Contact: sagnik001@gmail.com,  <https://orcid.org/0000-0003-2113-5396> (SD); ravi@cmu.edu,  <https://orcid.org/0000-0001-7603-1207> (RR); srinath@onera.com (SS)

Received: July 8, 2021

Revised: June 23, 2022

Accepted: September 26, 2022

Published Online in Articles in Advance:
December 9, 2022

<https://doi.org/10.1287/msom.2022.1164>

Copyright: © 2022 INFORMS

Abstract. *Problem definition:* We consider the setting where a retailer with many physical stores and an online presence seeks to fulfill online orders using an omnichannel fulfillment program, such as buy-online ship-from-store. These fulfillment strategies try to minimize cost while fulfilling orders within acceptable service times. We focus on single-item orders. Typically, all online orders for the item are sent to a favorable set of locations to be filled. Failed trials are sent back for further stages of trial fulfillment until the process times out. The multistage order fulfillment problem is thus an interplay of the pick-failure probabilities at the stores where they may be shipped from and the picking, shipping, and cancellation costs from these locations. *Methodology:* We model the problem as one of sequencing the stores from which an order is attempted to be picked and shipped in the most cost-effective way over multiple stages. We solve the fulfillment problem optimally by taking into account the changing pick-failure probabilities as a result of other online order fulfillment trials by casting it as a network flow problem with convex costs. We incorporate this as the second stage of a two-stage online order acceptance problem and generalize earlier results to the case with pick failures at stores. *Results:* We investigate the real-world performance of our methods and models on real order data of several of the top U.S. retailers that use our collaborating e-commerce solutions provider to optimize their fulfillment strategies. *Academic/Practical Relevance:* Our work enables retailers to incorporate pick failure in their order management systems for ship-from-store programs. Our new online order-acceptance policies that take into account pick failures can thus create significant savings for omnichannel retailers.

Supplemental Material: The online appendices are available at <https://doi.org/10.1287/msom.2022.1164>.

Keywords: omnichannel fulfillment • pick failure • multistage stochastic programming • dynamic programming • infinitesimal perturbation analysis

1. Introduction

The share of e-commerce as a percentage of total retail sales has steadily grown from 4% in 2010 to 13.6% in 2020 in the United States (U.S. Department of Commerce 2021). The Covid pandemic further accelerated the adoption of e-commerce among customers as the percentage of online sales jumped from 11.8% in quarter 1 2020 to 16.1% in quarter 2 2020. Such trends have both driven and been driven by retailers with traditional brick-and-mortar stores adopting online channels to connect to customers. Omnichannel retailing is the merging of in-store and online channels of demand and fulfillment, where the customer can select from a combination of online and physical channels to place and receive orders. We study the omnichannel ship-from-store (SFS) programs where the retailer uses the inventory at its traditional brick-and-mortar stores to home deliver an order, as opposed to using a dedicated warehouse or fulfillment center.

The advantages of leveraging traditional (brick-and-mortar) stores for fulfillment are many. Making in-store inventory available to fulfill online demand reduces the potential of stocking out. Shipping from a store could be a cheaper and faster option because it is very likely that a store exists near a typical online customer, whereas dedicated fulfillment centers could be much farther away. Finally, with the advent of one- or two-day delivery commitment to customers, a store that is closer to the customer might be able to fulfill the order within the Service Level Agreement, whereas a remote fulfillment center would not be able to do that. Traditional retailers that aim to compete with internet counterparts also view this as a competitive strategy (CNBC 2021).

Challenges in SFS fulfillment arise from the explosion of choices of the location to fulfill an order from, the influence of in-store demand on the process, and the phenomenon of pick failures when online orders are unsuccessfully tried

for fulfillment at stores. Typically, the retailer makes the decision of assigning accepted online orders to stores at the end of a day after the in-store demand at the stores are satisfied. We can solve the choice of locations to fulfill from as a transportation problem with the online orders as demands and leftover inventory at stores after satisfying in-store demand as supplies. Inventory depletion at stores due to in-store demand necessitates the reevaluation of traditional online order acceptance policies: In particular, accepting online orders up to the inventory level of stores could lead to order cancellation after depletion from in-store demand, whereas accepting too few orders could lead to lost sales. This trade-off faced by the retailer was studied in Karp (2017) who provided an optimal joint acceptance and fulfillment policy. In this paper, we focus on modeling the third challenge of pick failures stemming from inventory inaccuracy at stores in the SFS fulfillment and order acceptance process.

In omnichannel SFS programs, as stores become an integral part of a retailer's fulfillment strategy, the likelihood of pick failures increases as a result of inaccurate inventory counts. The problem of inventory inaccuracy in traditional brick-and-mortar stores is known to be significant (DeHoratius and Raman 2008, Shabani et al. 2021). Although warehouses are built for pushing pick, pack, and shipping efficiencies to the limits (Masae et al. 2019), brick-and-mortar stores are not inherently designed for this purpose (Sheffi 2016, Smith 2020). The problem of inventory inaccuracy intensifies in brick-and-mortar stores because of the complexity of the stores, where everything is unpacked and individually displayed and potentially hundreds of people have access to each item and can move (or remove) them without tracking. Even if the inventory is present in the store, the store may be busy, in which case they may fail to pick one or more items because of labor capacity constraints. Other causes of pick failures at stores include shrinkage, theft, damage, incoming delivery receiving errors, labeling and identification issues, human error, disorganized pick locations, and infrequent cycle counting.

Pick success rate ranges from 35% to a high of only 60% (Caro and Sadr 2019). In their study, Skorupa (2015) states that 43.2% of omnichannel retailers found that last minute issues (such as store stockouts) are big issues

preventing omnichannel fulfillment in a timely, efficient manner. Early studies of online service failures by Holloway and Beatty (2003) already highlight nondelivery and late delivery after acceptance of online orders as the most important challenges for companies, which are primarily caused by pick failures and trials respectively in the omnichannel setting.

We observe pick failure in our order fulfillment data (described in more detail in Section 6), as shown in Figure 1, where pick failure is more likely at low inventory levels. We further observe the overall pick-failure rate to be as high as 20% for one of the retailers, meaning one in five orders that are tried at the retailer end up in pick failure. This necessitates a careful incorporation of this feature in modeling the costs in the order fulfillment process.

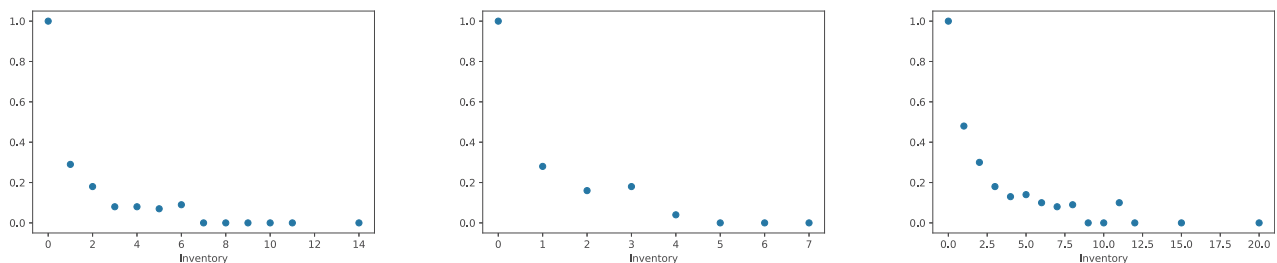
1.1. Model

Order management involves the seamless integration of orders from multiple channels with inventory databases across the entire fulfillment network. For proper execution, the process involves real-time visibility into the entire order life-cycle starting from the placement of order and ending in the fulfillment of the order. This paper is a collaboration with an e-commerce solutions provider (henceforth referred to simply as the *provider*) who helps several of the largest high-end North American retailers with SFS fulfillment. Our designed solutions are integrated with the retailer's Order Management System (OMS).

In practice, retailers usually implement a threshold on the number of orders of a stock keeping unit (SKU) that can be accepted in a day based on the network inventory levels at the beginning of the day. This is implemented by making the SKU unavailable online after accepting these many orders. In the beginning of a typical day, the OMS provides the decision engine the inventory levels of items across the fulfillment network and expects a threshold for the number of orders that can be accepted online.

Once an order placed on the retailer's website is accepted for fulfillment, the OMS provides the decision engine with the customer's zip code, the candidate stores that can fulfill them, and requests routing instructions. A response is expected with routing instructions to route

Figure 1. (Color online) Pick-Failure Probabilities of Single-SKU Orders at Three Stores as a Function of the Inventory



each item in that order to a store, either in real time or at the end of each day. The OMS then executes those routing instructions by notifying the respective store associates on the store terminal to pick, pack, and ship the items that are routed to be fulfilled from that store. Some items will be successfully fulfilled and others will not. A failure would result in a call to the provider's system by the OMS, and the process continues until a maximum number of trials after which all unfulfilled orders are canceled (see Figure 2).

Order acceptance and fulfillment decisions involve three distinct costs: (i) lost-sales cost, (ii) cancellation cost, and (iii) fulfillment cost. The lost-sales costs and the cancellation costs determine the trade-off between not accepting an order that can potentially be filled and accepting an online order that gets cancelled. In our models, we use the profit margin of a product as the lost-sales cost. The cancellation cost is typically set by the retailer to account for customer dissatisfaction. The fulfillment costs associated with one stage of attempted routing fall largely into three buckets. The first is the cost associated with shipping packages to the customer's destination zip code from the store's origin zip code that can be calculated from a shipping rate table. The second cost is the labor cost associated with each pick attempt at each store. This primarily depends on the wages in the store and any excess workload that may have accumulated because of an inflexible workforce. Finally, if too many attempts to route the item have failed, we incur a predefined *delayed cancellation cost* per item cancelled. Note that the delayed cancellation involves an order that is attempted to be picked (potentially several times) and failed, whereas the regular cancellation cost is incurred when it is cancelled without any attempt to fulfill. Because the former involves a delayed communication of a negative (cancellation) event, the delayed cancellation cost is typically set higher than the cancellation cost.

We describe the order fulfillment models under pick failure for the OMS under stochastic online and physical demand, and extend one of them to devise order acceptance policies incorporating pick failures in the scheme of Karp (2017) and Jia et al. (2022). We restrict our study

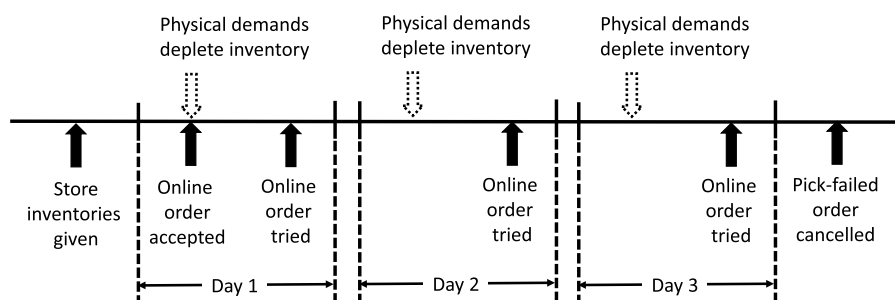
to single-SKU orders to simplify the problem. (We note that single-SKU orders are known to form the bulk of online orders in practice.) Recall that online orders are attempted to be filled in several stages. The input to each of our models is the pick-failure probability as a function of inventory level at each store at each stage. We make the assumption that the pick-failure probability for an SKU at a store is a nonincreasing function of available inventory at the store. This is natural because inventory inaccuracy at low inventory levels is more likely to cause a pick failure (as seen in Figure 1). The other inputs to the models are the various cost components.

We propose three order fulfillment models where physical and online demand are both sparse, the physical demand is dense, and both demands are dense, respectively. We then extend the third model to incorporate the order acceptance decision.

1. In Section 2, we model the online order fulfillment problem when the physical and online demands are sparse. Sparse online and physical demands ensure that fulfilling these demands does not significantly impact the inventory level at stores. Given this sparsity, the OMS can make real-time multistage fulfillment decisions for an accepted online order. The fulfillment decision for an accepted online order is a sequence of stores that minimizes the expected fulfillment cost. In computing the fulfillment cost, the pick failure-probability for a store is determined using the current inventory levels at the stores and is assumed to remain the same in future stages in this case of sparse demands. We provide a polynomial time dynamic programming-based algorithm to solve for the optimal fulfillment decision based on a simple observation about the structure of optimal solutions.

2. In Section 3, we generalize the previous model by considering stores that have nonsparse physical demand. Similar to the previous model, the sparsity of online orders implies that the fulfillment decisions of online orders do not significantly influence one another. Again, the OMS can make real-time multistage fulfillment decisions for an accepted online order. Using the current inventory level, the distribution of nonsparse physical demands at various

Figure 2. Timeline for Three-Stage (Day) Single-Order Models in Sections 2 and 3



Note. Dotted arrows feature in Section 3.

stores, and the pick-failure probability function, the OMS can estimate the future pick-failure probabilities at the stores in subsequent stages. The fulfillment problem to determine the optimal sequence of stores for an accepted online order is solved by considering the fulfillment costs and the imputed pick-failure probabilities for stores across stages. We improve upon the naive dynamic programming algorithm for this case and provide an efficient solution with much better complexity.

3. In Section 4, we propose a general fulfillment model with nonsparse physical and online demand. Multiple online orders for the same SKU accepted from various customer shipping zones are to be either cancelled or fulfilled using inventory at multiple stores. The inventory at the stores is shared by the online and physical demands. Because the online orders are no longer sparse, we assume that fulfillment decisions for all the accepted online orders are made and executed at the end of a day by the OMS after physical demands at stores have been satisfied (thus giving implicit preference for physical demand at each location). Because of the complexity of the problem, we consider only the single-stage version of the fulfillment problem, where the first pick failure of an order leads to cancellation without retrials (see Figure 3). We provide a network-flow-based algorithm to solve this fulfillment problem.

4. Finally, we propose the joint order acceptance and fulfillment model with nonsparse physical and online demand. The two decisions in the model are (i) real-time acceptance/rejection for an online order originating from various customer shipping zones and (ii) fulfillment decisions involving store assignment for the accepted orders after physical demand at the stores has been satisfied. The real-time acceptance/rejection decision involves setting a threshold on the online orders that can be accepted. In addition to accounting for fulfillment cost of an accepted online order as modeled above, the joint acceptance and fulfillment model accounts for the trade-off between the lost sales and cancellation cost for a rejected and cancelled online order, respectively. We model the problem as a two-stage stochastic program

where the first stage involves the acceptance/rejection of online orders and the second stage deals with the optimal fulfillment decisions for the accepted orders that can be attempted for fulfillment. Our fulfillment model in Section 4 can be conveniently adapted to solve this second stage problem while incorporating pick-failure costs. Using this to calculate gradients, we propose an Infinitesimal Perturbation Analysis-based method algorithm to efficiently solve the joint acceptance and fulfillment problem (Section 5).

Table 1 summarises the four models in the paper.

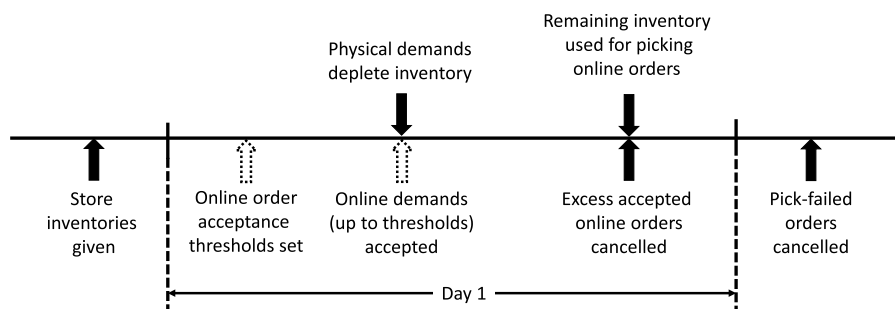
1.2. Related Work

1.2.1. Omnichannel Fulfillment. Xu et al. (2009) considered the reconsolidation of as-yet unshipped orders to minimize the number of shipments.

Acimovic and Graves (2014) study the problem of fulfillment decisions in real-time, as opposed to batched processing. Their work is similar to ours because it is derived from the retail use case. Their focus is to model future demand and the potential resulting splits of multi-item orders that could occur in the future because inventory could be depleted suboptimally without considering correlation among the demands for items in future orders. Although their model assumes no inventory inaccuracy or pick failure, we focus on optimally solving the current order that needs to be fulfilled, with the possibility of pick failure at stores. We have kept our models oblivious to future orders, but stochastic from a current order's evolution perspective, while restricting our study to single-item orders.

Markowicz (2017) models the batched version of the fulfillment problem using replenishment and in-store demand, studies the trade-offs between distribution center and store fulfillment, and finds routing strategies heuristically using a linear program (LP) that increases profitability. Their work primarily focuses on the benefits of batching orders for fulfillment, whereas we focus on near real-time fulfillment decisions that are more pertinent in a retail environment.

Figure 3. Timeline for Multiunit Orders (Sections 4 and 5)



Note. Dotted arrows feature in Section 5.

Table 1. Features of Models Studied in the Paper

Model	Section	Online demand	Physical demand	Stages	Solution method
Single-order static-pick-fail fulfillment	(Section 2)	Sparse	Sparse	Multiple	DP
Single-order dynamic-pick-fail fulfillment	(Section 3)	Sparse	Dense	Multiple	DP
Multiorder fulfillment	(Section 4)	Dense	Dense	Single	Net-flow
Multiorder acceptance & fulfillment	(Section 5)	Dense	Dense	Single	IPA

Note. DP, dynamic programming; IPA, infinitesimal perturbation analysis; Net-flow, network flow.

The work of Jasin and Sinha (2015) is close to our work in that they minimize the picking and shipping cost of items. They consider a batch of multi-item orders with inventory constraints at fulfillment centers/stores. Jasin and Sinha (2015) incorporate a shipping function that has a fixed component and a variable component. They formulate the fulfillment problem as a linear program and modify the solution of the LP using correlated rounding (or coupling) among the decision variables. The motivation behind correlated rounding is to consolidate items and, therefore, reduce the number of shipments similar in spirit to Xu et al. (2009). Our model differs from theirs as we explicitly model pick failure (and therefore allow multiple trials) and solve the problem for single orders as they arrive rather than batches. The work in Jasin and Sinha (2015) has been extended to joint pricing and fulfillment in Lei et al. (2018a) and further to the joint pricing, display, and fulfillment problem in Lei et al. (2018b).

Harsha et al. (2019) study the problem of price optimization under an omnichannel framework to unify the in-store and online prices. In their work, they mitigate pick failure by prescribing a thresholding policy that does not allow order fulfillment from a store when inventory goes below a threshold. In contrast, in our work, we assume probabilities of pick failure for stores are given typically as a function of inventory. Karp (2017) has discussed the estimation of pick-failure probabilities as a function of inventory from data.

Acimovic and Farias (2019) and Andrews et al. (2019) optimize total reward earned from feasibly fulfilling all orders that arrive at the system over a finite horizon. They formulate the problem as a large dynamic programming problem and use approximate dynamic programming techniques to make the problem tractable by assuming a separable reward function for each item.

1.2.2. Omnichannel Acceptance to Mitigate Failures. The features we model and study, like pick failure and multiple trials for order fulfillment, have not been considered explicitly in prior work. The closest work that has tried to circumvent pick failure in the literature is Harsha et al. (2019) who have a threshold inventory level below which items ordered online are not attempted to be picked at stores. Karp (2017) provides policies for computing such threshold inventory levels optimally or nearly so based on the knowledge of the demand distributions

corresponding to online and in-store walk-in demand. In batch processing (Acimovic and Farias 2019, Andrews et al. 2019), one might argue that pick failure is considered implicitly in the sense that it is less likely for (real) inventory at a store to go to zero over the longer horizon for batch processing due to replenishment. However, the batch processing model provides limited decision support for routing real-time orders dynamically that are the norm in modern omnichannel order systems.

Our work is also the first to integrate pick-failure consideration into the tactical fulfillment problem and use this in extending the order acceptance problem formulated in Karp (2017) to incorporate pick failures in the fulfillment cost. This allows us to realistically adjust the thresholds for online order acceptance more conservatively by incorporating the additional costs of fulfillment that have not been modeled in prior work.

2. Single Order: Static Pick-Failure Probabilities

In this section, we model an OMS making real-time fulfillment decisions for online orders when online and physical demands are sparse. Low demand leads to stable inventory levels at stores. Therefore, the pick-failure probabilities at stores are assumed to be constant across the stages of fulfillment of the current order. We formally describe the multistage single-order fulfillment model and provide a polynomial time dynamic programming algorithm to solve the fulfillment problem optimally based on a structural result.

Consider fulfilling an online order from a fulfillment network of J stores in L stages. At each stage, an attempt to pick the order incurs a fixed attempt cost b_j at store $j \in [J]$ where $[J]$ denotes $\{1, 2, \dots, J\}$. A pick attempt at store j fails with probability ϕ_j , which is independent of the attempts at any other store and of the stage of fulfillment trial using our assumption of sparse demand. If the pick attempt at store j is successful, a shipping cost s_j is incurred. In the event of a pick failure, we proceed to the next stage or attempt. Finally, after L pick attempts, we terminate paying a delayed cancellation cost of d .

We define the *unreliability factor* r_j for store j to be $r_j = s_j + \frac{b_j}{1-\phi_j}$. A *fulfillment policy* σ is a (nonrepeating) sequence of L stores where $\sigma(l)$ represents the store tried

in stage l . The expected cost $\mathcal{V}(\sigma)$ of fulfillment policy σ can be expressed as follows (with the convention that $\phi_{\sigma(0)} = 1$).

$$\mathcal{V}(\sigma) = \left[\sum_{l=1}^L \left(\prod_{l'=1}^{l-1} \phi_{\sigma(l')} \right) \cdot [b_{\sigma(l)} + (1 - \phi_{\sigma(l)})s_{\sigma(l)}] \right] + \left[\left(\prod_{l=1}^L \phi_{\sigma(l)} \right) \cdot d \right] \quad (1)$$

$$= \underbrace{\left[\sum_{l=1}^L \left(\prod_{l'=1}^{l-1} \phi_{\sigma(l')} \right) \cdot [(1 - \phi_{\sigma(l)}) \cdot r_{\sigma(l)}] \right]}_{\text{expected cost at stage } l \text{ if pick success}} + \underbrace{\left[\left(\prod_{l=1}^L \phi_{\sigma(l)} \right) \cdot d \right]}_{\text{expected cancellation cost}} \quad (2)$$

The objective is to determine the policy σ^* that minimizes expected fulfillment cost.

Lemma 1. *Let σ^* be the optimal L -stage policy. Then, $r_{\sigma^*(l)} < r_{\sigma^*(l+1)}$ for any l in $1, \dots, L-1$. In words, the unreliability factor of a store in the optimal policy is lower than that of all subsequent stores in that policy.*

The proof relies on the fact that swapping a store with higher r_j value tried at an earlier position with a store with lower r_j value tried at a later position yields a policy with lower fulfillment cost.

From Lemma 1, it may seem that the optimal policy is simply trying the first L stores ordered in nondecreasing r_j values. However, this is false because of the effect of the delayed cancellation cost d in the overall cost, which we illustrate with a simple counterexample. Consider a one-stage problem with two stores, say store 1 and store 2, such that $(r_1, r_2) = (\$10, \$15)$, $(\phi_1, \phi_2) = (0.9, 0.2)$ and $d = \$30$. We have $\mathcal{V}(1) = \$28$, whereas $\mathcal{V}(2) = \$18$ even though $r_1 < r_2$.

2.1. Dynamic Program

We propose a dynamic program to solve the problem using the Lemma 1. Reorder the set of stores $[J]$ in nondecreasing order of r_j —this can be done in $O(J \log J)$ time. Define $W(j', l')$ to be the fulfillment cost of the optimal subsequence of l' remaining picks to be performed as a subsequence of the set of stores $[j', j' + 1, \dots, J]$ in this order, for j' ranging from one to L . By Lemma 1, these values obey the following recurrence.

$$W(j', l') = \min_{j \in [j', \dots, J-l']} \{(1 - \phi_j) \cdot r_j + \phi_j \cdot W(j+1, l'-1)\}$$

Note that $W(j, 0) = d$ for all stores $j \in [J]$. Because we index the optimal values in the subproblems by stores and stages, there are $J \cdot L$ such values and each one can be

computed by examining up to J terms giving a total running time $O(J^2L)$.

Theorem 1. *For the single-order fulfillment problem under pick failure with J stores and L stages, there exists an $O(J^2L)$ time algorithm to compute the optimal fulfillment policy.*

3. Single Order: Dynamic Pick-Failure Probabilities

In this section, we expand the scope of the OMS decisions from Section 2 by considering stores with dense physical demand but sparse online demand. Because the online demand is sparse, each accepted online order must be filled before the next one; thus, the requirement of real-time fulfillment decisions for an incoming order is justified. We generalize the result of the last section to the case when the pick-failure probabilities at stores in subsequent stages vary because of inventory depletion from incoming physical demand.

We generalize the notation ϕ_j used in Section 2 for pick-failure probability at store j to ϕ_{jl} for pick-failure probability at store j at stage l . If there is no replenishment at store j , we would expect ϕ_{jl} to be nondecreasing as a function of l ; but we make no assumptions on these functions in our solution below. Similar to the fulfillment policy in Section 2, σ is a sequence of stores such that $\sigma(l)$ represents the store tried in stage l . The expected cost of fulfillment policy σ for the L stage problem with dynamic pick-failure probabilities is

$$\mathcal{V}(\sigma) = \left[\sum_{l=1}^L \left(\prod_{l'=1}^{l-1} \phi_{\sigma(l'), l'} \right) \cdot [b_{\sigma(l)} + (1 - \phi_{\sigma(l), l})s_{\sigma(l)}] \right] + \left[\left(\prod_{l=1}^L \phi_{\sigma(l), l} \right) \cdot d \right] \quad (3)$$

$$= \underbrace{\left[\sum_{l=1}^L \left(\prod_{l'=1}^{l-1} \phi_{(\sigma(l'), l')} \right) \cdot [(1 - \phi_{\sigma(l), l}) \cdot r_{\sigma(l), l}] \right]}_{\text{expected cost from picking and shipping}} + \underbrace{\left[\left(\prod_{l=1}^L \phi_{\sigma(l), l} \right) \cdot d \right]}_{\text{cancellation expected cost}} \quad (4)$$

The objective is to determine the policy that minimizes expected fulfillment cost $\min_{\sigma} \mathcal{V}(\sigma)$. We introduce a notation $\sigma[l : L]$, which denotes a sequence of nonrepeating stores for stages l through L .

3.1. A Natural Dynamic Program

A natural dynamic programming algorithm for the problem would use $O(J^L)$ states: we consider all J stores for the first stage of fulfillment; for each of these J stores, we have

a choice of $J - 1$ choices for the next stage. Proceeding this way, the number of states in stage L grows to $J \cdot J - 1 \cdots (J - L) = J!/L! = O(J^L)$. In typical instances, the number of potential stores for fulfillment $J \gg L$, the number of stages of fulfillment. Hence, we devise a different DP algorithm with running time $O(J \log J \cdot L!)$, which is much faster in this regime.

Lemma 2. Let σ^* be the optimal L -stage fulfillment policy. Among all stores $j \in J \setminus \sigma^*[l+1, L]$, let \mathcal{J}_l be the set of l stores that achieve the l lowest values of $(1 - \phi_{jl}) \cdot r_{jl} + \phi_{jl} \cdot \mathcal{V}(\sigma^*[l+1 : L])$. Then, there exists a store $j \in \mathcal{J}_l$ such that $\sigma^*(l) = j$.

Proof. Consider the set $\mathcal{J}_l' = \mathcal{J}_l \setminus \{\sigma^*(1), \dots, \sigma^*(l-1)\}$. Because $|\mathcal{J}_l| = l$, there exists a store $j \in \mathcal{J}_l'$. Suppose $\sigma^*(l) = j'$ such that $j' \notin \mathcal{J}_l'$. Substituting j' with j will yield a lower cost policy than $\mathcal{V}(\sigma^*)$ giving a contradiction. \square

3.2. A More Efficient DP

We proceed in stages from stage L to stage 1. A node or state in stage l is represented by $(\sigma[l : L])$ with value $\mathcal{V}(\sigma[l : L])$. Construct a dummy node (NULL) for initial stage $L + 1$ with value $\mathcal{V}(\text{NULL}) = d$. We start with stage L and obtain \mathcal{J}_L , the set of L stores with the lowest values of $(1 - \phi_{jl}) \cdot r_{jl} + \phi_{jl} \cdot \mathcal{V}(\text{NULL})$ among stores $j \in J$. For each of these L stores, we create nodes (j) with value $\mathcal{V}(j)$.

We proceed in this fashion for an arbitrary stage l . Consider a node in stage $l + 1$, $(\sigma[l + 1 : L])$ with value $\mathcal{V}(\sigma[l + 1 : L])$. We obtain \mathcal{J}_l , the set of l stores with the lowest values of $(1 - \phi_{jl}) \cdot r_{jl} + \phi_{jl} \cdot \mathcal{V}(\sigma[l + 1 : L])$ among stores $j' \in J \setminus \sigma[l + 1 : L]$. For each such node $j' \in \mathcal{J}_l$ create a node $\sigma[l : L] = j' \cup \sigma[l + 1 : L]$ with value $\mathcal{V}_l(j' \cup \sigma[l + 1 : L])$ equal to that of trying j' in level l followed by the sequence $\sigma[l + 1 : L]$ in the following stages. See Algorithm 1 for details.

Algorithm 1 (Algorithm to Solve the L -Stage Fulfillment Problem)

Input: $[J], [L], d, \phi_{jl}$ and $r_{jl} \forall j \in [J], l \in [L]$
Output: $\sigma^*, \mathcal{V}(\sigma^*)$

- 1: Create a dummy node at stage $L + 1$, (NULL)
- 2: $\mathcal{V}(\text{NULL}) \leftarrow d$.
- 3: **for** stage l in $L, \dots, 1$ **do**
- 4: **for all** nodes $(\sigma[l + 1 : L])$ in stage $l + 1$ **do**
- 5: $\tilde{J} \leftarrow [J] \setminus \sigma[l + 1 : L]$
- 6: **for** $j' \in \tilde{J}$ **do**
- 7: $U[j'] \leftarrow (1 - \phi_{jl}) \cdot r_{jl} + \phi_{jl} \cdot \mathcal{V}(\sigma[l + 1 : L])$
- 8: $\mathcal{J}_l \leftarrow l$ stores in \tilde{J} with lowest values of $U[j']$
 store $j \in \tilde{J}$ (Break ties arbitrarily).
- 9: **for all** $j \in \mathcal{J}_l$ **do**
- 10: $\sigma[l : L] \leftarrow j \cup \sigma[l + 1 : L]$
- 11: Create node $(\sigma[l : L])$ in stage l
- 12: $\mathcal{V}(\sigma[l : L]) \leftarrow U[j]$
- 13: $\sigma^* \leftarrow \arg \min_{\sigma[1:L]} \mathcal{V}(\sigma[1 : L])$
- 14: $\mathcal{V}(\sigma^*) \leftarrow \min_{\sigma[1:L]} \mathcal{V}(\sigma[1 : L])$

3.3. Time Complexity of Algorithm 1

For each node in stage $l + 1$, (i) we generate a set of l nodes \mathcal{J}_l and then (ii) create l nodes in this stage. The sorting according to the value of U takes $O(J \log J)$ time. There are $L!/(l-1)!$ nodes in stage l , so constructing stage l along with the values of all the nodes in it takes $O(\frac{L!}{(l-1)!} J \log J)$ time. Stage 1 takes the maximum time $O(L! \cdot J \log J)$.

Theorem 2. There exists a $O(L! \cdot J \log J)$ algorithm that finds the optimal fulfillment policy for the L -stage fulfillment problem when the pick-failure probabilities at stores vary across stages.

Note that Algorithm 1 also solves the L -stage fulfillment problem with static pick-failure probabilities across stages in $O(L! \cdot J \log J)$ time as an alternate to the Dynamic Programming algorithm in Section 2, which solves the problem in $O(J^2 L)$ time and hence will be faster when $J \gg L!$.

4. Multiple Orders with Single Stage of Fulfillment

Let us consider the fulfillment problem where online demand accepted from various customer shipping zones and physical demand arising at stores are nonnegligible and hence must be accounted for their effect on inventory depletion, which in turn influences the pick failures. We describe the fulfillment problem in the time frame of a day to simplify the exposition. During the day, the retailer observes stochastic online demand from customer shipping zones and stochastic physical demand at stores. During the day, the online demands originating from various zones are accepted/rejected in real time. We defer the modeling of this real-time acceptance/rejection decision to the next section. For now, we assume that we are given the accepted online orders at various locations as input to our model. Retailers typically avoid making real-time fulfillment decisions for accepted online orders for the following reasons.

1. Pick attempts at stores during the day hampers the daily operations of a store.
2. Holding online orders until the end of the day prevents cases when physical demand at a store is unfulfilled on account of satisfying online demand.
3. A real-time fulfillment decision made for an accepted online order impacts the inventory at the assigned store and therefore affects the fulfillment decision of subsequent online orders. Thus, making real-time fulfillment decisions for each accepted online order could potentially lead to suboptimal fulfillment decisions compared with batching them at the end of the day.

For efficient fulfillment planning, the OMS accumulates all accepted online orders until physical demands at stores have been satisfied, typically by the end of the day. The inputs to the fulfillment model are A_i^O , i.e., the

number of online orders accepted from customer shipping zone $i \in [I]$ and Q_j^p , i.e., the inventory level at store j at the end of the day. At fulfillment time, an online order accepted from customer shipping zone i is either cancelled incurring a cancellation cost c_i or attempted for fulfillment at one of the stores in the fulfillment network. An order accepted from zone i that is attempted for fulfillment at store j (i) incurs a labor cost b_j for the pick trial, (ii) a shipping cost s_{ij} if it is found and shipped, and (iii) a delayed cancellation cost d_i if the pick attempt(s) was unsuccessful and the order was eventually cancelled. (Note that we may assume without loss of generality that $d_i \geq c_i$.) Thus, the expected fulfillment cost of an online order at a store depends on the pick-failure probability, which is a function of the inventory level at the store during the pick attempt. For store j , we define a pick-failure probability $\phi_j(q)$ as a function of the store's inventory level q . Note that if there is no inventory, then $\phi_j(0) = 1$, meaning picks will always fail. We use a natural assumption that the pick-failure probability for an item at a store is a nonincreasing function of available inventory at the store.

Assumption 1. $\phi_j(q)$ is a nonincreasing function of inventory level q at store j .

The inventory level at stores in the future stages of fulfillment depends on the fulfillment decision (assignment of accepted online orders to stores) and the outcome of the various pick attempts. This makes it hard to model the problem for multiple stages of pick attempts. We therefore restrict ourselves in this section to a single-stage model in which a failed pick attempt of an online order leads to cancellation without retries.

We define the expected fulfillment or trial cost of an order from customer shipping zone i at store j when its inventory level is q as follows.

$$t_{ij}(q) = b_j + [1 - \phi_j(q)] \cdot s_{ij} + \phi_j(q) \cdot d_i$$

A successful pick attempt for an online order decreases the inventory level by one, whereas a failed attempt does not impact the inventory level. (Note that we do not allow replenishment in our model.) It follows that the inventory level at which an online order assigned to a store is tried depends on the pick success/failure outcome of the online orders tried before at the same store. Therefore, the inventory level at which an order is attempted depends on its *position* among all the other online orders that are also tried at the same store. We introduce the index k to represent the position of an online order among all the other online orders that are assigned to be tried at that store. An online order with position k at store j is simply the k^{th} online order that is assigned to be tried at store j .

To account for potential drops in the inventory levels during the pick attempts for online orders, we define

Q_{jk}^O , which represents the (random) inventory level at store j when the k^{th} online order is tried at store j . We can now define $t_{ijk}(Q_j^p)$ as the expected trial cost for an order from customer shipping zone i if it is the k^{th} order assigned to be tried at store j as follows. Because location j starts with physical inventory Q_j^p at the end of the day, when the k^{th} online order is tried, its inventory value must lie in the range $[Q_j^p - (k - 1), Q_j^p]$.

$$t_{ijk}(Q_j^p) = \sum_{q=Q_j^p-(k-1)}^{Q_j^p} \Pr[Q_{jk}^O = q] \cdot t_{ij}(q) \quad (5)$$

The expression for the expected fulfillment cost can be simply seen as the sum of expected fulfillment costs at potential inventory levels for an order tried at position k weighted by the likelihood of those inventory levels when this order is tried.

A *fulfillment policy* is an assignment of accepted orders to either a store for fulfillment or for cancellation. The objective is to compute the fulfillment policy that minimizes the cancellation and expected fulfillment cost. In the absence of pick failure, the fulfillment problem can be modeled as a network flow problem (Karp 2017). This approach cannot be directly extended to account for pick failures because the fulfillment costs for different online orders assigned to a store are no longer identical but depend on their trial position. In this section, we show that even under pick failure, the fulfillment costs for trying online orders at a store are convex, which still makes the problem amenable to a network flow formulation.

We now state a key lemma that implies the convexity of the fulfillment costs and is proved (in Online Appendix A.2) using the intuition that it is costlier to try an online order at a later position in a store that will be at a more disadvantageous inventory position.

Lemma 3. Under Assumption 1, $t_{ijk}(Q_j^p)$ is nondecreasing in k .

4.1. Linear Program

We now describe a linear program to solve the fulfillment problem. Define x_{ijk} to be a (binary) variable that takes the value one if an accepted online order from customer shipping zone i is assigned to be the k^{th} online order attempted for picking at store j after all the physical demand at store j is satisfied and zero otherwise. Recall that a successful pick attempt reduces the inventory level at a store by one. This implies that any order assigned to a store in a position higher than Q_j^p is only tried at inventory level zero, incurring a delayed cancellation cost. Because the (regular) cancellation cost is no more than the delayed cancellation cost, it is suboptimal to try to fulfill an order at a store at nonpositive inventory levels. Therefore, we can restrict the index k for the decision variable x_{ijk} up to Q_j^p .

The fulfillment objective function minimizes the sum of cancellation and expected fulfillment costs. Each

accepted order can either be assigned to a store for fulfillment or cancelled.

$$\sum_{i \in [I]} \sum_{j \in [J]} \sum_{k \in [Q_j^P]} t_{ijk}(Q_j^P) \cdot x_{ijk} + \underbrace{\sum_{i \in [I]} c_i \cdot \left(A_i^O - \sum_{j \in [J]} \sum_{k \in [Q_j^P]} x_{ijk} \right)}_{\text{orders cancelled at fulfillment}}$$

The objective function can be rearranged and written as

$$\underbrace{\sum_{i \in [I]} c_i \cdot A_i^O}_{\text{constant}} + \sum_{i \in [I]} \sum_{j \in [J]} \sum_{k \in [Q_j^P]} [t_{ijk}(Q_j^P) - c_i] \cdot x_{ijk}.$$

We can interpret this expression as if we incur a cancellation cost for all accepted orders and then incur the expected fulfillment cost minus the savings of the cancellation cost for the orders that are assigned to stores for fulfillment. Because the first term in the objective function does not involve decision variables x_{ijk} , we can rewrite the mathematical programming formulation of the fulfillment problem as follows.

$$\sum_{i \in [I]} c_i \cdot A_i^O + \min_{x_{ijk}} \sum_{i \in [I]} \sum_{j \in [J]} \sum_{k \in [Q_j^P]} [t_{ijk}(Q_j^P) - c_i] \cdot x_{ijk} \quad (6)$$

$$\text{such that } \sum_{j \in [J]} \sum_{k \in [Q_j^P]} x_{ijk} \leq A_i^O \quad \forall i \in [I] \quad (7)$$

$$\sum_{i \in [I]} x_{ijk} \leq 1 \quad \forall j \in [J], k \in [Q_j^P] \quad (8)$$

$$x_{ijk} \in \{0, 1\} \quad \forall i \in [I], j \in [J], k \in [Q_j^P] \quad (9)$$

Constraints (7) ensure that the number of orders assigned from each zone to stores are lower than the number of orders accepted from that zone. Constraints (8) ensure

that at most one order can be assigned to each position at each store.

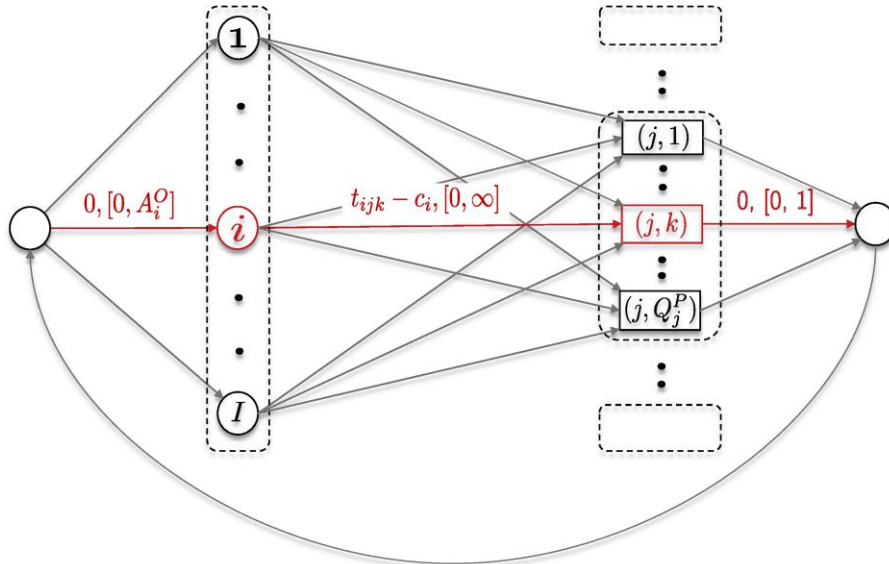
Theorem 3. For the multiple-unit fulfillment problem under pick failure with I zones and J stores, if the trial costs t_{ijk} can be estimated efficiently, then the optimal fulfillment policy can be computed by solving a min-cost circulation problem with $O(I + JK)$ nodes and $O(IJK)$ arcs, where $K = \max_{j \in [J]} Q_j^P$.

Proof. We show that the integer program (6) can be formulated as a min-cost circulation problem and therefore solved efficiently. To construct the formulation as a minimum cost circulation problem (see Figure 4), we join a new source node to nodes corresponding to zone $i \in [I]$ with arcs of capacity A_i^O and cost zero. Connect zone node i with node (j, k) representing store $j \in [J]$ in position $k \in [Q_j^P]$ with arcs of cost $t_{ijk} - c_i$. Finally, we connect nodes (j, k) to a new sink node with zero-cost capacity-1 arcs. Using Lemma 3, we see that if any flow is assigned to store j at position k (i.e., going via the node labelled (j, k)) in a minimum-cost circulation, then we can assume that there must be unit flow in all nodes of lower index (j, k') for $k' < k$; otherwise, the same flow routed through a lower index position has no larger cost. Thus, any optimal solution to the circulation problem gives a feasible fulfillment schedule for all accepted orders.

Observe that an online order accepted from zone i is assigned to position k at a store j if the expected fulfillment cost of that order is lower than the cancellation cost, i.e., $t_{ijk} < c_i$. Consequently, the optimal circulation will have negative cost corresponding to the second term in the objective function (6).

Thus, when the trial costs t_{ijk} on the arcs is given, using known algorithms for solving minimum cost

Figure 4. (Color online) Network Flow Formulation for Multiple Online Order Fulfillment



Note. Arcs are labelled with cost, [lower bound, upper bound] of the flow.

circulation, such as Tardos (1985), we can obtain an optimal solution to the fulfillment problem in time polynomial in I , J , and K . \square

Even though the fulfillment problem can be written as a min-cost circulation problem with polynomial number of nodes and arcs, the problem may not be solvable in polynomial time because of the complexity in computing the cost of the arcs under a general decreasing pick-failure probability function. In Online Appendix B, we describe how we can compute these probabilities by making some simplifying assumptions. In particular, we fit a simple parametric one-step function to the data provided for the pick-failure probability ϕ_j at store j as a function of the initial inventory and the current trial number k and use this to evaluate the expected trial cost t_{ijk} . In other words, rather than use the more complex expression in Equation (5), we use $t_{ijk}(Q_j^P) = b_j + (1 - \phi_j) \cdot s_{ij} + \phi_j \cdot d_i$ for k large enough and $b_j + s_{ij}$ otherwise.

5. Joint Acceptance and Fulfillment with Stochastic Demands

We generalize the fulfillment problem in the previous section by integrating the online order acceptance decision into the model. We also extend the joint acceptance and fulfillment problem studied by Karp (2017) by incorporating pick failure. As in the previous section, we restrict our focus to a single-stage fulfillment model where a failed pick attempt of an online order leads to cancellation without retrials.

We continue with the time frame of a day to describe the joint acceptance and fulfillment problem. During the day, the retailer observes stochastic online demands \mathcal{D}_i^O from zone $i \in [I]$ and stochastic physical demands \mathcal{D}_j^P at stores $j \in [J]$. Let the inventory at the beginning of the day at store j be \bar{Q}_j . The decision making of the OMS occurs in two stages. The online demands originating from various zones during the day are accepted/rejected in real time, and then the fulfillment decisions for all the accepted online orders are made at the end of the day.

5.1. Costs

In the previous section, we studied the trade-off between trying to fulfill an accepted order (incurring an expected fulfillment cost, which can be negative) and cancelling an accepted order (incurring a cancellation cost). Because the decision to accept/reject an online order is now in the purview of our model, we have to take into account the lost-sales cost associated with rejecting an order that could have been fulfilled. As a result, each unit of online demand up to the total inventory minus total physical demand incurs one of the following costs: lost sales if it is rejected, cancellation cost if it is accepted but not tried for fulfillment, and expected fulfillment cost when it is tried.

The notations of all the cost components carry over from the previous section with the exception of lost-sales cost, which we introduce in this section. As in Karp (2017), we use the profit margin p as the lost-sales cost for an online order.

5.2. Order Acceptance Policies

We follow Karp (2017) and consider the following classes of thresholding policies for real-time acceptance/rejection of online orders.

1. **Global Threshold Policy:** The retailer enforces a *global* threshold on the total number of online orders that can be accepted. In practice, the retailer ceases to offer the product online once the designated number of online orders has been accepted.

2. **Local Threshold Policy:** In the local threshold policy, the retailer accepts an online order up to a *local* threshold designated for each customer zone i .

We first describe a simple version of the problem in Section 5.3 and then proceed to solve the general problem in Section 5.4.

5.3. Simple Model: One Store and One Customer Shipping Zone

We describe the joint order acceptance and fulfillment problem under pick failure with one store and one customer shipping zone. Let $Q^P = \max\{0, \bar{Q} - \mathcal{D}^P\}$ be a random variable representing the inventory level at the store after physical demand \mathcal{D}^P has been satisfied. Let S be the decision variable that represents the threshold on the number of online orders that can be accepted. Note that in this simple variant, the global and local threshold policies coincide. Our objective is to determine the optimal threshold S such that the cancellation and expected fulfillment costs are minimized. The number of accepted online orders is $\min\{\mathcal{D}^O, S\}$. We use the following assumption.¹

Assumption 2 (Cost Structure). *The cost of trying to fulfill an order is less than the cancellation cost, i.e., $t_k(Q^P) < c \quad \forall k$.*

Under the assumption that expected fulfillment cost is lower than cancellation cost, it is optimal to try fulfillment of accepted online orders up to inventory Q^P at the store for fulfillment. The number of such orders is $\min\{\mathcal{D}^O, S, Q^P\}$. The remaining accepted orders not assigned for trial are naturally cancelled. The number of such orders is $\min\{\mathcal{D}^O, S\} - \min\{\mathcal{D}^O, S, Q^P\}$.

The lost-sales cost is incurred on orders that are rejected but could have been tried for fulfillment. The number of such orders can be expressed as the minimum of the number of orders that are rejected and the inventory at the store after physical demand and online orders assigned for trial at the store. The joint order acceptance and fulfillment problem reduces to solving the following

(one-stage) stochastic program.

$$\begin{aligned} \min_S \quad & \mathbb{E}_{\mathcal{Q}^O, \mathcal{Q}^P} \left[\sum_{k=1}^{\min\{\mathcal{Q}^O, S, \mathcal{Q}^P\}} t_k(\mathcal{Q}^P) + c \cdot [\min\{\mathcal{Q}^O, S\} \right. \\ & \left. - \min\{\mathcal{Q}^O, S, \mathcal{Q}^P\}] \right] \\ & + \mathbb{E}_{\mathcal{Q}^O, \mathcal{Q}^P} [p \cdot \min\{\mathcal{Q}^O - \min\{\mathcal{Q}^O, S\}, \mathcal{Q}^P \\ & - \min\{\mathcal{Q}^O, S, \mathcal{Q}^P\}\}] \end{aligned} \quad (10)$$

Lemma 4. *The objective function (10) is unimodal in S .*

This lemma is proved by examining the effect of a unit change in S on the objective using its derivative and its properties. The proof is in Online Appendix A.3.

Corollary 1. *The optimal threshold for the single store problem with no shipping and base trial costs in the absence of pick failure, as obtained by Karp (2017), is greater than S^* .*

The corollary is proved using the observation that trials are less profitable in the presence of pick failure. The proof is in Online Appendix A.4.

Using the closed form expression for the derivative of the objective function and the fact that the objective function is unimodal in S (and hence quasi-convex), we can apply gradient descent methods to solve the problem to optimality.

5.4. General Model: Multiple Stores and Zones

5.4.1. Order Acceptance Thresholds as Decision Variables. With some abuse of notation,² let S be the decision variable representing the threshold on the number of online orders that can be accepted. In the global threshold class, up to S orders (originating from any customer shipping zone) are accepted. In the local threshold class, up to S_i orders from customer shipping zone i are accepted, and the remaining orders (if any) are canceled in real time.

5.4.2. Objective Function with Costs. Let $\mathcal{A}_i^O(S)$ be a random variable representing the number of online orders accepted from zone i . Let $\mathcal{Q}_j^P = \max\{0, \bar{\mathcal{Q}}_j - \mathcal{P}_j^P\}$ be the random variable that represents the inventory level at store j at the end of the day after physical demands at stores have been satisfied. We can write the objective function in terms of the decision variables and cost components as follows. We use the notation from previous sections.

$$\begin{aligned} & \sum_{i \in [I]} c_i \cdot \mathcal{A}_i^O(S) + \sum_{i \in [I]} \sum_{j \in [J]} \sum_{k \in [\mathcal{Q}_j^P]} [t_{ijk}(\mathcal{Q}_j^P) - c_i] \cdot x_{ijk}(S) \\ & + p \cdot \min \left\{ \sum_{i \in [I]} (\mathcal{Q}_i^O - \mathcal{A}_i^O(S)), \sum_{j \in [J]} \left(\mathcal{Q}_j^P - \sum_{k \in [\mathcal{Q}_j^P]} x_{ijk}(S) \right) \right\} \end{aligned} \quad (11)$$

Observe that the first two terms are identical to the expression of the objective function in the LP (6), with

accepted orders and inventory after physical demand as random variables. The third term represents the lost sales cost. Observe that the expression (11) is nonlinear in $\mathcal{A}_i^O(S)$ and therefore nonlinear in S . We rewrite the objective. We rearrange the expression to get the following expression that is linear in $\mathcal{A}_i^O(S)$.

$$\begin{aligned} & p \cdot \min \left\{ \sum_{i \in [I]} \mathcal{Q}_i^O, \sum_{j \in [J]} \mathcal{Q}_j^P \right\} + \sum_{i \in [I]} c_i \cdot \mathcal{A}_i^O(S) \\ & + \sum_{i \in [I]} \sum_{j \in [J]} \sum_{k \in [\mathcal{Q}_j^P]} [t_{ijk}(\mathcal{Q}_j^P) - c_i - p] \cdot x_{ijk}(S) \end{aligned} \quad (12)$$

The interpretation for expression (12) is via an alternate order of accounting for the costs: first, we incur lost-sales cost for every online demand up to the total inventory in the network after physical demand; Then, we incur a cancellation cost for every accepted order. Finally, we incur expected fulfillment cost minus cancellation cost minus lost-sales cost (as savings) for every order assigned to a store for trial.

5.4.3. Two-Stage Stochastic Program. The joint order acceptance and fulfillment problem can be modeled as a two-stage stochastic program, with the first stage as the order acceptance problem where the random variables are the physical and online demands and the second stage as the fulfillment problem. The first stage can be written as

$$\begin{aligned} \min_S \quad & p \cdot \mathbb{E}_{\mathcal{Q}^O, \mathcal{Q}^P} \left[\min \left\{ \sum_{i \in [I]} \mathcal{Q}_i^O, \sum_{j \in [J]} \mathcal{Q}_j^P \right\} \right] \\ & + \min_S \mathbb{E}_{\mathcal{Q}^O, \mathcal{Q}^P} [G(S)]. \end{aligned} \quad (13)$$

The second stage problem is

$$\begin{aligned} G(S) = & \sum_{i \in [I]} c_i \cdot \mathcal{A}_i^O(S) \\ & + \min_{x_{ijk}(S)} \sum_{i \in [I]} \sum_{j \in [J]} \sum_{k \in [\mathcal{Q}_j^P]} [t_{ijk}(\mathcal{Q}_j^P) - c_i - p] \cdot x_{ijk}(S), \end{aligned} \quad (14)$$

$$\text{s.t.} \quad \sum_{j \in [J]} \sum_{k \in [\mathcal{Q}_j^P]} x_{ijk}(S) \leq \mathcal{A}_i^O(S) \quad \forall i \in [I], \quad (15)$$

$$\sum_{i \in [I]} x_{ijk}(S) \leq 1 \quad \forall j \in [J], k \in [\mathcal{Q}_j^P], \quad (16)$$

$$x_{ijk}(S) \geq 0 \quad \forall i \in [I], j \in [J], k \in [\mathcal{Q}_j^P]. \quad (17)$$

Observe that the linear program (14) is identical to the linear program (6) with the exception that the savings of the lost sales cost for a tried order has been added in this LP. An order accepted from zone i is tried at position k in store j only if the cost of fulfillment is lower than the sum of cancellation and lost-sales cost. The proof of validity

of the linear program (14) is identical to the proof of correctness of the linear program (6).

Lemma 5. For a global threshold policy, $\mathbb{E}[G(S)]$ in the objective function (13) is unimodal in S .

Lemma 6. For a local threshold policy, $\mathbb{E}[G(S_i)]$ in the objective function (13) is unimodal when all other local threshold parameters, $S_{i'}$ for $i' \neq i$, are fixed.

The proof of the first lemma appears in Online Appendix A.5, whereas the second uses a similar argument.

Lemma 7. $\mathbb{E}[G(S)]$ for global threshold and $\mathbb{E}[G(S_i)]$ for local threshold (when all other local threshold parameters, $S_{i'}$ for $i' \neq i$, are fixed) in the objective function (13) are quasi-convex and Lipschitz continuous.

Proof. The quasi-convexity of $\mathbb{E}[G(S)]$ and $\mathbb{E}[G(S_i)]$ for global and local threshold policies follows from their unimodality shown in Lemmas 5 and 6, respectively. The Lipschitz-continuity of these functions is straightforward, as the absolute value of the slope of $\mathbb{E}[G(S)]$ and $\mathbb{E}[G(S_i)]$ cannot exceed the maximum of cost parameters p , c_i , b_j , s_{ij} , and d_i . \square

Because $\mathbb{E}[G(S)]$ and $\mathbb{E}[G(S_i)]$ in the objective (13) for global and local threshold, respectively, are quasi-convex and Lipschitz continuous in S , it follows that the corresponding sample gradients ($\partial G(S)/\partial S$ and $\partial G(S)/\partial S_i$) are unbiased gradient estimates (Glasserman and Ho 1991). Essentially, we have shown that

$$\begin{aligned} \frac{\partial}{\partial S_i} \mathbb{E}_{\mathcal{D}^P, \mathcal{D}^O} [G(S)] &= \mathbb{E}_{\mathcal{D}^P, \mathcal{D}^O} \left[\frac{\partial}{\partial S_i} G(S) \right] \quad \text{and} \\ \frac{\partial}{\partial S} \mathbb{E}_{\mathcal{D}^P, \mathcal{D}^O} [G(S)] &= \mathbb{E}_{\mathcal{D}^P, \mathcal{D}^O} \left[\frac{\partial}{\partial S} G(S) \right] \end{aligned}$$

for local and global threshold policies, respectively. These technical conditions allow us to apply a theorem of Hazan et al. (2015) to prove that the fulfillment problem can be efficiently optimized as in Karp (2017). The theorem proves that the Stochastic Normalized Gradient Descent algorithm will find an ϵ -optimal minimum solution with $\text{poly}(1/\epsilon)$ unbiased gradient estimates and optimization steps. We restate the theorem.

Theorem 4 (Hazan et al. 2015, theorem 5.1). An ϵ -optimal minimum solution $\mathbb{E}[G(S)]$ (for global threshold) and $\mathbb{E}[G(S_i)]$ (for local threshold) can be obtained with $\text{poly}(1/\epsilon)$ unbiased gradient estimates and optimization steps by a Stochastic Normalized Gradient Descent algorithm.

We have so far established that the general single-stage omnichannel fulfillment problem can be solved efficiently using unbiased gradient estimates. In Section 5.5, we describe the Infinitesimal Perturbation Analysis (IPA) method to computationally solve our two-stochastic programming problem and derive expressions for sample gradients.

5.5. IPA

IPA is a sample-path optimization technique that relies on sample gradients to estimate the gradient of the optimization problem at hand.

5.5.1. IPA Algorithm. We describe the IPA algorithm for the global threshold class of policies. The procedure starts with an arbitrary value for the threshold S . A simulated instance of the online demand (D_i^O) for customer shipping zone i and physical demand (D_j^P) at store j is generated. Using the realized demands, Q_j^P and A_i^O are determined. The second-stage problem is solved in a deterministic fashion to compute the optimal solution using Q_j^P and A_i^O as parameters. The gradient of the objective function (derivatives with respect to the threshold S) is estimated and accumulated over regenerative cycles; the average gradient value is then used to update the values of S . The procedure is summarized in a pseudo-code format, where N denotes the number of steps taken in a path search, U represents the number of regenerative cycles, γ_n represents the step size at iteration n , and $S^{(n)}$ represents the threshold for n^{th} iteration.

Algorithm 2. (IPA)

- 1: Initialize N, U
- 2: Initialize $S^{(1)}$ possibly based on demand distribution
- 3: **for** $n \in \{1, \dots, N\}$ **do**
- 4: **for** $u \in \{1, \dots, U\}$ **do**
- 5: i. Generate an instance of the online and physical demands $D_i^O \forall i, D_j^P \forall j$.
- 6: ii. $Q_j^P \leftarrow \min\{\bar{Q} - \min\{\bar{Q}, D_j^P\}\} \forall j$
- 7: iii. Obtain A_i^O from D_i^O and $S^{(n)} \forall i$.
- 8: iv. Solve the linear program (14) to obtain the optimal dual values of Constraints (15).
- 9: v. Obtain the (unbiased) gradient estimates δ_{nu} from optimal dual values.
- 10: Calculate the desired gradient(s), $\delta_n \leftarrow (1/U) \cdot \sum_{u=1}^U \delta_{nu}$
- 11: Update $S^{(n+1)} \leftarrow S^{(n)} - \gamma_n \cdot \delta_n$

Obtaining A_i^O from D_i^O . We now describe the procedure to obtain sample derivatives from the optimal dual values of Constraints (7) after solving the linear program (14) for the three classes of order acceptance policies.

- Local threshold:

$$A_i^O \leftarrow \min\{D_i^O, S_i^{(n)}\}.$$

- Global threshold: Assuming Poisson distributions for online demands such that $\mathcal{D}_i^O \sim \text{Poisson}(\lambda_i)$,

$$A_i^O \leftarrow \begin{cases} D_i^O & \text{if } \sum_{i \in [I]} D_i^O < S^{(n)} \\ \left(\lambda_i / \sum_{i \in [I]} \lambda_i \right) \cdot S^{(n)} & \text{otherwise} \end{cases}$$

5.5.2. Sample Gradient Estimates. We derive the expressions for sample derivative estimates.

Lemma 8 (Sample gradient for Local Threshold Class). Let $\alpha_i^*(S)$ be the optimal dual variable corresponding to constraint i of the set of Constraints (15) of the linear program (6).

$$\frac{\partial}{\partial S_i} G(S) = (c_i + \alpha_i^*(S)) \cdot \mathbb{1}(D_i^O \geq S_i)$$

Proof. Let us consider constraint i of the set of Constraints (15) of the linear program (14). We know from linear programming theory that $c_i + \alpha_i^*(S_i)$ at optimality represents the rate of change of objective function with respect to $A_i^O(S_i)$.

If $D_i^O > S_i$, then $A_i^O(S_i) = S_i$. On the other hand, if $D_i^O < S_i$, then $A_i^O(S_i) = D_i^O$.

$$\frac{\partial}{\partial S_i} G(S) = \frac{\partial G(S)}{\partial A_i^O(S_i)} \cdot \frac{\partial A_i^O(S_i)}{\partial S_i} = (c_i + \alpha_i^*(S)) \cdot \mathbb{1}(D_i^O > S_i) \quad \square$$

Lemma 9 (Sample Gradient for Global Threshold Class). Let us consider Poisson distributions for online demands such that $\mathcal{D}_i^O \sim \text{Poisson}(\lambda_i)$. Let $\alpha_i^*(S)$ be the optimal dual variable corresponding to constraint i of the set of Constraints (15) of the linear program (14).

$$\frac{\partial}{\partial S} G(S) = \sum_{i \in [I]} \left(\frac{\lambda_i}{\sum_{i \in [I]} \lambda_i} \right) \cdot (c_i + \alpha_i^*(S)) \cdot \mathbb{1} \left(\sum_{i \in [I]} D_i^O \geq S \right)$$

Proof. We know from linear programming theory that $c_i + \alpha_i^*(S)$ represents the rate of change of $G(S)$ with respect to $A_i^O(S)$.

The probability that the first rejected order is from customer shipping zone i is $\lambda_i / (\sum_{i \in [I]} \lambda_i)$ (a property of the minimum of independent exponential random variables). If $\sum_{i \in [I]} D_i^O < S$, then $A_i^O(S) = D_i^O$. On the other hand, when $\sum_{i \in [I]} D_i^O \geq S$, then $\sum_{i \in [I]} A_i^O(S) = S$.

$$\begin{aligned} \frac{\partial}{\partial S} G(S) &= \sum_{i \in [I]} \Pr[\text{First rejected order from } i] \\ &\quad \cdot \frac{\partial G(S)}{\partial A_i^O(S)} \cdot \frac{\partial A_i^O(S)}{\partial S} \\ &= \sum_{i \in [I]} \left(\frac{\lambda_i}{\sum_{i \in [I]} \lambda_i} \right) \cdot (c_i + \alpha_i^*(S)) \cdot \mathbb{1} \left(\sum_{i \in [I]} D_i^O \geq S \right) \quad \square \end{aligned}$$

Theorem 5. From theorem 5.1 of Hazan et al. (2015), using

$$\begin{aligned} &\frac{c_i + \alpha_i^*(S)}{|c_i + \alpha_i^*(S)|} \cdot \mathbb{1}(D_i^O > S_i) \quad \text{and} \quad \sum_{i \in [I]} \left(\frac{\lambda_i}{\sum_{i \in [I]} \lambda_i} \right) \cdot \frac{c_i + \alpha_i^*(S)}{|c_i + \alpha_i^*(S)|} \\ &\quad \cdot \mathbb{1} \left(\sum_{i \in [I]} D_i^O \geq S \right). \end{aligned}$$

as the normalized gradients for the local and global threshold policies, respectively, our IPA-based algorithm, Algorithm 2,

obtains an ϵ -optimal threshold policy respectively with $\text{poly}(1/\epsilon)$ total samples of linear program (14).

6. Computational Experiments

We now turn to evaluate our policies on simulated instances. To this end, we use real-world data to characterize all the parameters in our generative model. This is of interest not only from a purely descriptive perspective but also to create a reasonably realistic setting to test improvements of our methods over simple alternates that retailers have been traditionally using in practice.

6.1. Data Description

We perform this analysis by examining four retailers who are clients of the provider. The provider receives inventory feeds and online order feeds (unfortunately without zip codes), which we leverage here to do our experimental analysis and evaluation. The four retailers (named A–D) in our study are representative of high-end retail stores in the United States (see Table 2).

6.2. Extracting Parameters from Data and Sampling Orders

We use the data from all of the above retailers to demonstrate the broad applicability of the problem and results. In particular, we use the data to study the characteristics of the following parameters of our fulfillment models: (i) distribution of number of stores for an SKU; (ii) distribution of shipping costs; (iii) distribution of labor costs; (iv) inventory distribution and pick-failure probability. We use the sampled distributions to generate synthetic single-SKU orders with all the required cost parameters. The details of the extraction of the parameters and sampling orders are in Online Appendix B.

6.3. Value of Modeling Pick Failure

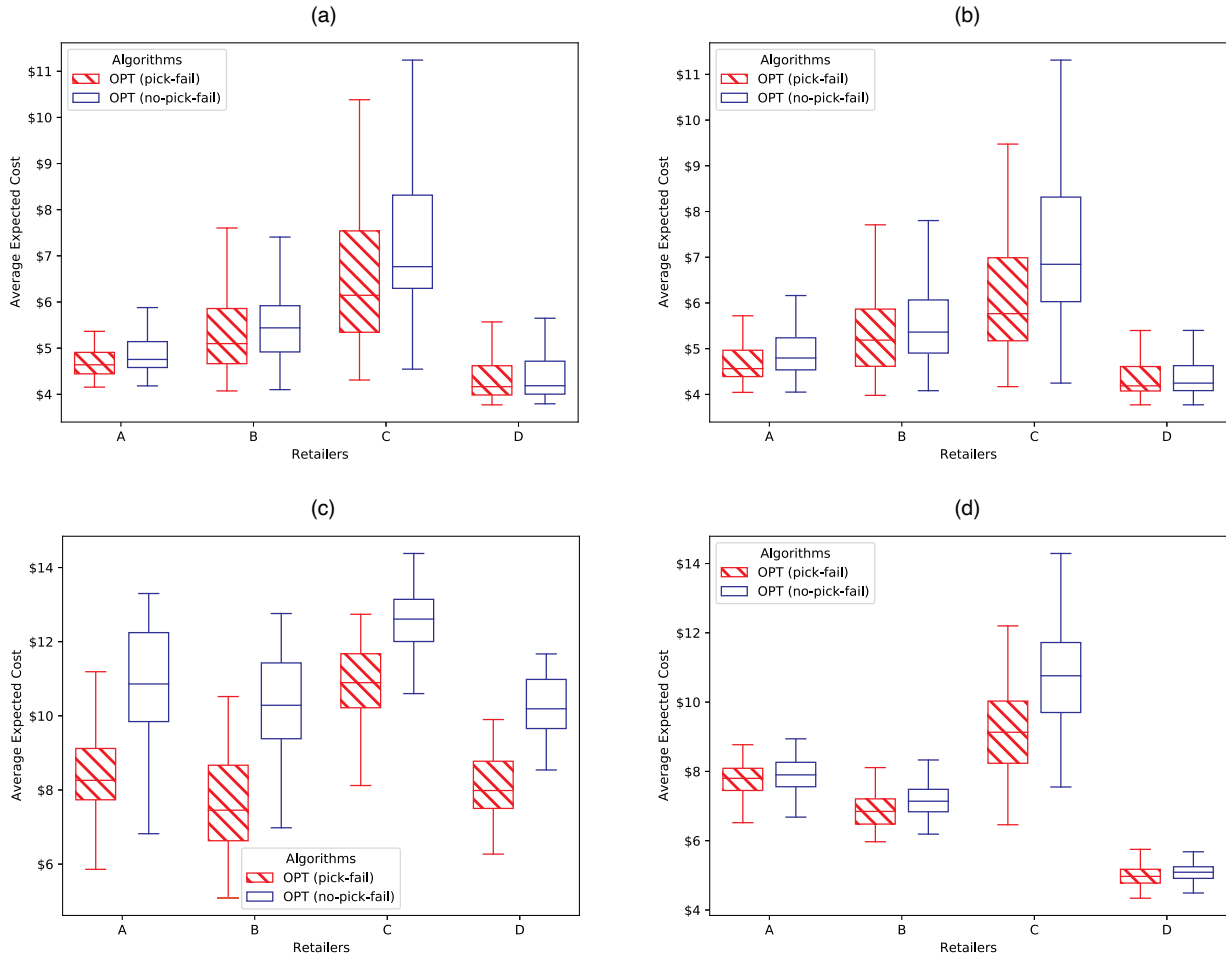
In our computational experiments, we demonstrate the value of modeling pick failure. For each of our models corresponding to Sections 2, 3, 4, and 5, we compare the performance of our optimal algorithms, which take into account pick failure at stores with the optimal algorithm that does not take pick failure into account (Figure 5). We conducted tests on 100 instances for each retailer to obtain these results.

1. *Single order: static pick-failure probability (Section 2):* We compare our optimal dynamic programming algorithm to the greedy algorithm used in practice in which

Table 2. Overview of Retail Data Used in This Analysis

ID	Num SKUs	Num stores	Description
A	399,343	1,154	Multibillion department
B	503,875	830	Multibillion sports goods
C	104,945	204	Multibillion clothing
D	73,370	600	Specialty sports goods

Note. Num, number.

Figure 5. (Color online) Value of Modeling Pick Failure

Notes. (a) Single-order fulfillment under static pick-failure probabilities (Section 2); (b) single-order fulfillment under dynamic pick-failure probabilities (Section 3); (c) multiorder fulfillment (Section 4); (d) multiorder joint acceptance and fulfillment: global thresholding policy (Section 5). Average savings (a) A: 3.44%; B: 3.45%; C: 9.12%; D: 0.66%; (b) A: 4.61%; B: 3.5%; C: 11.81%; D: 0.55%; (c) A: 21.91%, B: 24.9%, C: 13.21%, D: 20.67%; (d) A: 1.61%, B: 4.01%, C: 14.35%, D: 1.96%.

the L stores with the lowest sum of labor and shipping costs ($b_j + s_j$) are chosen in order of total (Figure 5(a)). The greedy algorithm is optimal when we do not account for pick failure at stores. We set the maximum number of stages to explore to three before canceling with a delayed cancellation cost (d) of \$25, which is higher than all shipping and labor costs used.

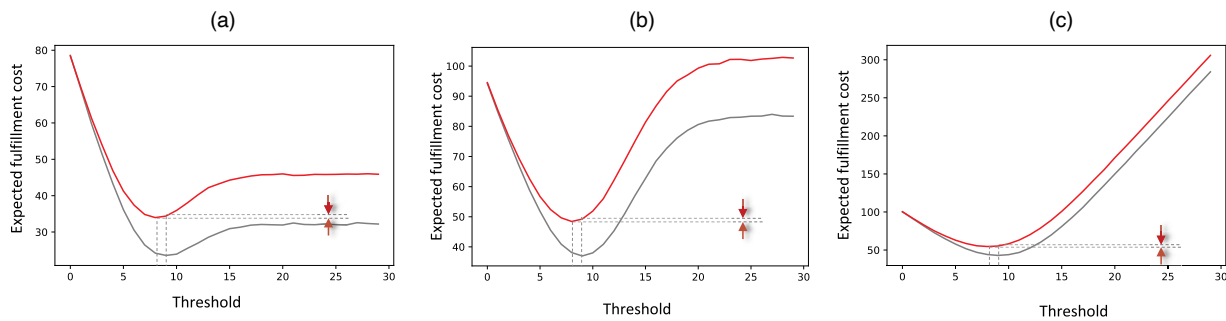
2. *Single order: dynamic pick-failure probability (Section 3):* Similar to the single order with static pick-failure probability, we compare our optimal dynamic programming (Algorithm 1) to the same greedy algorithm, which does not account for pick failure at the stores (Figure 5(b)). We used the same setup of three stages and $d = \$25$.

3. *Multiorders (Section 4):* In Figure 5(c), we computationally observe the costs and savings associated with modeling pick failure when the orders for the same item originating from different shipping zones are to be fulfilled in stores. Without accounting for pick failure, the optimal algorithm for the multiorder problem is to solve a deterministic transportation problem with

(i) accepted orders from the shipping zones as the demands and (ii) inventory at the stores after physical demands are satisfied as supplies. The cost of fulfillment of the policy returned by this transportation problem is then evaluated by calculating the expected fulfillment cost in the presence of pick failure using a simple one-step approximation (See Online Appendix B). Note that we use a single stage of fulfillment in this setup. The average expected costs in Figure 5(c) are the per order costs obtained by dividing the total costs by the number of orders.

4. *Joint order acceptance & fulfillment (Section 4):* We compare our IPA-based joint optimal order acceptance and fulfillment policy with the policy that is optimal for fulfillment in the absence of pick failure (as in Karp 2017) in Figure 5(d). We test our algorithm for the global thresholding class of policies. The average expected costs in Figure 5(d) are the per order costs obtained by dividing the total costs by the mean number of orders.

Figure 6. (Color online) Value of Modeling Pick Failure for the Joint Order Acceptance and Fulfillment Problem for One Store



Notes. (a) $\mathcal{D}^O = \text{Poisson}(10)$, (b) $\mathcal{D}^O = \text{Poisson}(15)$, (c) $\mathcal{D}^O = \text{Poisson}(50)$. $p = \$10$, $c = \$15$, $b = 0$, $s = 0$, $d = \$15$, $\bar{Q} = 30$, $\mathcal{D}^P \sim \text{Poisson}(20)$. Here $S_{\text{no pick failure}}^* = \bar{Q} - F_p^{-1}\left(\frac{c}{p+c}\right) = 9$, whereas $S_{\text{with pick failure}}^* = 8$. The differences pointed out are the expected cost savings from considering pick failures.

6.4. Value of Modeling Pick Failure in a Single-Store Joint Order Acceptance and Fulfillment Problem (Section 5.3)

In Figure 6, we demonstrate the value of modeling pick failure for the joint order acceptance and fulfillment problem in a stylized model with a single store (Section 5.3). In Figure 6, we highlight the variation of optimal fulfillment costs under various order acceptance thresholds. Whereas the upper line represents the total optimal fulfillment costs as a function of threshold, the lower line represents the fulfillment cost of the optimal fulfillment policy as a function of order acceptance threshold when pick failure is not modeled (as in Karp 2017). As can be seen in the figure, the optimal threshold with pick failure is eight. If pick failure is not considered (as in Karp 2017), the optimal threshold is nine. The savings is therefore the difference in optimal expected fulfillment cost (upper line) at thresholds eight and nine. Also observe that the optimal thresholds for both models are independent of the online demand.

We have conducted additional computational experiments to investigate the sensitivity of savings to the other parameters of our models; the results are in Online Appendix C.

7. Conclusion and Future Work

In this paper, we incorporated picking costs in the problem of optimizing omnichannel fulfillment in SFS schemes under pick failure. We modeled this problem for single-SKU orders under various settings of physical and online demands being sparse or dense. We modeled the fulfillment problem as a network flow with convex costs and used this in deriving effective policies for online order acceptance incorporating picking costs. Using data from North American retailers to design experiments, we demonstrated that our algorithms achieved cost savings of up to 22% by incorporating pick failures. Our study demonstrates that the modeling of the stochastic nature of pick failures along with their interaction with picking and shipping costs is an important component in optimizing SFS fulfillment costs for large retailers.

There are several natural avenues for extending our work, such as extending the model of Section 4 to more than two stages of fulfillment and the extension of all our models to multi-item orders with synergies in fulfillment costs (such as shipping cost savings from consolidation). A more immediate extension of our models to the case when an additional dedicated fulfillment center for online orders is added to the possibilities is also interesting and may highlight interesting trade-offs that are pertinent to the upstream problem of consolidating and planning new fulfillment centers.

Endnotes

¹ Even without this assumption, the problem can be modelled as a two-stage stochastic program and solved using an IPA-based algorithm as described in Section 5.4

² S is (i) a scalar for global threshold policies and (ii) an I -sized vector for local threshold policies

References

- Acimovic J, Farias VF (2019) The fulfillment-optimization problem. Netessine S, ed. *Operations Research & Management Science in the Age of Analytics*, INFORMS Tutorials in Operations Research (INFORMS, Catonsville, MD), 218–237.
- Acimovic J, Graves SC (2014) Making better fulfillment decisions on the fly in an online retail environment. *Manufacturing Service Oper. Management* 17(1):34–51.
- Andrews JM, Farias VF, Khojandi AI, Yan CM (2019) Primal–dual algorithms for order fulfillment at Urban Outfitters, Inc. *Interfaces* 49(5):355–370.
- Caro F, Sadr R (2019) The internet of things (IoT) in retail: Bridging supply and demand. *Bus. Horizons* 62(1):47–54.
- CNBC (2021) Walmart bets bigger on online grocery as it ramps up automated fulfillment at stores. Accessed November 1, 2022, <https://www.cnbc.com/2021/01/27/walmart-to-ramp-up-automated-fulfillment-at-its-stores-as-online-grocery-grows.html>.
- DeHoratius N, Raman A (2008) Inventory record inaccuracy: An empirical analysis. *Management Sci.* 54(4):627–641.
- Glasserman P, Ho Y (1991) *Gradient Estimation Via Perturbation Analysis* (Springer, Dordrecht, Netherlands).
- Harsha P, Subramanian S, Uichanco J (2019) Dynamic pricing of omnichannel inventories: Honorable mention—2017 M&SOM practice-based research competition. *Manufacturing Service Oper. Management* 21(1):47–65.

- Hazan E, Levy K, Shalev-Shwartz S (2015) Beyond convexity: Stochastic quasi-convex optimization. Cortes C, Lee DD, Sugiyama M, Garnett R, eds. *Proc. 28th Internat. Conf. Neural Inform. Processing Systems*, vol. 1 (MIT Press, Cambridge, MA), 1594–1602
- Holloway BB, Beatty SE (2003) Service failure in online retailing: A recovery opportunity. *J. Service Res.* 6(1):92–105.
- Jasin S, Sinha A (2015) An LP-based correlated rounding scheme for multi-item ecommerce order fulfillment. *Oper. Res.* 63(6):1336–1351.
- Jia S, Karp J, Ravi R, Tayur S (2022) Effective online order acceptance policies for omnichannel fulfillment. *Manufacturing Service Oper. Management* 24(3):1650–1663
- Karp J (2017) Models and methods for omni-channel fulfillment. Unpublished PhD thesis, Carnegie Mellon University, Pittsburgh.
- Lei Y, Jasin S, Sinha A (2018a) Joint dynamic pricing and order fulfillment for e-commerce retailers. *Manufacturing Service Oper. Management* 20(2):269–284.
- Lei YM, Jasin S, Uichanco J, Vakhutinsky A (2018b) Joint product framing (display, ranking, pricing) and order fulfillment under the MNL model for e-commerce retailers. Preprint, submitted November 9, https://papers.ssrn.com/sol3/papers.cfm?abstract_id=3282019.
- Markowicz F (2017) Optimizing order-routing decisions: Leveraging omni-channel supply chain fulfillment. Unpublished PhD thesis, Massachusetts Institute of Technology, Cambridge, MA.
- Masae M, Glock CH, Grosse EH (2019) Order picker routing in warehouses: A systematic literature review. *Internat. J. Production Econom.* 224:107564.
- Shabani A, Maroti G, de Leeuw S, Dullaert W (2021) Inventory record inaccuracy and store-level performance. *Internat. J. Production Econom.* 235:108111.
- Sheffi Y (2016) Guest voices: In-store fulfillment is no defense against Amazon. *Wall Street Journal* (July 27), <https://www.wsj.com/articles/guest-voices-in-store-fulfillment-is-no-defense-against-amazon-1469635863>.
- Skorupa J (2015) Revealing causes of omnichannel failure. Accessed November 1, 2022, <https://risnews.com/revealing-causes-omnichannel-failure>.
- Smith J (2020) Smaller is big in new e-commerce warehouses. *Wall Street Journal* (November 8), <https://www.wsj.com/articles/smaller-is-big-in-new-e-commerce-warehouses-11604840400>.
- Tardos É (1985) A strongly polynomial minimum cost circulation algorithm. *Combinatorica* 5(3):247–255.
- U.S. Department of Commerce (2021) Quarterly retail ecommerce sale. Accessed November 1, 2022, https://www.census.gov/retail/mrts/www/data/pdf/ec_current.pdf.
- Xu PJ, Allgor R, Graves SC (2009) Benefits of reevaluating real-time order fulfillment decisions. *Manufacturing Service Oper. Management* 11(2):340–355.

Appendix A: Proofs

A.1. Proof of Lemma 1

Proof. The proof relies on the observation that swapping adjacent stores when the earlier store has a higher unreliability factor results in a lowering of the expected cost.

Assume for the sake of a contradiction that $r_m = r_{\sigma^*(l)} > r_{\sigma^*(l+1)} = r_n$.

$$\begin{aligned} r_m > r_n &\implies \frac{r_m}{1-\phi_n} - \frac{\phi_n \cdot r_m}{1-\phi_n} > \frac{r_n}{1-\phi_m} - \frac{\phi_m \cdot r_n}{1-\phi_m} \\ &\implies (1-\phi_m) \cdot r_m + \phi_m \cdot (1-\phi_n) \cdot r_n > (1-\phi_n) \cdot r_n + \phi_n \cdot (1-\phi_m) \cdot r_m \end{aligned} \quad (18)$$

Expanding the cost of the optimal policy, we get

$$\begin{aligned} \mathcal{V}(\sigma^*) &= \left[\sum_{l=1}^{q-1} \left(\prod_{l'=1}^{l-1} \phi_{\sigma^*(l')} \right) \cdot (1-\phi_{\sigma^*(l)}) \cdot r_{\sigma^*(l)} \right] \\ &\quad + \left(\prod_{l=1}^{q-1} \phi_{\sigma^*(l)} \right) \cdot (1-\phi_m) \cdot r_m + \left(\prod_{l=1}^q \phi_{\sigma^*(l)} \right) \cdot (1-\phi_n) \cdot r_n \\ &\quad + \left[\sum_{l=q+2}^L \left(\prod_{l'=1}^{l-1} \phi_{\sigma^*(l')} \right) \cdot (1-\phi_{\sigma^*(l)}) \cdot r_{\sigma^*(l)} \right] + \left[\left(\prod_{l=1}^L \phi_{\sigma^*(l)} \right) \cdot d \right] \end{aligned} \quad (19)$$

Examining the middle two terms corresponding to the l^{th} and $(l+1)^{st}$ stores we get

$$\begin{aligned} (19) &= \left(\prod_{l=1}^{q-1} \phi_{\sigma^*(l)} \right) \cdot \left[(1-\phi_m) \cdot r_m + \phi_m \cdot (1-\phi_n) \cdot r_n \right] \\ &> \left(\prod_{l=1}^{q-1} \phi_{\sigma^*(l)} \right) \cdot \left[(1-\phi_n) \cdot r_n + \phi_n \cdot (1-\phi_m) \cdot r_m \right] \quad \text{using (18)} \end{aligned} \quad (20)$$

Interchanging the trial positions of stores m and n would yield a policy with cost term (19) in $\mathcal{V}(\sigma^*)$ replaced by (20). The cost of this policy is lower than $\mathcal{V}(\sigma^*)$, giving the desired contradiction. \square

A.2. Proof of Lemma 3

Table 3 Notations

\mathcal{Q}_{jk}^O	: inventory level for the k^{th} online order to be tried at store j .
$t'_{ij}(q)$: expected trial cost at store j with inventory level q
$t'_{ij}(q) = b_j + (1-\phi_j(q)) \cdot s_{ij} + \phi_j(q) \cdot d_i$	

Recall

$$t_{ijk}(\mathcal{Q}_j^P) = \sum_{q=\mathcal{Q}_j^P-(k-1)}^{\mathcal{Q}_j^P} \mathbf{Pr}[\mathcal{Q}_{jk}^O = q] \cdot t'_{ij}(q)$$

1. We observe from the definition of $t_{ijk}(\mathcal{Q}_j^P)$ that it is a convex combination of expected fulfillment costs at different feasible inventory levels.

2. The expected fulfillment cost at inventory level q , $t'_{ij}(q)$ is the sum of a constant b_j and convex combination of s_{ij} and d_i . Since $s_{ij} < d_i$ and $\phi_j(q)$ is a non-increasing function (assumption 1), it follows that the expected trial cost $t'_{ij}(q)$ is non-increasing in q .
3. $\Pr[\mathcal{Q}_{jk}^O = q]$ is the probability that the leftover inventory is q after the trial of $k-1$ online orders. The $k-1$ trials can be decomposed into two events (i) successful attempts of $\mathcal{Q}_j^P - q$ online orders one each at inventory level $\{q+1, \dots, \mathcal{Q}_j^P\}$ and (ii) failed attempts of the remaining $(k-1) - (\mathcal{Q}_j^P - q)$ online orders in the set of inventory levels $\{q, \dots, \mathcal{Q}_j^P\}$. The probability of event (ii) decreases with k because when more items are independently tried at same pick failure probability levels, more items are likely to fail.

$$\Pr[\mathcal{Q}_{jk}^O = q] = \underbrace{\left(\prod_{q'=q+1}^{\mathcal{Q}_j^P} [1 - \phi_j(q')] \right)}_{\text{(i):independent of } k} \cdot \underbrace{\Pr[(k-1) - (\mathcal{Q}_j^P - q) \text{ failures}]}_{\text{(ii):non-increasing in } k} \quad (21)$$

Therefore, $\Pr[\mathcal{Q}_{jk}^O = q]$ is non-increasing in k .

4. We are now ready to compare $t_{ijk}(\mathcal{Q}_j^P)$ and $t_{ij(k-1)}(\mathcal{Q}_j^P)$.

$$\begin{aligned} t_{ijk}(\mathcal{Q}_j^P) &= \sum_{q=\mathcal{Q}_j^P-(k-1)}^{\mathcal{Q}_j^P} \Pr[\mathcal{Q}_{jk}^O = q] \cdot t'_{ij}(q) \\ &= \sum_{q=\mathcal{Q}_j^P-(k-2)}^{\mathcal{Q}_j^P} \Pr[\mathcal{Q}_{jk}^O = q] \cdot t'_{ij}(q) + \Pr[\mathcal{Q}_{jk}^O = \mathcal{Q}_j^P - (k-1)] \cdot t'_{ij}(\mathcal{Q}_j^P - (k-1)) \\ &\stackrel{\text{Step 2}}{\geq} \sum_{q=\mathcal{Q}_j^P-(k-2)}^{\mathcal{Q}_j^P} \left(\Pr[\mathcal{Q}_{jk}^O = q] + \underbrace{\left[\Pr[\mathcal{Q}_{j(k-1)}^O = q] - \Pr[\mathcal{Q}_{jk}^O = q] \right]}_{\geq 0 \text{ (Step 3)}} \right) \cdot t'_{ij}(q) \\ &\quad + \underbrace{\left(\Pr[\mathcal{Q}_{jk}^O = \mathcal{Q}_j^P - (k-1)] + \sum_{q=\mathcal{Q}_j^P-(k-2)}^{\mathcal{Q}_j^P} \Pr[\mathcal{Q}_{jk}^O = q] \right)}_{=1 \text{ (Step 1)}} \cdot t'_{ij}(\mathcal{Q}_j^P - (k-1)) \\ &\quad - \underbrace{\left(\sum_{q=\mathcal{Q}_j^P-(k-2)}^{\mathcal{Q}_j^P} \Pr[\mathcal{Q}_{j(k-1)}^O = q] \right)}_{=1 \text{ (Step 1)}} \cdot t'_{ij}(\mathcal{Q}_j^P - (k-1)) \\ &= \sum_{q=\mathcal{Q}_j^P-(k-2)}^{\mathcal{Q}_j^P} \Pr[\mathcal{Q}_{j(k-1)}^O = q] \cdot t'_{ij}(q) \\ &= t_{ij(k-1)}(\mathcal{Q}_j^P) \end{aligned}$$

□

A.3. Proof of Lemma 4

We have the following objective function to be minimized.

$$G(S) = \mathbb{E}_{\mathcal{D}^O, \mathcal{D}^P} \left[\sum_{k=1}^{\min\{\mathcal{D}^O, S, \mathcal{Q}^P\}} t_k(\mathcal{Q}^P) + c \cdot [\min\{\mathcal{D}^O, S\} - \min\{\mathcal{D}^O, S, \mathcal{Q}^P\}] \right] \\ + \mathbb{E}_{\mathcal{D}^O, \mathcal{D}^P} [p \cdot \min\{\mathcal{D}^O - \min\{\mathcal{D}^O, S\}, \mathcal{Q}^P - \min\{\mathcal{D}^O, S, \mathcal{Q}^P\}\}]$$

Consider the derivative of the objective function. Recall that \bar{Q} is the initial store inventory and $\mathcal{Q}^P = \max(0, \bar{Q} - \mathcal{D}^P)$.

$$G'(S) = \Pr[\mathcal{D}^O \geq S] \Pr[\mathcal{D}^P \leq \bar{Q} - S] \cdot \mathbb{E}_{\mathcal{D}^P}[t_S(\mathcal{Q}^P)] \\ + c \cdot \Pr[\mathcal{D}^O \geq S] \cdot (1 - \Pr[\mathcal{D}^P \leq \bar{Q} - S]) \\ - p \cdot \Pr[\mathcal{D}^O \geq S] \cdot \Pr[\mathcal{D}^P \leq \bar{Q} - S] \\ = \Pr[\mathcal{D}^O \geq S] \cdot F(\bar{Q} - S) \cdot \mathbb{E}_{\mathcal{D}^P}[t_S(\mathcal{Q}^P)] \\ + c \cdot \Pr[\mathcal{D}^O \geq S] \cdot [1 - F(\bar{Q} - S)] - p \cdot \Pr[\mathcal{D}^O \geq S] \cdot F(\bar{Q} - S) \\ = \Pr[\mathcal{D}^O \geq S] \cdot \left[(\mathbb{E}_{\mathcal{D}^P}[t_S(\mathcal{Q}^P)] - p - c) \cdot F(\bar{Q} - S) + c \right]$$

Here F is the cdf of \mathcal{D}^P . Note that the first derivative, $G'(S)$ is independent of the distribution of online orders.

Let us consider S^* such that $G'(S^*) = 0$. Let $\delta > 0$.

$$G'(S^* + \delta) = \Pr[\mathcal{D}^O \geq S^* + \delta] \cdot \left([\mathbb{E}_{\mathcal{D}^P}[t_{S^*+\delta}(\mathcal{Q}^P)]] - p - c \right) \cdot F(\bar{Q} - S^* - \delta) + c \\ \geq \Pr[\mathcal{D}^O \geq S^* + \delta] \cdot \left([\mathbb{E}_{\mathcal{D}^P}[t_{S^*}(\mathcal{Q}^P)]] - p - c \right) \cdot F(\bar{Q} - S^*) + c \quad (22) \\ = 0 \quad G(S^*) = 0$$

$$G'(S^* - \delta) = \Pr[\mathcal{D}^O \geq S^* - \delta] \cdot \left([\mathbb{E}_{\mathcal{D}^P}[t_{S^*-\delta}(\mathcal{Q}^P)]] - p - c \right) \cdot F(\bar{Q} - S^* + \delta) + c \\ \leq \Pr[\mathcal{D}^O \geq S^* - \delta] \cdot \left([\mathbb{E}_{\mathcal{D}^P}[t_{S^*}(\mathcal{Q}^P)]] - p - c \right) \cdot F(\bar{Q} - S^*) + c \quad (23) \\ = 0 \quad G(S^*) = 0$$

(22) and (23) rely on $F(\cdot)$ being a cdf and t_k being an increasing function in k (Lemma 3). Since, $G'(S)$ is decreasing when $S < S^*$ and increasing when $S > S^*$, S^* is a global minimum. \square

A.4. Proof of Corollary 1

Proof. Let $\tilde{S} = \bar{Q} - F^{-1}\left(\frac{c}{p+c}\right)$ be the optimal threshold for the one store one zone problem without pick failure, as shown in Karp (2017). Let us further assume that $b = 0$ and $s = 0$, so that we are in the same setup as Karp (2017), though the corollary holds for any positive values of b and s .

$$\begin{aligned}
 G'(\tilde{S}) &= \mathbf{Pr} \left[\mathcal{D}^O \geq \tilde{S} \right] \cdot \left(\left[\mathbb{E} [t_{\tilde{S}}(\mathcal{Q}^P)] - p - c \right] \cdot F(\bar{Q} - \tilde{S}) + c \right) \\
 &= \mathbf{Pr} \left[\mathcal{D}^O \geq \tilde{S} \right] \cdot \left(\left[\mathbb{E} [t_{\tilde{S}}(\mathcal{Q}^P)] - p - c \right] \cdot \frac{c}{p+c} + c \right) && \text{definition of } \tilde{S} \\
 &= \mathbf{Pr} \left[\mathcal{D}^O \geq \tilde{S} \right] \cdot \left(\mathbb{E} [t_{\tilde{S}}(\mathcal{Q}^P)] \cdot \frac{c}{p+c} \right) \\
 &> 0
 \end{aligned}$$

Since, $G'(S)$ is decreasing when $S < S^*$ and increasing when $S > S^*$, as shown in Lemma 4, it follows that $S^* < \tilde{S}$. □

Note that our single store problem reduces to the single store problem in Karp (2017) if b (the picking trial cost), s (the shipping cost) and $\phi(q)$ (the pick failure probability) are set to 0.

A.5. Proof of Theorem 5

Proof adapted from the proof of (Karp 2017, Theorem 2). We know from Theorem 3 that the linear program (14) can be solved using a min-cost circulation network. We first state a lemma about the supermodularity of objective value of min cost flow as a function of the supplies. Using this lemma, we proceed to prove unimodality of the objective function.

LEMMA 10 (Karp (2017), Lemma 2). *In a minimum-cost single-commodity flow problem with multiple sources, one sink, and integer supplies, demands, and capacities, the objective value of a minimum cost feasible flow as a function of the supplies at the source nodes is supermodular.*

Let S^* be an optimal solution. We will consider two cases of threshold S , (i) $S < S^*$ and (ii) $S > S^*$. For each of the cases, let us consider realizations of demands $\mathcal{D}_i^O = D_i^O$ and $\mathcal{D}_j^P = D_j^P$. Let the online demand realizations D_i^O be such that $\sum_{i \in I} D_i^O \geq S$, because for other realizations, the objective value $G(S)$ does not change with respect to S . Therefore the total number of orders accepted $\sum_{i \in [I]} A_i^O(S)$ is S .

The min-cost flow problem has multiple sources with supplies. Let $V(S) = [A_1^O(S), \dots, A_I^O(S)]$ be the supplies at zones with realized demands D_i^O, D_j^P at threshold S . We say $V(S_1) \subseteq V(S_2)$ if $A_i^O(S_1) \leq A_i^O(S_2)$ for $i \in [I]$.

Case 1: $[S \geq S^]$* We observe that $V(S^*) \subseteq V(S^* + 1) \subseteq V(S) \subseteq V(S + 1)$. Now by Lemma 10 (supermodularity of $G(S)$), $G(S + 1) - G(S) \geq G(S^* + 1) - G(S^*)$. This implies that $G(S + 1) - G(S)$ is non-decreasing for $S > S^*$

Case 2: $[S < S^]$* We observe that $V(S - 1) \subseteq V(S) \subseteq V(S^* - 1) \subseteq V(S^*)$. Now by the Lemma 10 (supermodularity of $G(S)$), $G(S) - G(S - 1) \leq G(S^*) - G(S^* - 1)$. This implies that $G(S + 1) - G(S)$ is non-increasing for $S < S^*$

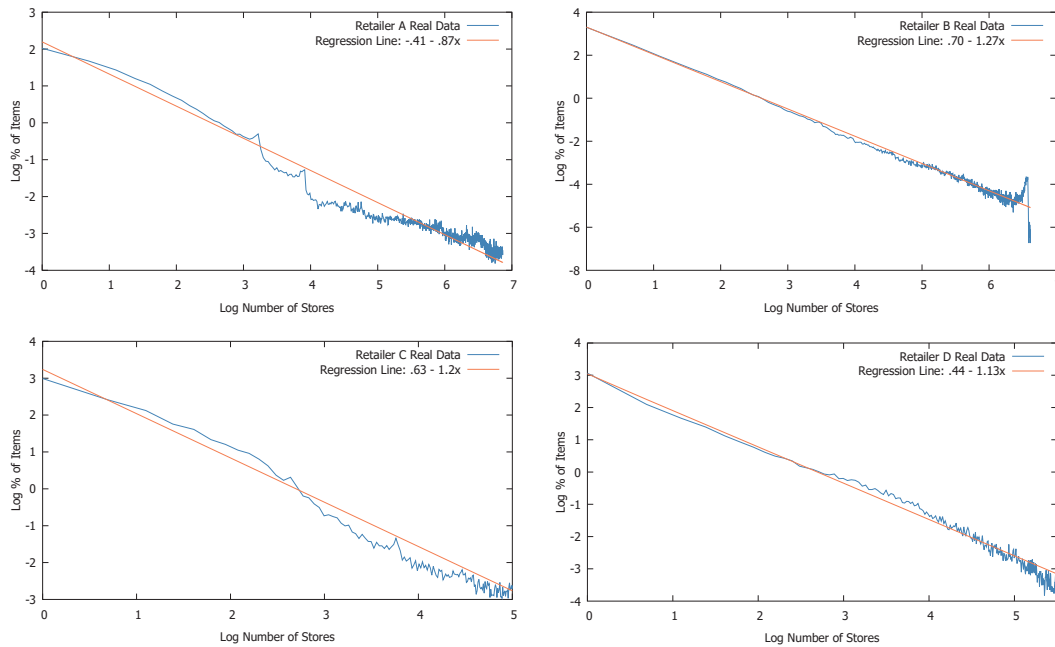
Since, $G(S + 1) - G(S)$ is non-increasing when $S < S^*$ and non-decreasing when $S > S^*$, S^* is a global minimum. □

Appendix B: Extracting model parameters from data and sampling orders

We discuss in detail the process of extraction of the model parameters from the data provided by our partner solutions provider. After the estimation of the distribution of model parameters from data, we describe the sampling of an order using the estimated parameter distributions.

1. *Distribution of number of stores for an SKU.* We consider the distribution of store-counts of SKUs, i.e., the number of stores that carry each SKU and made available online for SFS. To do this, we examined the inventory counts of the entire store network for every single SKU of the five retailers. For each SKU, we counted the number of locations where it is available to be fulfilled from and aggregated these counts in a log – log plot in Figure 7.

Figure 7 Log-log plot of the distribution of the number of stores in which a typical SKU is available for fulfillment segregated by retailer



The clear straight line plots between the log of percentage of SKUs and the log of number of stores as shown in Figure 7 demonstrate a power law degree distribution for the store availability of SKUs .

Suppose γ_r^k is the *percentage* of SKUs having k stores for retailer r . The regression computed the best fit for $\log \gamma_r^k = \delta_r + \eta_r \log k$.

Table 4 Power law degree parameters for each retailer

Retailer	A	B	C	D
δ	10.486	11.832	10.195	9.641
η	-0.871	-1.268	-1.2	-1.134

The power law and the long-tail nature of the SKU availability in stores is a major motivation for our work for two reasons: (i) An SKU can be present in multiple stores and therefore it is important to find the right store to fulfill it in order. (ii) Very few copies of the SKU are likely to be present in each store; hence there is considerable chance that we are unable to find the SKU in the store resulting in multiple tries. To model and sample from the fitted distributions, we truncate the power law at 500 stores and normalize to construct a synthetic distribution.

2. *Distribution of Shipping Costs.* USPS classifies the origin-destination distance into 8 zones, with zone 1 and zone 8 representing the shortest and longest distance respectively. Each zone has a shipping cost given as a function of the weight of the SKU (in lbs.) shipped. We use a publicly available data set of all the US zip codes along with their estimated population produced by the census (?), to come up with the distribution of zones for a store, given a random customer location. It is typically the case that stores are distributed close to population centers. Therefore, to simulate the customer-store zone distribution, we randomly sample pairs of (origin, destination) zip codes (using census data) and then use the USPS zone charts website (?) to determine the shipping zone for that pair of zip codes. The resulting distribution of zones is shown in Figure 8.

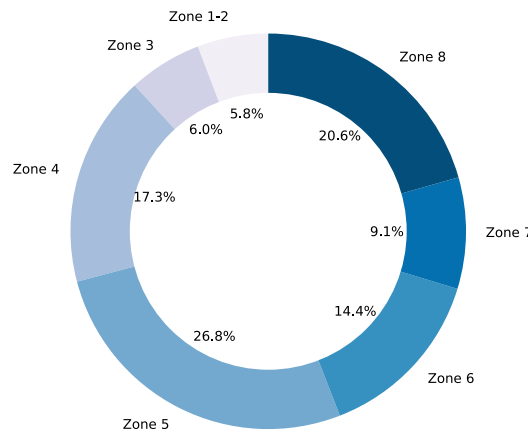
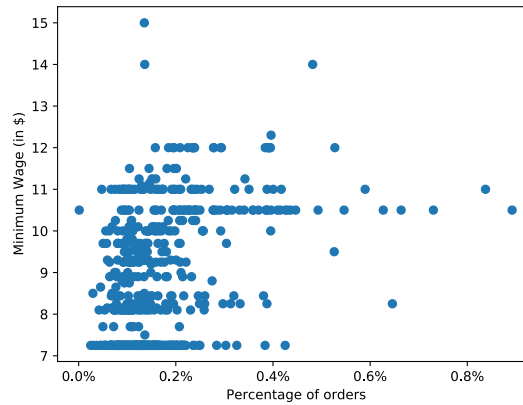
Figure 8 Percentage of stores in different zones with respect to a typical customer

Figure 8 represents the distribution of zones of stores from a customer location, generated at random using census data. Assuming each SKU weighs 4lbs., the shipping cost (s_j for store j) is determined from the USPS Ground website (?) for a given zone in our simulations.

3. *Distribution of Labor Costs.* To sample labor costs, we use a representative retailer's per-store volume and per-store minimum wage. Figure 9 represents the distribution of store volume and store minimum wage provided by the representative retailer.

Figure 9 Minimum hourly wage distribution for a representative retailer



Note. A dot in the plot represents a store, with its x -coordinate showing the percentage of total retailer's volume which that store handles, and the y -coordinate represents the minimum wage at that store location.

To sample a store's labor cost, we sample a store from the list of stores for this retailer where the probability of picking a store is proportional to the store's sales volume. Once the store is picked we estimate the labor cost from the minimum wage data of the store based on its location.

These labor cost estimates are upper bounds on real costs. Minimum wage is not representative of actual handling costs due to sunk labor costs and the fact that typically 4-5 items can be processed within an hour. We scale these costs *down* by 4 to be realistic but scaling them by any other reasonable value does not impact the relative gains of our methods.

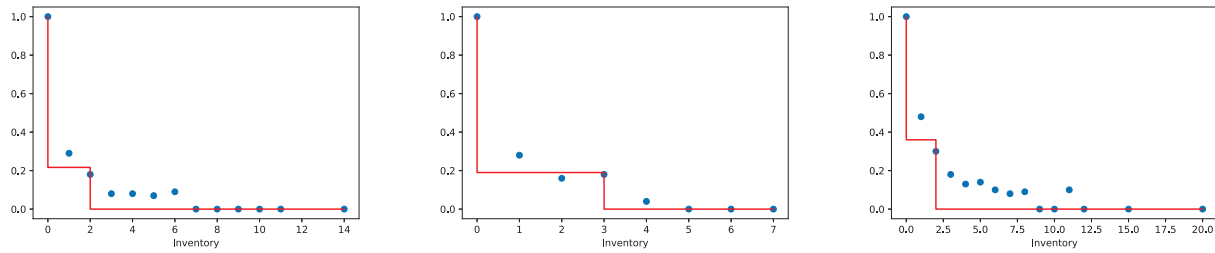
4. *Inventory Distribution & Pick Failure Probability.* By far the most important feature in predicting the pick failure probability is the location of the store itself. We received data from the provider where each row consists of an SKU that is ordered, the store that was attempted with its inventory of that SKU and the pick success/failure information.

We set the pick failure probability function at store j to be a simple step function of inventory so that there is constant pick failure ϕ_j up to a certain level of inventory \tilde{Q}_j and no pick failure beyond that. The fulfillment costs can be illustrated as

$$t_{ijk}(Q_j^P) = \begin{cases} b_j + (1 - \phi_j) \cdot s_{ij} + \phi_j \cdot d_i & \text{if } k \geq (Q_j^P - \tilde{Q}_j)^+ \\ b_j + s_{ij} & \text{otherwise} \end{cases}$$

Figure 10 shows the fit of the pick failure probabilities as a function of inventory levels at three stores.

Figure 10 One-step fit of pick failure probabilities at 3 stores as a function of the inventory



Sampling an order. We generate a random order for each retailer for the single-order fulfillment models (Sections 2 and 3) using the following procedure. Note that pick failure probabilities, labor costs and shipping costs at the store sampled for an SKU are all generated independently of each other.

1. *Number of stores that carry an SKU (J).* The number of stores that carry an SKU is sampled using a power law distribution with parameter values taken from Table 4 for each retailer. For example, if $J(B)$ is the random variable for the number of stores that carry an SKU at retailer B, then $\Pr[J(B) = k] \propto \exp(11.832) \cdot k^{-1.268}$ for $k = 1, \dots, 500$.
2. *Pick failure probability at store j , (ϕ_j).* For each retailer, we pick the top 5 SKUs that have the highest volume of pick failure. For each of these 5 SKUs, we take the union of all stores that feature those SKUs. For each store in this collection, we fit the best 1-step pick failure probability function.

To sample a store's pick failure probability (for any SKU),

- (a) We sample the store j itself from the list of selected stores in a way that the probability of picking that store is proportional to that store's sales volume.
- (b) *Sampling inventory (\bar{Q}_j).* For each store, we sample the inventory of the store \bar{Q}_j from a multinomial distribution modeling the inventory level.

- (c) *Generating pick failure probability* (ϕ_j, ϕ_{jl}) For the static pick failure model (Section 2), the pick failure probability ϕ_j is simply obtained by applying the one-step pick failure function to \bar{Q}_j . For the dynamic pick failure model (Section 3), we set the physical demand at store j , \mathcal{D}_j^P at each stage of fulfillment, to be $Poisson(0.6 * \bar{Q}_j)$.

$$\phi_{jl} = \mathbb{E} \left[\phi_j \left(\bar{Q}_j - \sum_{l'=1}^l \mathcal{D}_{j'}^P \right) \right]$$

where $\phi_j(\cdot)$ is the one step pick failure probability distribution function.

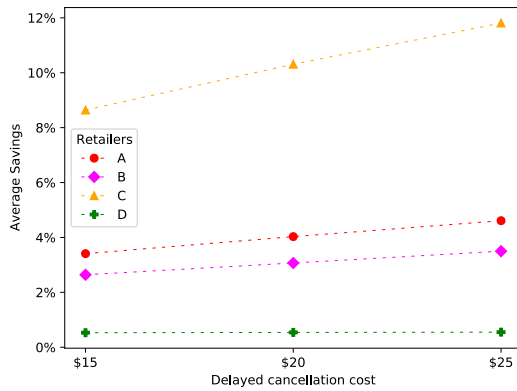
3. *Labor cost at store j* (b_j). As described previously, we sample the store with a probability proportional to its volume and assign all SKUs carried by that store a labor cost equal to the minimum wage at that store location divided by a scaling factor of 4.
4. *Shipping cost at store j* (s_j). For each store associated with the order, we sample its zone independently from the distribution in Figure 8 and then assign the shipping cost from USPS website (?). We assume each item weighs 4lbs.
5. *Delayed cancellation cost* (d). The delayed cancellation cost is set to \$25 per order at any store.

For multi-order models (Sections 4 and 5), we fix the number of stores, J , to 10 and the number of customer shipping zones, I , to 7. The initial inventory \bar{Q}_j at store j is sampled from the multinomial of inventory levels at stores. The physical demand \mathcal{D}_j^P for store j is $Poisson(0.6 * \bar{Q}_j)$. The online demand from zone i , \mathcal{D}_i^O is $Poisson(0.5 * z_i * \sum_j \bar{Q}_j)$, where z_i is the proportion of orders from zone i calculated as shown in Figure 8. The cancellation cost c and lost-sales cost p are set to \$15 and \$10 per order respectively

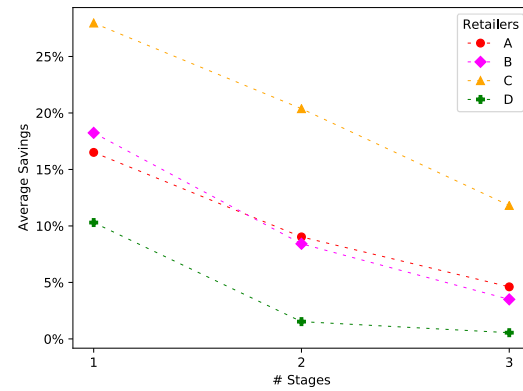
Appendix C: Additional Computational Results

Sensitivity to delayed cancellation cost and the number of stages in single-order fulfillment model under dynamic pick failure probabilities (Section 3). We examine the sensitivity of the cost savings obtained from modeling pick failure to the parameters of our models. We study the variation of savings obtained by our efficient dynamic programming algorithm with respect to the baseline greedy algorithm which is optimal when pick failure is not accounted for. We analyze the savings for our single-order fulfillment model under dynamic pick failure probabilities (Section 3). Figure 11a shows that the average savings increases as the delayed cancellation cost increases. This is because the greedy algorithm is oblivious to pick failure and therefore does not account for the delayed cancellation cost. Figure 11b shows that the savings decrease as the number of stages that an order is tried increases. In other words, the value of modelling pick failure is high when the number of stages of fulfillment is low.

Figure 11 Sensitivity of savings to delayed cancellation cost (d) and number of stages (L)



(a) Average savings across delayed cancellation costs

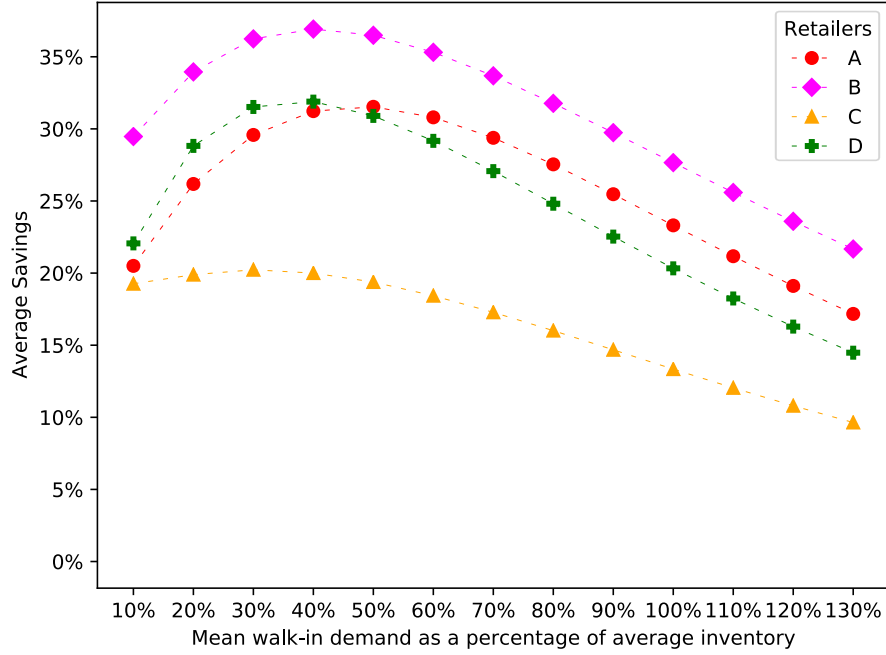


(b) Average savings across number of stages ($d = \$25$)

($L = 3$)

Sensitivity to physical demand in multi-order fulfillment model (Section 4). We vary the means of the physical demands ($\mathbb{E}[\mathcal{D}_j^P]$) as a percentage of available inventory at stores at the beginning of the day (\bar{Q}_j).

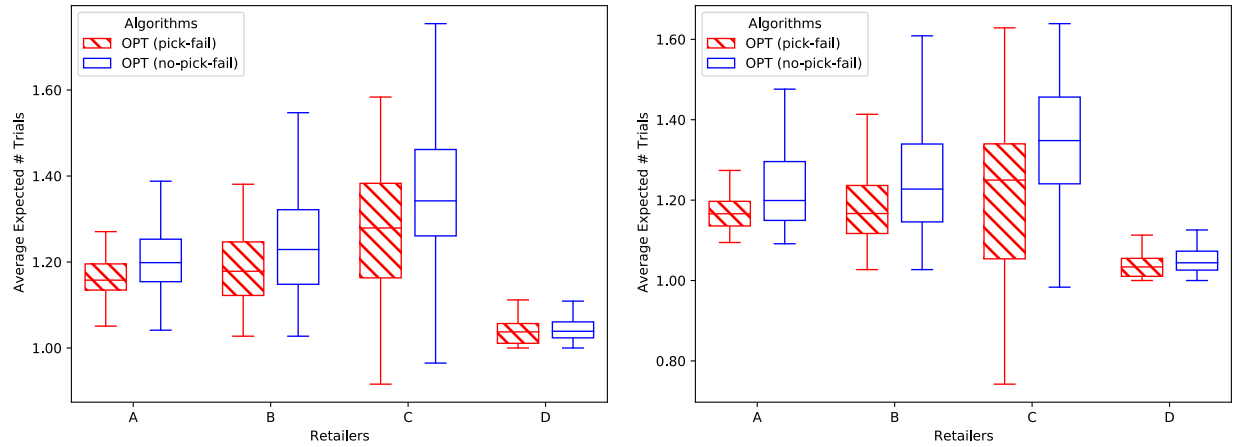
Figure 12 Variation of savings with respect to physical demand as a fraction of inventory when $d = \$25$ and $L = 3$



We observe in Figure 12 that the savings peak when the means of the physical demands are around 40% of initial inventory at the stores. When the physical demand is low, the value of modeling pick failure drops because of lower pick failures at higher inventory levels. On the other hand when the physical demand is significant with respect to the inventory at stores, the optimal policy that takes into account pick failure is relatively ineffective since the store demand depletes inventory much earlier and the delayed cancellation cost becomes a dominant portion of the fulfillment costs.

Comparison of expected number of trials of our policies. In addition to comparing costs of our optimal policies with respect to the benchmark policies, we compare the expected number of trials for our optimal policy against the benchmark policies for models in Sections 2 and 3 in Figure 13.

Figure 13 Improvement in average expected number of trials



(a) Single-order fulfillment under static pick failure probabilities (Section 2) (b) Single-order fulfillment under dynamic pick failure probabilities (Section 3)

We use an example to explain the computation of expected number of trials for a policy. Let's consider a 3-stage problem with a fulfillment policy with stores, say store 1, store 2 and store 3 in order. Let $(\phi_1, \phi_2, \phi_3) = (0.2, 0.6, 0.5)$. The expected number of trials for fulfillment policy $[1, 2, 3]$ is $1 * (1 - 0.2) + 2 * 0.2 * (1 - 0.6) + 3 * 0.2 * 0.6 * (1 - 0.5)$.