1. Introduction

Distribution networks for online order fulfillment usually operate under stochasticity. A network that transports goods from fulfillment centers to customers to fulfill purchase orders experiences stochasticity both in customer purchases and in inventory replenishment. Customer purchases occur with varying purchase times, delivery destinations, and item quantities. Replenishment occurs at specific times and is subject to delays due to stochastic travel times or delays from further up the supply chain. Consequently, stochasticity is a characteristic of realistic operations.

Mathematical techniques for modeling stochasticity are well established. Unfortunately, solving stochastic optimization models of distribution networks is often computationally challenging for the large networks encountered in practice. Networks may include hundreds of nodes organized in multiple layers with tens of thousands of individual products available for shipment. The number of individual decisions, i.e. at each time step, whether to ship a specific item from a specific fulfillment center, directly or through multiple layers of sort centers, can result in a correspondingly large number of integer variables. Decisions also need to be made relatively quickly to be useful for real-time operations. Therefore, optimizing models with explicit stochasticity is a difficult computational problem. Even deterministic optimization problems that ignore the stochasticity are often too large to be solved to optimality quickly enough for real-time control of large networks.

This paper proposes a novel approach for the optimal control of distribution networks under stochasticity. Rather than formulate a stochastic model that will be solved computationally, we propose to identify mathematical properties of the model that correspond to useful performance metrics and find computationally efficient policies that satisfy those properties. Specifically, we consider a distribution network in which purchase orders and corresponding packages accumulate at fulfillment centers and intermediate nodes until they are delivered to the customer. We define *stability* as the long-run average number of accumulated packages remaining bounded. Since customers are continuously generating new purchase orders, achieving stability requires that the average rate of customer fulfillment via package delivery is equal to the average rate of purchase orders. Otherwise, unfulfilled purchase orders and/or packages would accumulate over time and prevent boundedness. All reasonable control policies achieve stability for some rates of customer demand. *Maximum stability* is a property of control policies that achieve stability whenever possible. In other words, a maximum-stable control policy fulfills all customer purchase orders whenever the network is capable of it under some control policy. Maximum stability is therefore equivalent to maximizing network throughput for any demand that can be stabilized.

We present the first attempt at designing a real-time max-pressure control policy for distribution networks that is proven to achieve maximum stability. As a first attempt, we make assumptions

designed to enable the proof of maximum stability, but it remains to be seen whether these assumptions constitute a reasonable approach to distribution network control. The main benefits are how the policy achieves favorable computational properties while simultaneously achieving maximum stability. The policy only involves optimizing the decisions for the next time step, as opposed to looking ahead multiple time steps. Furthermore, the information required to achieve maximum stability is quite low. For instance, future and average customer demand is not required, nor are future or average replenishment shipments. The policy is also decentralized so that each node can compute its optimal control independent of other nodes, which makes it easy to parallelize and perhaps possible to compute in real-time even for large operations. It is possible that using available information or centralizing the decision could improve some performance metrics. However, variants of max-pressure control have even been shown to be optimal or asymptotically optimal for some performance metrics such as average delay in communication networks (Stolyar 2004, Ganti, Modiano, and Tsitsiklis 2007). While we do not yet derive equivalent results here, the literature suggests that max-pressure control may eventually be shown to achieve performance guarantees for logistics networks as well. We believe that this approach of designing a simple, computationally efficient policy that is proven to maximize stability is novel to the supply chain literature and potentially useful for practical, large-scale supply chain operations.

Max-pressure control was first developed by Tassiulas and Ephremides (1990) for packet routing in communication networks, which transmit packets of information between computers. Their novel result achieved maximum network throughput in a multilayer network setting with stochastic demand using limited, local information for a decentralized control. Their seminal paper resulted in hundreds of subsequent papers, including further work on communication network control and a textbook on the application of Lyapunov drift to stability of stochastic networks (Neely 2022). Max-pressure control is a staple of its field and has started to receive attention for use on other problems.

Despite its use for communication networks, it is not immediately obvious how to use max-pressure control for logistics networks. The assumptions associated with computer packet routing apply partially, but not entirely, to distribution networks. For instance, warehouse inventory constrains order fulfillment like service capacity but is endogenous in logistics networks since it depends on previous actions of the policy, unlike capacity in communication networks. Nevertheless, similar limitations have been overcome for other applications. Max-pressure control was proposed for traffic signal timing by Wongpiromsarn et al. (2012) and Varaiya (2013). Despite the differences between road networks controlled by traffic signals and communication networks, numerical simulation results suggest that maximum-stability control is effective there (Sun and Yin 2018, Barman

and Levin 2022). Max-pressure control has also been used for the dispatch of ridesharing vehicles (Kang and Levin 2021, Li et al. 2021). To the best of our knowledge, this is the first paper applying max-pressure control to supply chains networks.

1.1. Literature review

The design and operation of distribution networks encompasses many issues including delivery service facilities, customer pricing, supply management, and internal warehouse design. Most of these issues have been studied in the literature (Agatz, Fleischmann, and Van Nunen 2008). We focus on online order fulfillment and middle-mile package routing, which excludes the vehicle routing problem for last-mile deliveries (Sluijk et al. 2022) and delivery time slot management (Agatz et al. 2011). The problem usually centers around which fulfillment centers should be used to fulfill customer orders. Using fulfillment centers near the customer will reduce shipping costs, but that may overload certain fulfillment centers while leaving others under-utilized due to limited inventories and/or fulfillment center capacities. Order fulfillment has been studied in previous work, including with stochastic modeling. Acimovic and Graves (2015) and Jasin and Sinha (2015) formulated the problem as a Markov decision process (MDP) and used linear programming heuristics to solve it. Jasin and Sinha (2015) and Alibeiki, Hassanlou, and Jorjani (2020) solved the stochastic problem with a linear program on the expected value of random variables. Due to the curse of dimensionality, solving this MDP exactly using conventional methods is not computationally tractable. Sarimveis et al. (2008) reviewed the various general modeling and control approaches for dynamic supply chain systems. Besides heuristics to solve the MDP, another option is using model predictive control. Acimovic and Farias (2019) described two approaches to solving online fulfillment optimization, linear programming and a primal-dual algorithm. Andrews et al. (2019) presented a case study where a primal-dual algorithm was used for Urban Outfitters. These previous models and this paper do not consider additional savings possible by combining items into a single shipment (Xu, Allgor, and Graves 2009). The complexity of the model is already challenging without these additional variables.

Online order fulfillment has also been integrated with other distribution network decisions. Lei, Jasin, and Sinha (2018) jointly optimized customer pricing and order fulfillment. Inventory allocation is also a natural choice for joint optimization because inventory availability constrains order fulfillment. Lim, Jiu, and Ang (2021) solved a joint inventory allocation and fulfillment optimization using a two-phase approach, and Xu et al. (2020) proposed a dynamic fulfillment policy for a finite horizon.

Omni-channel fulfillment, which refers to fulfilling orders from a mixture of fulfillment centers and retail stores, has received considerable attention recently (Taylor et al. 2019). The problem itself

can be formulated similarly to other order fulfillment approaches, but incorporates more variables and options for shipping. Bayram and Cesaret (2021) compared several approaches for dynamic fulfillment across multiple channels in a single framework. Jiu (2022) constructed a stochastic model, and used a two-phase approach to find a solution. Govindarajan, Sinha, and Uichanco (2021) and Abouelrous, Gabor, and Zhang (2022) modeled the joint inventory and fulfillment problem with omni-channels, and used stochastic programming to solve it. The mathematical approach we use could be extended to omni-channel networks, but choosing the shipping option is an orthogonal issue to the stability results that we aim to introduce here. Lin et al. (2022) jointly optimized fulfillment optimization and facility location by modeling delivery costs and solved it using branch-and-cut with outer approximation.

Overall, previous work on order fulfillment has recognized the stochastic nature of customer orders in their modeling approaches. However, due to the number of variables, solution approaches have used deterministic optimization or other approaches based on the problem structure as heuristics. Research attention has then moved to extended versions of the order fulfillment problem, such as joint optimization or omni-channel fulfillment. The alternative solution approach in this paper will not achieve optimality in terms of shipping costs or other metrics, but it will achieve a mathematical guarantee of throughput performance. Considerations of shipping costs and other objectives can then be added to the approach without losing the mathematical performance guarantee.

1.2. Contributions

The contributions of this paper are as follows: we model the real-time operations of a distribution network under stochastic demand and inventory replenishment, and we seek to control variables associated with customer fulfillment. We propose the mathematical concepts of stability and maximum stability which correspond to network throughput properties. We define a decentralized max-pressure control and we prove that it achieves maximum stability. In the process, we derive equations describing the set of demand that can be served, and we show how these equations can be used for topology optimization. We compare the max-pressure control with a greedy policy on a network based on Amazon's facility locations using randomly generated demand. Numerical results compare stability for varying levels of demand and demonstrate the benefits of stability for performance metrics.

The remainder of this paper is organized as follows: Section 2 introduces the network model of the distribution network and the mathematical concept of stability. Section 3 defines the max-pressure control and proves its stability properties. Section 4 discusses how the methodology can be applied to network topology design. We study the max-pressure control on a large numerical example in Section 5, and we conclude in Section 6.

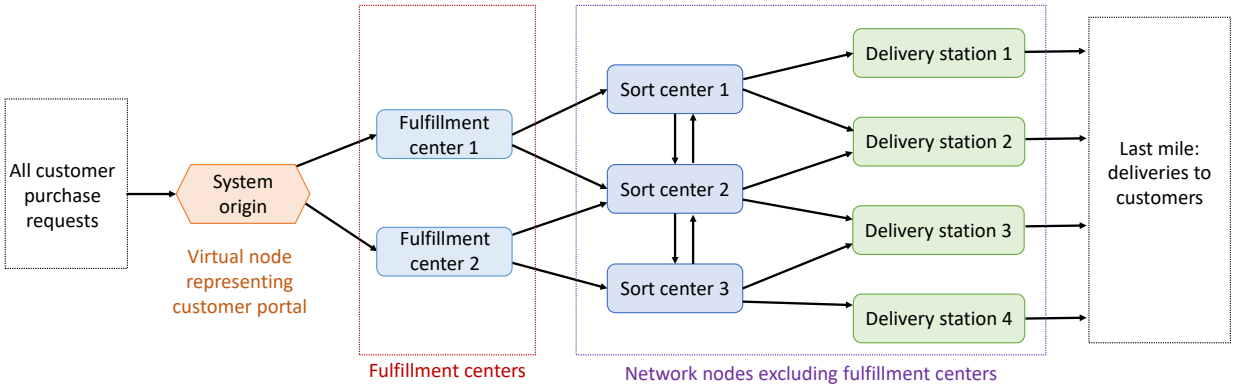


Figure 1 **Example network illustrating network model**

2. Network model

We consider a distribution network problem in which customers send purchase orders to a central system to be fulfilled. These orders must be routed to a fulfillment center with the correct product in inventory and converted into a shipment that is transported through the distribution network until reaching the destination. We describe the distribution network as an abstract collection of nodes and links. Some nodes correspond to destinations, others represent fulfillment centers where shipments originate, and the remainder are intermediate nodes to organize the transport of shipments. Directed links specify which nodes are connected. We aim to optimize the real-time control of several variables: which fulfillment center fulfills each purchase order and when, and when and how the corresponding shipment is routed from the fulfillment center to the order destination. We assume stochasticity in purchase orders and product inventory that the real-time control must adapt to. Solving a real-time stochastic optimization problem for the network would yield the optimal solution, but the required computation time discourages this approach. Instead, we will propose a simple control and prove that it maximizes the network throughput in terms of customer fulfillment. This section describes the network characteristics and our performance goals. Section 3 will present the real-time control and its favorable mathematical properties.

2.1. Distribution network

Consider a package distribution network $\mathcal{G} = (\mathcal{N}, \mathcal{A})$ where $\mathcal{N}$ is the set of nodes and $\mathcal{A}$ is the set of links. Packages originate at fulfillment centers in response to a product order, and then are moved through the network until reaching the order destination. Order destinations are nodes in $\mathcal{N}$ and can represent individual delivery locations or an intermediate location in which packages are transferred to an external service. Let $\mathcal{F} \subset \mathcal{N}$ be the set of fulfillment centers. Let $\Gamma_i^+ \subseteq \mathcal{N}$ and $\Gamma_i^- \subseteq \mathcal{N}$ be the forward and backward stars of j in terms of nodes, respectively. $\Gamma_j^+ = \{k : (j,k) \in \mathcal{A}\}$ and $\Gamma_j^- = \{i : (i,j) \in \mathcal{A}\}$. The distribution network is illustrated in Figure 1.

Let ϕ_i^{sd} be the minimum cost-to-go from node i to node d for a package of size s. ϕ_i^{sd} could represent travel times, shipping cost, etc. for shipping a package. We assume that ϕ_i^{sd} is constant. Packages are not always shipped along the minimum cost path depending on the state of the network. If the network is empty (network capacity is heavily under-utilized), then ϕ_i^{sd} will provide most of the routing guidance. If the network is congested (many packages waiting for transport) then the effects of ϕ_i^{sd} on package routing will be much lower. ϕ_i^{sd} will be used in Section 3.

We track the state of the network over time, including the number of shipments at each node, the inventory of individual products, and the number of unfilled purchase orders. Assume that time is discretized into time steps. We further assume that each link has a travel time of 1 time step to take advantage of the Markov property. To satisfy this assumption, longer links (e.g. long truck trips) are separated into multiple segments each with travel time of 1 time step. The nodes between these links have a special control: all packages are forwarded to the next link without delay.

Let $\mathcal{S}$ be the set of package sizes. Let $x_i^{sd}(t)$ be the number of packages of size s destined for d at node i at time t, with $\mathbf{x}(t)$ the vector of package numbers. Let $y_{ij}^{sd}(t)$ be the number of packages of size s destined for d moved from i to j at time t, with $\mathbf{y}(t)$ the vector of package movement. For non-fulfillment center nodes, the number of packages at the node evolves through conservation:

$$x_j^{sd}(t+1) = x_j^{sd}(t) - \sum_{k \in \Gamma_j^+} y_{jk}^{sd}(t) + \sum_{i \in \Gamma_j^-} y_{ij}^{sd}(t) \qquad \forall j \in \mathcal{N}, \forall s \in \mathcal{S}, \forall d \in \mathcal{N}, \forall t \tag{1}$$

For destination nodes, $x_d^{sd}(t) = 0$. Packages are removed from the network once they reach their destination. If the destination d is not reachable from some node j, then packages destined for d should not be shipped to that node, i.e. $y_{ij}^{sd}(t) = 0$.

2.2. Purchase order model

Let $\mathcal{P}$ be the set of products. Customers enter purchase orders into a central system, which routes them to fulfillment centers with appropriate inventory to be converted into a shipment. Each purchase order is associated with a product p and a destination node d. Let $r^{pd}(t)$ be the number of new orders for product p to destination d at time t. We assume that $r^{pd}(t)$ are random variables with mean $\bar{r}^{pd}$, and we also assume that $r^{pd}(t)$ are independent and identically distributed with respect to time. The vector of new purchase orders is denoted $\mathbf{r}(t)$, and represents new customer demand at time t.

All orders for a product delivery start at the central origin for the delivery system, labeled o, which represents the customer ordering system. For instance, if customers enter orders into a website, o would represent that website. The origin must then route the orders to a fulfillment center to be converted into a shipment. We assume that this routing is electronic and achieved

within 1 time step. Fulfilment center allocation could depend on the inventory levels of product p at each fulfillment center and the travel cost from the fulfillment center to the destination. Orders wait at o until the product is in stock and the fulfillment center has sufficient capacity. After the conditions are met, the fulfillment center converts the order into a shipment that enters the distribution network.

Let $\chi^{pd}(t)$ be the number of orders for product p to be delivered to node d waiting for fulfillment at the origin o. Let $\gamma_i^{pd}(t)$ be the number of (p,d) orders fulfilled by fulfillment center i at time t. Let $\boldsymbol{\chi}(t)$ and $\boldsymbol{\gamma}(t)$ be the vectors of $\chi^{pd}(t)$ and $\gamma_i^{pd}(t)$, respectively. χ^{pd} evolves via conservation:

$$\chi^{pd}(t+1) = \chi^{pd}(t) + r^{pd}(t) - \sum_{i\in\mathcal{F}} \gamma_i^{pd}(t) \qquad \forall p\in\mathcal{P}, \forall d\in\mathcal{N}, \forall t \tag{2}$$

Purchase orders are converted into a shipment at the fulfillment center. Each product has an associated package size, and to simplify the math we assume that separate packages are generated for each order. Let $\mathcal{P}^s$ be the set of products of size s, and let $\imath(p)$ be the size of a package containing product p. Fulfilled orders are converted into a shipment at fulfillment center i:

$$x_i^{sd}(t+1) = x_i^{sd}(t) + \sum_{p\in\mathcal{P}^s} \gamma_i^{pd}(t) - \sum_{j\in\Gamma_i^+} y_{ij}^{sd}(t) \qquad \forall i\in\mathcal{F}, \forall s\in\mathcal{S}, \forall d\in\mathcal{N}, \forall t \tag{3}$$

When a product is converted into a package of the right size, the destination index d of the product is retained: $\gamma_i^{pd}(t)$ converts to $x_i^{sd}(t+1)$ in equation (3). Since packages can only leave the network when reaching their destination d, this ensures that the correct product p is routed to the destination d. The destination node could be either the actual customer's location, or a last-mile delivery station. If the latter, then the last-mile station would receive a package containing the correct product, and proceed with last-mile delivery from there. The purpose of converting product index p into package size index s is to reduce the number of variables. The number of package sizes is likely much smaller than the number of products.

2.3. Inventory model

We separately track the inventory of product p available at each fulfillment center i. Let $v_i^p(t)$ be the inventory available, and let $\sigma_i^p(t+1)$ be the maximum replenishment at time t. Since that replenishment will be available at time $t+1$, the time index $t+1$ is used to avoid later confusion in the notation. The inventory also evolves via conservation. However, maximum replenishment of the inventory could grow arbitrarily large without affecting the stability of order fulfillment and package shipments, but such a large inventory is unrealistic due to finite storage space. We therefore assume that an upper bound on inventory stored, V_i^p, also constrains the inventory.This forces the

inventory to be stable; at node i the inventory is bounded by $\sum_{p\in\mathcal{P}} V_i^p$. The actual replenishment will be based on the quantity of inventory to be replenished, with a maximum amount of $\sigma_i^p(t+1)$.

$$v_i^p(t+1) = \min\left\{v_i^p(t) - \sum_{d\in\mathcal{N}} \gamma_i^{pd}(t) + \sigma_i^p(t+1), V_i^p\right\} \qquad \forall i\in\mathcal{F}, \forall p\in\mathcal{P}, \forall t \tag{4}$$

We assume that $\sigma_i^p(t)$ are independent random variables that are identically distributed with respect to time with mean $\bar{\sigma}_i^p$. $\sigma_i^p(t)$ could be a constant, but as a random variable it could model variations in product availability at suppliers. We further assume that $V_i^p \geq \sigma_i^p(t)$, i.e. the storage exceeds any single replenishment. Intuitively, the average replenishment rates $\sum_{i\in\mathcal{F}} \bar{\sigma}_i^p$ should be related to the average rate of customer orders, $\sum_{d\in\mathcal{N}} \bar{r}^{pd}$. We will require the existence of a buffer, i.e. $\sum_{i\in\mathcal{F}} \bar{\sigma}_i^p > \sum_{d\in\mathcal{N}} \bar{r}^{pd}$, to ensure that temporarily high customer demands that occur randomly can still be fulfilled in a finite time.

2.4. Shipment constraints

We place several constraints on the movement of purchase orders and packages through the system. First, purchase orders can only be transmitted to a fulfillment center when they exist:

$$\sum_{i\in\mathcal{F}} \gamma_i^{pd}(t) \leq \chi^{pd}(t) \qquad \forall p\in\mathcal{P}, \forall d\in\mathcal{N}, \forall t \tag{5}$$

Purchase orders also require available inventory at the fulfillment center to be transmitted there:

$$\sum_{d\in\mathcal{N}} \gamma_i^{pd}(t) \leq v_i^p(t) \qquad \forall i\in\mathcal{F}, \forall p\in\mathcal{P}, \forall t \tag{6}$$

Finally, the number of orders transmitted to a fulfillment center per time step cannot exceed the fulfillment center capacity C_i:

$$\sum_{p\in\mathcal{P}}\sum_{d\in\mathcal{N}} \gamma_i^{pd}(t) \leq C_i \qquad \forall i\in\mathcal{F}, \forall t \tag{7}$$

Separate constraints are placed on moving packages. First, movement is only possible if the connection exists: $y_{ij}^{sd}(t) = 0$ if $(i,j)\notin\mathcal{A}$. Each node i has a capacity limit on the total outbound packages, C_i:

$$\sum_{j\in\Gamma_i^+}\sum_{s\in\mathcal{S}}\sum_{d\in\mathcal{N}} y_{ij}^{sd}(t) \leq C_i \qquad \forall i\in\mathcal{N}, \forall t \tag{8}$$

We assume that each connection (i,j) has a package capacity C_{ij}:

$$\sum_{s\in\mathcal{S}}\sum_{d\in\mathcal{N}} y_{ij}^{sd}(t) \leq C_{ij} \qquad \forall (i,j)\in\mathcal{A}, \forall t \tag{9}$$

and a limit on the total package space:

$$\sum_{s\in\mathcal{S}}\sum_{d\in\mathcal{N}} y_{ij}^{sd}\times s \le S_{ij} \qquad \forall (i,j)\in\mathcal{A},\forall t \tag{10}$$

We also constrain the number of active outbound connections per node i, which could represent a limit on the number of trucks they have to send out. Let $\zeta_{ij}(t)\in\{0,1\}$ indicates whether a shipment is sent from i to j.

$$\sum_{s\in\mathcal{S}}\sum_{d\in\mathcal{N}} y_{ij}^{sd}(t) \le C_{ij}\zeta_{ij}(t) \qquad \forall (i,j)\in\mathcal{A},\forall t \tag{11}$$

Then the number of outbound connections is constrained by Z_i:

$$\sum_{j\in\Gamma_i^+} \zeta_{ij}(t) \le Z_i \qquad \forall i\in\mathcal{N},\forall t \tag{12}$$

Finally, packages cannot be moved unless they are present:

$$\sum_{i\in\Gamma_j^-} y_{ij}^{sd}(t) \le x_i^{sd}(t) \qquad \forall j\in\mathcal{N},\forall s\in\mathcal{S},\forall d\in\mathcal{N},\forall t \tag{13}$$

Equation (13) is stronger than non-negativity because it is possible to satisfy equation (3) with $\sum_{p\in\mathcal{P}^s}\gamma_i^{pd}(t) > \sum_{j\in\Gamma_i^+} y_{ij}^{sd}(t) > x_i^{sd}(t)$.

2.5. Markov decision process

The state of the network is described by $\mathbf{x}(t)$, $\boldsymbol{\chi}(t)$, and $\mathbf{v}(t)$. Let $\mathbf{z}(t)=(\mathbf{x}(t),\boldsymbol{\chi}(t),\mathbf{v}(t))$ represent the entire network state. Let $\mathcal{Z}$ be the set of all possible states. The state at time $t+1$ depends only on the state at time t, control variables at time t, and random effects at time t. Therefore, it satisfies the Markov property. The control variables are $\mathbf{y}(t)$ and $\boldsymbol{\gamma}(t)$. $y_{ij}^{sd}(t)$ determines both the time of package movement and its routing, one step at a time. $\gamma_i^{pd}(t)$ determines which fulfillment center the purchase order is routed to and the start of its routing for delivery. Let $\mathbf{u}(t)=(\mathbf{y}(t),\boldsymbol{\gamma}(t)$ represent the network control. Let $\mathcal{U}$ be the set of feasible controls, i.e. the controls satisfying constraints (5)–(13). A control policy $\pi:\mathcal{Z}\to\mathcal{U}$ defines the feasible control $\mathbf{u}(t)\in\mathcal{U}$ taken when the state is $\mathbf{z}(t)\in\mathcal{Z}$. The system ($\mathcal{Z}$,$\mathcal{U}$) defines a Markov decision process, where the stochasticity in the state transitions comes from $\mathbf{r}(t)$ and $\boldsymbol{\sigma}(t)$.

Although standard algorithms could be applied to this Markov decision process, it suffers from the curse of dimensionality due to the problem size. Instead, our solution approach is to define a favorable mathematical property, maximum stability, that would be nice to achieve. We will then propose a control policy and prove that it achieves maximum stability.

2.6. Network stability

The concept of stability is closely related to network throughput, as will soon become clear. We define stability as follows:

DEFINITION 1 (STABILITY). The network is *stable* if there exists a $\kappa < \infty$ such that

$$\lim_{T\to\infty} \frac{1}{T} \sum_{t=1}^{T} \left(\sum_{p\in\mathcal{P}} \sum_{d\in\mathcal{N}} \mathbb{E}\left[\chi^{pd}(t)\right] + \sum_{s\in\mathcal{S}} \sum_{d\in\mathcal{N}} \sum_{i\in\mathcal{N}} \mathbb{E}\left[x_i^{sd}(t)\right] \right) \leq \kappa \tag{14}$$

Because purchase orders continue to arrive every time interval, $\chi^{pd}(t)$ will increase as those purchase orders arrive. Therefore, $\sum_{p\in\mathcal{P}} \sum_{d\in\mathcal{N}} \mathbb{E}[\chi^{pd}(t)]$ will increase to infinity unless purchase orders are filled at the same rate at which they arrive. They cannot be filled at a higher rate because of constraint (5). Similarly, as orders are fulfilled, they will generate packages to be shipped, which will add to $x_i^{sd}(t)$ for some i. Therefore, $\sum_{s\in\mathcal{S}} \sum_{d\in\mathcal{N}} \sum_{i\in\mathcal{N}} \mathbb{E}[x_i^{sd}(t)]$ will increase to infinity unless packages are shipped through the network to their destination at the same rate at which they are generated at fulfillment centers. Packages must reach their destination to avoid accumulating somewhere in the network; they can only be removed by shipping them to the destination. In other words, stability means that all purchase orders are fulfilled and all packages are delivered.

It is impossible for equation (14) to equal 0: some positive number of packages will exist in the network due to shipping time requirements. Some positive number of purchase orders will exist at fulfillment centers due to stochasticities in inventory and capacity vs. demand. This is a normal characteristic of all distribution networks. Stability is only an issue if equation (14) increases to infinity instead of remaining bounded.

All reasonable networks exhibit stability for some demands, and not for others. It is obvious that stability is impossible under certain simple situations, such as $\sum_{d\in\mathcal{N}} \bar{r}^{pd} > \sum_{i\in\mathcal{F}} \bar{\sigma}_i^p$ for any product p. The question concerning this paper is whether a network exhibits stability for as much demand as possible. To define this, we must first define when stability is possible.

DEFINITION 2 (STABLE REGION). Let $\mathcal{R}$ be the *stable region* of demand, i.e. the set of demands that are compatible with a stable network. More formally, if and only if $\bar{\mathbf{r}} \in \mathcal{R}$, then there exists a control policy π that will achieve a stable network when demand is $\bar{\mathbf{r}}$.

The network stable region includes the set of stable demands for all control policies. If a demand can be stabilized by some control policy, then that demand is included in the stable region. Therefore, serving all demand rates in the stable region means that we serve at least as much demand as every other control policy. We can now define maximum stability, the property we want to achieve:

DEFINITION 3 (MAXIMUM STABILITY). A policy π achieves *maximum stability* if it serves all demand in $\mathcal{R}$.

$\mathcal{R}$ defines the stable region, or the maximum set of demands that can be stabilized. Recall from Definition 1 that a policy π that stabilizes $\bar{\mathbf{r}}$ has network throughput equal to $\bar{\mathbf{r}}$. Therefore, if π stabilizes any demand in $\mathcal{R}$, then it has network throughput equal to any $\bar{\mathbf{r}} \in \mathcal{R}$. Since any $\bar{\mathbf{r}} \notin \mathcal{R}$ cannot be stabilized by Definition 2, serving any demand in $\mathcal{R}$ is the best that a policy can hope to achieve. Because stability means that purchase orders are both fulfilled at fulfillment centers and shipped to customers, maximum stability means that policy π serves customers as much as possible. In other words, maximum stability means that π achieves maximum network throughput whenever $\bar{\mathbf{r}} \in \mathcal{R}$. However, maximum stability does not make any claims about throughput when $\bar{\mathbf{r}} \notin \mathcal{R}$.

2.7. Stable region of demand

$\mathcal{R}$ itself is defined in terms of Definition 2. For the purposes of proving stability properties and network topology optimization, an explicit mathematical characterization of $\mathcal{R}$ is useful. We propose such a characterization, which we will call $\hat{\mathcal{R}}$, and we will later prove that $\hat{\mathcal{R}} = \mathcal{R}$.

A control policy π determines $\gamma_i^{pd}(t)$ and $y_{ij}^{sd}(t)$ in response to the network state $\mathbf{z}(t)$ which is partly determined by stochastic inputs. Defining $\hat{\mathcal{R}}$ involves giving an average description of the network controls. Let $\bar{\gamma}_i^{pd}$ be the average rate of movement of purchase orders from o to fulfillment center $i \in \mathcal{F}^p$:

$$\bar{\gamma}_i^{pd} = \lim_{T\to\infty} \frac{1}{T} \sum_{t=1}^{T} \gamma_i^{pd}(t) \tag{15}$$

where T is used for the time horizon. Let $\bar{y}_{ij}^{sd}$ be the average rate of shipments of packages of size s destined to d from i to j:

$$\bar{y}_{ij}^{sd} = \lim_{T\to\infty} \frac{1}{T} \sum_{t=1}^{T} y_{ij}^{sd}(t) \tag{16}$$

The following equations ensure conservation of flow throughout the network:

$$\sum_{i\in\mathcal{F}} \bar{\gamma}_i^{pd} = \bar{r}^{pd} \qquad \forall p \in \mathcal{P}, \forall d \in \mathcal{N} \tag{17}$$

$$\sum_{j\in\Gamma_i^+} \bar{y}_{ij}^{sd} = \sum_{p\in\mathcal{P}^s} \bar{\gamma}_i^{pd} \qquad \forall s \in \mathcal{S}, \forall d \in \mathcal{N}, \forall i \in \mathcal{F} \tag{18}$$

$$\sum_{i\in\Gamma_j^-} \bar{y}_{ij}^{sd} = \sum_{k\in\Gamma_j^+} \bar{y}_{jk}^{sd} \qquad \forall s \in \mathcal{S}, \forall d \in \mathcal{N}, \forall j \in \mathcal{N}/\{d\} \tag{19}$$

Equations (17)–(19) describe the average flows in and out of each node, corresponding to equations (1), (3), and (4).

The average controls are also limited by control constraints. Purchase order fulfillment is limited by inventory availability

$$\bar{\gamma}_i^p \le \bar{\sigma}_i^p \qquad \forall i \in \mathcal{F}, \forall p \in \mathcal{P} \tag{20}$$

and by fulfillment center capacity

$$\sum_{p\in\mathcal{P}} \sum_{d\in\mathcal{N}} \bar{\gamma}_i^p \le C_i \qquad \forall i \in \mathcal{F} \tag{21}$$

Constraints (20)–(21) correspond to control constraints (6)–(7). The shipment of packages from node i to node j is limited by capacity and shipment size constraints:

$$\sum_{s\in\mathcal{S}} \sum_{d\in\mathcal{N}} \bar{y}_{ij}^{sd} \le C_{ij} \qquad \forall (i,j) \in \mathcal{A} \tag{22}$$

$$\sum_{j\in\mathcal{N}} \sum_{s\in\mathcal{S}} \sum_{d\in\mathcal{N}} \bar{y}_{ij}^{sd} \le C_i \qquad \forall i \in \mathcal{N} \tag{23}$$

$$\sum_{s\in\mathcal{S}} \sum_{d\in\mathcal{N}} \bar{y}_{ij}^{sd} \times s \le S_{ij} \qquad \forall (i,j) \in \mathcal{A} \tag{24}$$

Constraints (22)–(24) correspond to control constraints (8)–(11). We will see in Lemma 1 that constraints (12)–(13) do not limit the average shipping rates because we can combine shipments to satisfy constraints (12)–(13) that on average match $\bar{\mathbf{y}}$. Let $\hat{\mathcal{R}}$ be the set of demands $\bar{\mathbf{r}}$ satisfying equations (17)–(24). Since equations (17)–(24) are linear, $\hat{\mathcal{R}}$ is a convex set.

LEMMA 1. *Assume that for all $i \in \mathcal{N}$, any selection of Z_i outbound links $\mathcal{A}_i^{\mathrm{Z}}$ satisfies*

$$\sum_{j\in\mathcal{A}_i^{\mathrm{Z}}} C_{ij} \ge C_i \tag{25}$$

Then equation (12) *does not constrain the average shipping rates for $\hat{\mathcal{R}}$.*

Proof. Consider a node i and a $\bar{y}_{ij}^{sd}$ satisfying equations (22)–(24) corresponding to a $y_{ij}^{sd}(t)$ control that satisfies equations (9)–(11) but not equation (12). Define shipment groups indexed by ξ satisfying

$$\sum_{s} \sum_{d} \bar{y}_{ij}^{sd}(\xi) \le C_{ij} \qquad \forall j \in \Gamma_i^+ \tag{26}$$

$$\sum_{j} \sum_{s} \sum_{d} \bar{y}_{ij}^{sd}(\xi) = C_i \tag{27}$$

$$\bar{y}_{ij}^{sd}(\xi) \le C_{ij} z_{ij}(\xi) \qquad \forall j \in \Gamma_i^+ \tag{28}$$

$$\sum_{ij} z_{ij}(\xi) \le Z_i \tag{29}$$

and consider all possible such shipment groups. Let $\lambda_\xi \in [0,1]$ be the proportion of time spent in shipment group ξ. Then there exists some $\boldsymbol{\lambda}$ satisfying

$$\sum_\xi \lambda_\xi \bar{\mathbf{y}}(\xi) = \bar{\mathbf{y}} \tag{30}$$

The chosen shipment groups use the full capacity by making equation (27) an equality, which is possible by assumption. Then we can choose $\boldsymbol{\lambda}$ with $\sum_\xi \lambda_\xi \leq 1$ because $\bar{\mathbf{y}}$ satisfies equation (21). This yields an alternative shipment pattern $\bar{\mathbf{y}}(\xi)$ which satisfies equation (12) but still achieves the original average shipment rates $\bar{\mathbf{y}}$. □

We aim to simultaneously establish that $\hat{\mathcal{R}} = \mathcal{R}$ and prove that our proposed control policy achieves maximum stability. To do so, we will first relate the boundary of $\hat{\mathcal{R}}$ to $\mathcal{R}$.

PROPOSITION 1. *If $\bar{\mathbf{r}} \notin \hat{\mathcal{R}}$, then $\bar{\mathbf{r}}$ cannot be stabilized by any control policy.*

Proof. Since $\bar{\mathbf{r}} \notin \hat{\mathcal{R}}$, there exists a quantity $\epsilon > 0$ such that for all $\bar{\mathbf{y}}$ and $\bar{\boldsymbol{\gamma}}$ satisfying equations (17)–(24), there exists a $p \in \mathcal{P}$ and $d \in \mathcal{N}$ such that

$$\bar{\gamma}_i^{pd} + \epsilon \leq \bar{r}^{pd} \tag{31}$$

Then in equation (2), $\mathbb{E}\left[\chi^{pd}(t+1) - \chi^{pd}(t)\right] = \epsilon$, and

$$\lim_{T\to\infty} \frac{1}{T} \sum_{t=1}^{T} \mathbb{E}\left[\chi^{pd}(t)\right] = \lim_{T\to\infty} \frac{1}{T} \sum_{t=1}^{T} \epsilon \times t \to \infty \tag{32}$$

which violates Definition 1 for stability. A similar argument will hold for equations (1) or (3) if the violated constraint causing $\bar{\mathbf{r}} \notin \mathcal{R}$ is pushed to equation (19) or (18), respectively. □

We also need to prove that if $\bar{\mathbf{r}} \in \hat{\mathcal{R}}$ then there exists a control policy π that stabilizes $\bar{\mathbf{r}}$. That requires finding a corresponding control policy $\pi^\star$, which will be achieved in Section 3. However, $\pi^\star$ requires the existence of a tiny gap between its average control and the constraints for stability. This is similar to the difference between null recurrence and positive recurrence of the Markov chain for a simple M/M/1 queue, where null recurrence occurs if the arrival rate equals the departure rate. Therefore, let $\hat{\mathcal{R}}^0$ be the interior of $\mathcal{R}$, where equations (20)–(24) are strict inequalities. We will now derive a control policy $\pi^\star$, which will be proven to stabilize any demand in $\hat{\mathcal{R}}^0$. The approach of proving stability for the interior of $\mathcal{R}$ is typical among the max-pressure literature (e.g. Tassiulas and Ephremides 1990, Varaiya 2013), and it is not known how to prove stability for the boundary of $\mathcal{R}$ (if that is even possible).

3. Maximum-stability control policy

Thus far, we have constructed a model of the distribution network, defined the stability property for a generic control policy π, and defined the stable region $\mathcal{R}$ as well as related set $\hat{\mathcal{R}}$. This section will define a control policy $\pi^\star$ that achieves maximum stability, and in the process complete the relationship between $\mathcal{R}$ and $\hat{\mathcal{R}}$. Finally, we will briefly discuss how the equations describing $\hat{\mathcal{R}}$ can be used for network topology optimization.

3.1. Max-pressure control

We define a max-pressure control inspired by Varaiya (2013). Let $w_{ij}^{sd}(t)$ be the pressure for moving packages destined for node d of size s from node i to node j. We define $w_{ij}^{sd}(t)$ as follows:

$$w_{ij}^{sd}(t) = x_i^{sd}(t) - x_j^{sd}(t) \tag{33}$$

$w_{ij}^{sd}(t)$ increases with the number of packages at i and decreases with the number of packages at j. Similarly, let $\omega_i^{pd}(t)$ be the pressure for fulfilling purchase orders of product p destined for node d at fulfillment center i. We define $\omega_i^{pd}(t)$ as follows:

$$\omega_i^{pd}(t) = \chi^{pd}(t) - x_i^{\jmath(p)d}(t) \tag{34}$$

We use $w_{ij}^{sd}(t)$ for the weight to move packages from i to j and $\omega_i^{pd}(t)$ as the weight to fulfill purchase orders. $\omega_i^{pd}(t)$ increases with the number of purchase orders at the origin and decreases with the number of packages at i waiting to be shipped. Intuitively, the max-pressure control seeks to move packages from nodes with a large backlog to nodes with a small backlog. Similarly, it seeks to fulfill purchase orders at fulfillment centers with a small backlog of packages to be shipped. However, these pressure functions are not determined through intuition, but instead take on a specific form necessary to complete the proof of maximum stability.

We now define policy $\pi^\star$ to be the solution to the following integer linear program:

$$\max \quad \sum_{p\in\mathcal{P}}\sum_{d\in\mathcal{N}}\sum_{i\in\mathcal{F}} \gamma_i^{pd}(t)\left(\omega_i^{pd}(t) - \beta_1\phi_i^{\jmath(p)d}\right) \quad + \sum_{s\in\mathcal{S}}\sum_{d\in\mathcal{N}}\sum_{(i,j)\in\mathcal{A}} y_{ij}^{sd}(t)\left(w_{ij}^{sd}(t) + \beta_2\left(\phi_i^{sd} - \phi_j^{sd}\right)\right) \tag{35a}$$

$$\text{s.t.} \quad \gamma_i^{pd}(t) \in \mathbb{Z}_+ \qquad \forall p\in\mathcal{P}, \forall d\in\mathcal{N}, \forall i\in\mathcal{F} \tag{35b}$$

$$y_{ij}^{sd}(t) \in \mathbb{Z}_+ \qquad \forall s\in\mathcal{S}, \forall d\in\mathcal{N}, \forall (i,j)\in\mathcal{A} \tag{35c}$$

$$\zeta_{ij}(t) \in \{0,1\} \qquad \forall (i,j)\in\mathcal{A} \tag{35d}$$

$$\text{Constraints (5)–(13)}$$

Objective (35a) chooses package movements $\mathbf{y}(t)$ based on the weight $\mathbf{w}(t)$ and purchase order fulfillment $\boldsymbol{\gamma}(t)$ based on the weight $\boldsymbol{\omega}(t)$. The $\beta_2 y_{ij}^{sd}(t)(\phi_i^{sd} - \phi_j^{sd})$ term is used to guide packages on the shortest path. In an empty network, packages will be moved through the network but

not necessarily on an efficient path towards their destination. The shipping cost-to-go values ϕ_i^{sd} is used to guide packages towards the destination based on shipping costs when the number of packages in the network is small. As $w_{ij}^{sd}(t)$ increases, it will dominate $\beta_2\phi_i^{sd}$ since $w_{ij}^{sd}(t)$ can grow arbitrarily large. The similar term $\beta_1\gamma_i^{pd}(t)\phi_i^{sd}(t)$ is added to encourage fulfillment of purchase orders at fulfillment centers near the destination. The constants β_1 and β_2 are used to adjust the relative importance of the pressure terms and the shipping costs. If β is large, then the pressure terms will not be important unless $x_i^{sd}(t)$ is large. That means the system will place a higher priority on moving packages along the shortest path and a lower priority on responding to large backlogs of packages. Other considerations besides shipping cost-to-go may be added in a similar fashion. As long as their value remains bounded while $\omega_i^{pd}(t)$ and $w_{ij}^{sd}(t)$ can grow arbitrarily large, the mathematical guarantee of stability will still hold.

Due to the assumption that all links have a travel time of 1 time step, some trips (e.g. long truck trips) involve multiple links. For such trips, the corresponding $y_{ij}^{sd}(t)$ decision variables in problem (35) can be replaced by the "decision" that $y_{ij}^{sd}(t) = x_i^{sd}(t)$, i.e. all packages are moved to the next link in the trip. This keeps longer trips in the model as separate links while accurately representing their behavior.

It is not immediately clear why problem (35) is worth solving. Problem (35) is just a one-step lookahead to optimize the flow for the next time step. In Section 3.3, we will show that problem (35) achieves maximum stability, which is the main goal of this paper. Meanwhile, we observe that the one-step lookahead lends itself to a nice decomposition of problem (35).

3.2. Decentralized property

Problem (35) is not particularly efficient to solve in that form. Fortunately, we will observe in Proposition 2 that problem (35) can be separated into multiple subproblems which have fewer variables and constraints.

Proposition 2. *Problem* (35) *can be solved by solving the following problem for order fulfillment:*

$$\max \quad \sum_{p\in\mathcal{P}}\sum_{d\in\mathcal{N}}\sum_{i\in\mathcal{F}} \gamma_i^{pd}(t)\left(\omega_i^{pd}(t) - \beta_1\phi_i^{s(p)d}\right) \tag{36a}$$

$$\text{s.t.} \quad \gamma_i^{pd}(t) \in \mathbb{Z}_+ \qquad \forall p\in\mathcal{P}, \forall d\in\mathcal{N}, \forall i\in\mathcal{F} \tag{36b}$$

$$\text{Constraints (5)–(7)}$$

and the following problem for all nodes i:

$$\max \quad \sum_{s\in\mathcal{S}}\sum_{d\in\mathcal{N}}\sum_{(i,j)\in\mathcal{A}} y_{ij}^{sd}(t)\left(w_{ij}^{sd}(t) + \beta_2\left(\phi_i^{sd} - \phi_j^{sd}\right)\right) \tag{37a}$$

$$\text{s.t.} \quad y_{ij}^{sd}(t) \in \mathbb{Z}_+ \qquad \forall s \in \mathcal{S}, \forall d \in \mathcal{N}, \forall (i,j) \in \mathcal{A} \tag{37b}$$

$$\zeta_{ij}(t) \in \{0,1\} \qquad \forall (i,j) \in \mathcal{A} \tag{37c}$$

Constraints (8)–(13)

The solutions to problems (36) *and* (37) *together comprise an optimal solution to problem* (35).

Proof. For problem (36), the variables and constraints associated with package movement along arcs (i,j) do not affect the fulfillment of purchase orders at time t. For problem (37), variables and constraints associated with purchase order fulfillment do not affect package shipping at time t. Furthermore, for any node i, the movement of packages between nodes $i' \neq i$ and any $j' \in \Gamma_{i'}^+$ do not affect package shipping from i to any $j \in \Gamma_i^+$ at time t. Therefore, problem (35) can be separated into problems (36) and (37) without affecting optimality. □

Proposition 2 demonstrates that problem (35) exhibits the *decentralized* property: the optimal solution for package movement outbound from i does not depend on any package shipping decisions at any other nodes. Furthermore, it only uses the information $x_i^{sd}(t)$, the backlog at i, and $x_j^{sd}(t)$ for all $j \in \Gamma_i^+$, the backlog at nodes immediately downstream of i. Therefore, the component of the optimal solution to problem (35) for outbound package shipping from i can be computed while ignoring all other decision variables and constraints. A similar property holds for the decisions about purchase order fulfillment at the origin within the optimal solution to problem (35).

Using Proposition 2, the max-pressure control $\pi^\star$ is formally described in Algorithm 1. Each time step t, problem (36) is solved to obtain the optimal order fulfillment $\gamma_i^{pd\star}(t)$, which respects relevant fulfillment center constraints. Simultaneously, problem (37) is solved per node i to obtain the optimal package movement $y_{ij}^{sd\star}(t)$, which respects relevant node constraints. If equations (2), (3), and (4) are accurate models of the inventory and package state, then lines 3, 6, and 9 of Algorithm 1 represent the actual fulfillment of orders, movement of packages, and replenishment of fulfillment centers, respectively.

Problem (36) weights $\gamma_i^{pd}(t)$ by $\omega_i^{pd}(t) - \beta_1 \phi_i^{\mathfrak{s}(p)d}$ in the objective function. In other words, if $\omega_i^{pd}(t) - \beta_1 \phi_i^{\mathfrak{s}(p)d} < 0$, then $\gamma_i^{pd}(t) = 0$. This can be interpreted intuitively. $\phi_i^{\mathfrak{s}(p)d}$ represents the cost of shipping product p from fulfillment center i to node d. If $\phi_i^{\mathfrak{s}(p)d}$ is large, then fulfillment center i will only ship product p to node d if the number of open orders is large because a large backlog of orders will increase $\omega_i^{pd}(t)$.

3.3. Stability properties of $\pi^\star$

Our goal is to prove that $\pi^\star$ achieves maximum stability, and the key to these stability properties is Lemma 2.

Algorithm 1 Max-pressure control policy

for $t \in T$ **do**
  Solve problem (36) to obtain $\gamma_i^{pd\star}(t)$
  Using $\gamma_i^{pd\star}(t)$, fulfill orders according to equation (2)
  for $i \in \mathcal{N}$ **do**
    Solve problem (37) to obtain $y_{ij}^{sd\star}(t)$
    Using $y_{ij}^{sd\star}(t)$, move packages according to equation (3)
  end for
  for $i \in \mathcal{F}$ **do**
    Update inventory stock using equation (4)
  end for
end for

LEMMA 2. *When $\bar{\mathbf{r}} \in \hat{\mathcal{R}}^0$ and policy $\pi^\star$ is used, there exists a Lyapunov function $\mathcal{L}(t) \geq 0$ and constants $\kappa < \infty$ and $\epsilon > 0$ satisfying*

$$\mathbb{E}\left[\mathcal{L}(t+1) - \mathcal{L}(t) | \mathbf{z}(t)\right] \leq \kappa - \epsilon \left(\sum_{i \in \mathcal{N}} \sum_{s \in \mathcal{S}} \sum_{d \in \mathcal{N}} x_i^{pd}(t) + \sum_{p \in \mathcal{P}} \sum_{d \in \mathcal{N}} \chi^{pd}(t) \right) \tag{38}$$

The proof of Lemma 2 appears in Appendix A. Using Lemma 2, we now establish the maximum stability property of $\pi^\star$.

PROPOSITION 3. *If $\bar{\mathbf{r}} \in \hat{\mathcal{R}}^0$ and control policy $\pi^\star$ is used, then the network is stable.*

Proof. By Lemma 2, there exists $\mathcal{L}(t) \geq 0$, $\kappa < \infty$, and $\epsilon > 0$ such that

$$\mathbb{E}\left[\mathcal{L}(t+1) - \mathcal{L}(t) | \mathbf{z}(t)\right] \leq \kappa - \epsilon \left(\sum_{s} \sum_{d} \sum_{i} x_i^{sd}(t) + \sum_{p} \sum_{d} \chi_{(}^{pd} t) \right) \tag{39}$$

Therefore

$$\frac{1}{T} \sum_{t=1}^{T} \mathbb{E}\left[\mathcal{L}(t+1) - \mathcal{L}(t) | \mathbf{z}(t)\right] \leq \kappa - \frac{\epsilon}{T} \sum_{t=1}^{T} \mathbb{E}\left[\sum_{s \in \mathcal{S}} \sum_{d \in \mathcal{N}} \sum_{i \in \mathcal{N}} x_i^{sd}(t) + \sum_{p \in \mathcal{P}} \sum_{d \in \mathcal{N}} \chi^{pd}(t) \right] \tag{40}$$

Simplifying and rearranging,

$$\begin{aligned} \frac{1}{T} \sum_{t=1}^{T} \mathbb{E}\left[\sum_{s \in \mathcal{S}} \sum_{d \in \mathcal{N}} \sum_{i \in \mathcal{N}} x_i^{sd}(t) + \sum_{p \in \mathcal{P}} \sum_{d \in \mathcal{N}} \chi^{pd}(t) \right] &\leq \frac{\kappa}{\epsilon} + \frac{\mathbb{E}\left[\mathcal{L}(1) - \mathcal{L}(T+1)\right]}{\epsilon T} \\ &\leq \frac{\kappa}{\epsilon} + \frac{\mathbb{E}\left[\mathcal{L}(1)\right]}{\epsilon T} \end{aligned} \tag{41}$$

Taking the limit as $T \to \infty$,

$$\lim_{T\to\infty} \frac{1}{T} \sum_{t=1}^{T} \mathbb{E}\left[\sum_{s\in\mathcal{S}} \sum_{d\in\mathcal{N}} \sum_{i\in\mathcal{N}} x_i^{sd}(t) + \sum_{p\in\mathcal{P}} \sum_{d\in\mathcal{N}} \chi^{pd}(t) \right] \leq \frac{\kappa}{\epsilon} \tag{42}$$

which satisfies Definition 1 for stability. □

By combining Propositions 1 and 3, we now know that any $\bar{\mathbf{r}} \in \hat{\mathcal{R}}^0$ will be stabilized by $\pi^\star$, and any $\bar{\mathbf{r}} \notin \hat{\mathcal{R}}$ cannot be stabilized by any policy. Purchase order rates exactly on the boundary of $\hat{\mathcal{R}}$ is a highly unlikely scenario and will be ignored hereafter. Therefore, we can conclude that $\pi^\star$ serves any demand that can be served.

PROPOSITION 4. *Control policy $\pi^\star$ achieves maximum stability.*

Proof. If $\bar{\mathbf{r}} \in \hat{R}^0$ then the network is stable (Proposition 3), and if $\bar{\mathbf{r}} \notin \hat{\mathcal{R}}$ then the network cannot be stabilized by any control policy (Proposition 1). Therefore $\pi^\star$ stabilizes as much demand as possible. □

The corresponding corollary to Proposition 4 is that $\hat{\mathcal{R}} = \mathcal{R}$, meaning that the equations used to characterize $\hat{\mathcal{R}}$ can be used to describe the stable region itself.

PROPOSITION 5. *The characterization of the stable region, $\hat{\mathcal{R}}$, satisfies $\hat{\mathcal{R}} = \mathcal{R}$.*

Proof. Follows from Propositions 1 and 3. □

3.4. Approximation scheme for larger problems

Problems (36) and (37) are integer programs and are therefore NP-hard problems. Even with the best computers, this can easily become a practical issue when the number of variables is large. Unfortunately, $\mathcal{P}$, the set of products offered, is likely to be very large for realistic systems. We therefore briefly discuss using a polynomial-time approximation as an alternative to solving problems (36) and (37) exactly.

Problems (36) is a knapsack problem, and problem (37) is very similar to a knapsack problem. It would be a knapsack problem if not for constraints (11)–(12). Knapsack problems have polynomial-time approximation algorithms (e.g. Chekuri and Khanna 2005). In this section, we aim to establish that a polynomial-time approximation scheme can still achieve stability given the correct level of approximation. Formally, we define an α-approximation scheme as a policy $\hat{\pi}$ that achieves $\hat{\mathbf{y}}(t)$ and $\hat{\boldsymbol{\gamma}}(t)$ satisfying

$$\hat{\mathbf{y}}(t) \cdot \mathbf{w}(t) \geq (1-\alpha)\mathbf{y}^\star(t) \cdot \mathbf{w}(t) \tag{43}$$

and

$$\hat{\boldsymbol{\gamma}}(t) \cdot \boldsymbol{\omega}(t) \geq (1-\alpha)\boldsymbol{\gamma}^\star(t) \cdot \boldsymbol{\omega}(t) \tag{44}$$

LEMMA 3. *Let $\bar{\mathbf{r}} \in \hat{\mathcal{R}}^0$ and let ϵ be the smallest gap between constraints* (21)–(24)*. If we approximate $\pi^\star$ to a sufficiently small α satisfying*

$$\epsilon > \alpha \left(\max_{i \in \mathcal{N}} C_i \right) \tag{45}$$

then there exists a Lyapunov function $\mathcal{L}(t) \geq 0$ and constants $\kappa < \infty$ and $\epsilon' > 0$ satisfying

$$\mathbb{E}\left[\mathcal{L}(t+1) - \mathcal{L}(t) | \mathbf{z}(t)\right] \leq \kappa - \epsilon' \left(\sum_{i \in \mathcal{N}} \sum_{s \in \mathcal{S}} \sum_{d \in \mathcal{N}} x_i^{pd}(t) + \sum_{p \in \mathcal{P}} \sum_{d \in \mathcal{N}} \chi^{pd}(t) \right) \tag{46}$$

The proof of Lemma 3 appears in Appendix B. Using Lemma 3, we can prove a result similar to Proposition 3.

PROPOSITION 6. *Let $\bar{\mathbf{r}} \in \hat{\mathcal{R}}^0$ and let ϵ be the smallest gap between constraints* (21)–(24)*. If we approximate $\pi^\star$ to a sufficiently small α satisfying equation* (45) *then the network is stable.*

The proof of Proposition 6 is similar to the proof of Proposition 3, except using ϵ' instead of ϵ based on Lemma 3.

Condition (45) is fairly intuitive. The parameter ϵ represents the slack between the demand on the network and the available capacity. As that slack decreases, the accuracy of the approximation must correspondingly increase (causing α to decrease) to maintain stability. This creates a tradeoff between computational power and network capacity. If more excess capacity is built, then a less accurate approximation can be used while maintaining stability. Conversely, if less capacity is built, then a more accurate approximation is needed, necessitating more computational power to achieve a stabilizing real-time control.

4. Relationship to network topology design

Proposition 5 established that the equations defining $\hat{\mathcal{R}}$ also describe the stable region. Therefore, these equations can be useful for making network topology decisions relating to the average customer demand rates. Customer demand will likely vary throughout the year, but the max-pressure policy will adapt network operations to any demand in the stable region $\mathcal{R}$. If we can predict future customer demand rates to be $\tilde{\mathcal{R}}$, then we can design a network with a stable region $\mathcal{R}$ that includes $\tilde{\mathcal{R}}$, i.e. design for $\mathcal{R} \supseteq \tilde{\mathcal{R}}$. Policy $\pi^\star$ can then be used to manage the network in real-time in response to stochastic customer demand and inventory arrivals.

Let $\rho_i \in \{0, 1\}$ indicate whether node i (including fulfillment centers) is built. Then the stable region can be used to derive a mixed integer linear program to optimize the network topology.

Suppose a budget B exists to construct buildings, and b_i is the building cost associated with building node i. Consider the following program:

$$\begin{aligned}
\max \quad & \sum_{p\in\mathcal{P}}\sum_{d\in\mathcal{N}} \bar{r}^{pd} && && \text{(47a)}\\
\text{s.t.} \quad & \sum_{j\in\mathcal{N}}\sum_{s\in\mathcal{S}}\sum_{d\in\mathcal{N}} \bar{y}_{ij}^{sd} \le C_i\rho_i && \forall i\in\mathcal{N} && \text{(47b)}\\
& \sum_{i\in\mathcal{N}} b_i\rho_i \le B && \forall i\in\mathcal{N} && \text{(47c)}\\
& \rho_i \in \{0,1\} && \forall i\in\mathcal{N} && \text{(47d)}\\
& (17)\text{–}(24)
\end{aligned}$$

where $\bar{r}^{pd}$ is a decision variable representing the average amount of fulfilled demand. Objective function (47a) attempts to maximize the number of purchase orders that can be served. Other objectives can be considered here, such as the profit or revenue per purchase order or matching served purchase orders to demand. Constraint (47b) ensures that outbound flows from node i are 0 unless i is built, which also prevents fulfillment centers from fulfilling orders and placing them into the network. Constraint (47c) is the budget limitation on building nodes. The average flow rates must also satisfy the stable region equations (17)–(24) to be a valid network flow.

In addition to ρ_i, the other decision variables inherent in equations (17)–(24) are $\bar{\mathbf{y}}$, $\bar{\boldsymbol{\gamma}}$, and $\bar{\boldsymbol{\sigma}}$. $\bar{\mathbf{y}}$ and $\bar{\boldsymbol{\gamma}}$ describe the average operation of this network in terms of the average flows of packages through the network and the average number of packages shipped from each fulfillment center. $\bar{\boldsymbol{\sigma}}$ describes the average inventory requirements for each fulfillment center in terms of product arrivals for inventory.

Problem (47) is a facility design problem to optimize both facility site selection and average flows. Other constraints may be added to problem (47) to represent existing network topology and candidate locations, constraints on inventory deliveries, etc. Objective (47a) could be modified to more realistically represent customer demand growth and associated revenue. Since the purpose of this paper is to establish the maximum stability properties of $\pi^\star$, we will leave exploration of problem (47) to future work.

5. Numerical results

The purpose of these numerical results is to demonstrate numerically the favorable stability properties of the max-pressure control, and to study other performance metrics besides stability. We aim to demonstrate these numerical results on a network of considerable size, and we have created a network based on public Amazon facility location data. We simulated the movement of packages on a long time horizon and studied the performance metrics. Besides simulating the evolution of

conservation equations (1)–(4) in response to control decisions, we also tracked each individual order to obtain metrics such as transportation time and time before fulfillment. For the purposes of comparing performance metrics, we defined a greedy fulfillment policy. Although it is not optimal, we believe this greedy policy is reasonable and serves to demonstrate the value of the throughput properties.

5.1. Network characteristics

To create a network of significant size and geographic spread, our numerical results use a fascimile of the Amazon fulfillment network constructed using only publicly available data on building locations. This network is not intended to accurately represent the details of Amazon's system, and has been designed without using proprietary information. All data was taken from websites available to the public (e.g. Google Maps). Buildings were separated into fulfillment centers, sort centers, and delivery stations based on building categories posted on public websites by third parties. Building locations and their type are illustrated in Figure 2 to show the geographic spread. Note that at the scale of the illustration, some delivery stations are hidden as they are overlapped by fulfillment centers and/or sort centers. In total, the network includes 60 fulfillment centers, 43 sort centers, and 136 delivery stations. This network may not perfectly represent Amazon's current network because the data used may be out of date or incorrect. However, the map in Figure 2 shows that the building locations are clustered in major population centers and represents a reasonable online fulfillment network design. The graph is completely connected as described by equations (1)–(3). Fulfillment centers are not connected to each other. Each fulfillment center is connected to all sort centers and delivery stations. Each sort center is connected to all other sort centers and delivery stations. Delivery stations do not have outbound connections to fulfillment centers or sort centers.

Distances between locations were computed using the great circle distance, and travel times were based on an average speed of 60mi/hr. In reality, the road topology would affect distances and travel speeds. Furthermore, some items could be shipped using alternative modes, such as air transportation. However, we believe that this simple calculation of travel times is sufficient for a numerical demonstration. In equation (3), travel times between nodes are assumed to be one time step. This will not hold in reality, so virtual "nodes" are created where needed to increase the travel times. For instance, if the travel time from i to j is 4 time steps, than 3 virtual nodes are created in-between i and j to cause a travel delay. These nodes do not have capacity or connection constraints, and are only have connections in series to facilitate movement from i to j. Their only available action for these nodes is to move packages along from the incoming link to the outgoing link. Optimization problem (37) has a fixed action for virtual nodes and therefore does not require a solver.

Customers were aggregated into the centroid of US 5-digit zipcodes. To reduce the problem size sufficiently to run the simulation on the desktop computer available to the authors, only the 1000 most populated zipcodes were used for customer locations, and the rest were ignored. Destination indices d were the 5-digit zip codes. Delivery stations could serve any zip code that is within 1 time step of travel time from the delivery station. For each scenario, the total customer demand, total replenishment supply, and node-specific capacities were inputs to the model. Each node was assumed to have the same capacities for simplicity. The total customer demand was randomly allocated to individual zipcodes, weighted by population. The demand and replenishment supply were further allocated randomly (uniformly distributed) to individual products. The total demand and supply, and the individual values per product and per destination are randomly generated and not based on Amazon data. The number of products is relatively low to enable running multiple simulations on a desktop computer. In reality, the number of distinct products shipped by an online fulfillment system is much larger.

All of the aforementioned characteristics are listed as rates, such as the average demand per unit time $\bar{\mathbf{r}}$ and average replenishment rate $\bar{\sigma}$. The number of virtual nodes depend on the travel times. The values per time step depend on the length of one time step. For these simulations, we used a time step of 1hr and simulated a time horizon of 30 days consisting of 24 time steps per day. This time step ignores any batching that occurs within the hour, such as trucks departing on some set schedule instead of hourly. In reality, nighttime hours may have less activity in terms of customer orders, warehouse activity, and package movement, but we believe that this setup is sufficient to demonstrate the performance. Furthermore, we assume that random variables are indentically distributed with respect to time, and a nighttime adjustment might violate this assumption.

We consider 20 different product types ($|\mathcal{P}| = 20$) and 2 different product sizes ($|\mathcal{S}| = 2$), with half of the products allocated to each size. Realistic systems will have many more products. However, we restricted the problem size to reduce the computation times for simulations consisting of 720 time steps. The numerical results are sufficient to demonstrate the benefits of stability, and the low computation times per integer program solution shows that larger problems could be solved with this approach in real-time.

5.2. Greedy comparison policies π^{gdy} and π^{gdy2}

The system we study in this paper can be modeled as a Markov decision process. However, the number of dimensions make it impossible to solve using standard algorithms like value iteration. Therefore, we cannot compare the max-pressure policy against an optimal solution that minimizes shipping costs or customer delivery times. For the purposes of comparing these metrics, we define alternative greedy policies π^{gdy} and π^{gdy2} that attempt to ship products along the minimum-time path.

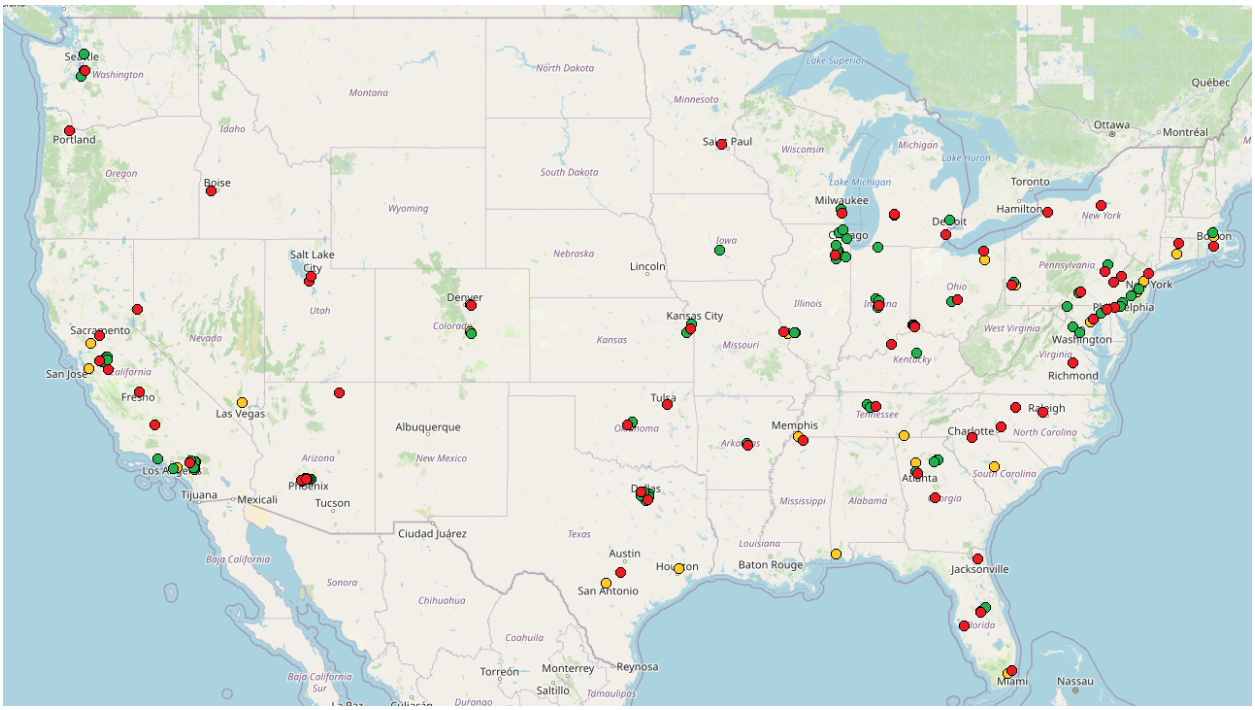


Figure 2 **Map of nodes used for the fulfillment network. Red circles indicate fulfillment centers, yellow circles indicate sort centers, and green circles indicate delivery stations. At this scale, some facilities overlap.**

The π^{gdy} and π^{gdy2} policies can be described as follows, for each time step t. For each product p and destination d, we try to fulfill open orders $\chi^{pd}(t)$ at the fulfillment center with the shortest travel time to d that has available inventory of p and available capacity. If no such fulfillment center is found, then the order is not fulfilled at time step t. At each node i, we try to move packages $x_i^{sd}(t)$ towards their destination along the shortest-time path. For each package of size s destined for d, move it to the node j on the shortest-time path to d from i. If capacity or other constraints for the immediate downstream link would be violated, packages are not moved and held at i instead. The greedy algorithms do not look farther downstream to identify capacity or other constraints that might be violated in the future.

For π^{gdy}, we use 1 as the link travel time as defined by equation (1). (Longer transportation trips are separated into smaller segments with travel time 1 time step, each represented by a separate link.) For π^{gdy2}, the link length ℓ_{ij} for shortest path calculations is

$$\ell_{ij} = 1 + \left\lfloor \frac{\sum_{s\in\mathcal{S}} \sum_{d\in\mathcal{N}} x_j^{sd}(t)}{C_j} \right\rfloor \tag{48}$$

The link travel time is 1 time step as defined by equation (1). $\sum_{s\in\mathcal{S}} \sum_{d\in\mathcal{N}} x_j^{sd}(t)$ is the total backlog for node j. Since C_j is the capacity of node j, we anticipate that backlog requiring $\lfloor (\sum_{s\in\mathcal{S}} \sum_{d\in\mathcal{N}} x_j^{sd}(t))/C_j \rfloor$ time steps to clear. For example, a backlog of $\sum_{s\in\mathcal{S}} \sum_{d\in\mathcal{N}} x_j^{sd}(t) = 2.5C_i$ could be cleared in 3 time steps of shipping C_j packages (operating at capacity C_j). If we added

1 more package to that backlog, it could still be processed and shipped to the next node in the 3rd time step. Sending a package through a node requires a minimum of 1 time step by equation (1), so we would need to add 2 time steps to estimate the backlog delay. The link cost of $1 + \lfloor 2.5C_j/C_j \rfloor = 3$ represents that 3 time step processing time. For completeness, this policy is formally described in Appendix C. The resulting solution respects all constraints and attempts to reduce the travel time associated with delivery. However, it does not consider network congestion, and the MP control should perform better when the network is saturated with orders and packages. In particular, the order in which orders and packages are considered is arbitrary and could lead to suboptimal prioritization for fulfillment or shipping.

5.3. Stability

We first demonstrate the value of the stability properties. We create a scenario in which each of the 60 fulfillment centers has a capacity of 100 orders per time step, for a total of 6000 orders per time step. We randomly distributed an average demand of 5455 orders per time step among the 20 products and the 1000 destination nodes. The number 5455 represents a ϵ of 10% compared with the total capacity. However, due to the random distribution of individual product demand and the weighted distribution of demand towards major population centers, the geographic spread of the demand is highly non-uniform. We also provided an ϵ of 10% in average replenishment $\bar{\sigma}_i^p$, satisfying $\sum_{i\in\mathcal{F}} \bar{\sigma}_i^p = (1+\epsilon)\sum_{d\in\mathcal{N}} \bar{r}^{pd}$. Recall that ϵ is the difference between average supply and average demand in the distribution network; some buffer is needed so that stochastic demand (that sometimes exceeds supply) can still be served. However, stochasticity in the distribution of product replenishment to fulfillment centers means that product availability was often geographically separated from product demand. We used $\beta_1 = \beta_2 = 0.05$ for these results. Inventory was initially 0, which is unrealistic, but any fulfillment delays caused by 0 initial inventory should be disappear in the average over the long time horizon.

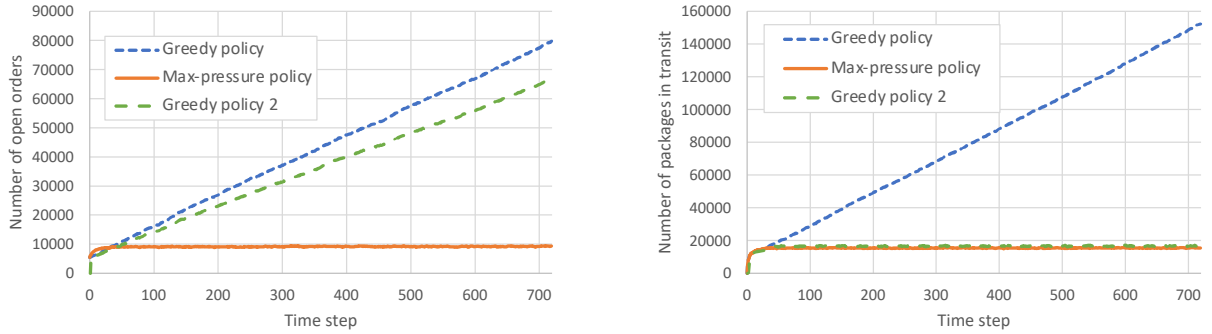
Each of the 43 sort centers was given a capacity of 400 packages per time step. Since some packages can be moved directly from fulfillment centers to their destination (bypassing sort centers), this capacity is sufficient for network stability, and indeed the $\pi^\star$ policy achieves stability. However, sort centers are spread out geographically, and due to asymmetric customer demand, some sort centers are closer to customer destinations than others. π^{gdy} routed an average of 3640 packages per hour through the sort centers, compared with the total capacity of 17,200 packages per hour across 43 sort centers. Therefore, the greedy policy π^{gdy}, which uses the shortest path for package deliveries, will be constrained by sort center capacity whereas $\pi^\star$ will adapt. Note that although the sort centers in this simulation are highly underutilized on average, there are some individual sort centers that are still bottlenecks when sending packages on the shortest path (ignoring queue lengths).

Figure 3a plots the number of unfulfilled orders, $\sum_{p\in\mathcal{P}}\sum_{d\in\mathcal{N}}\chi^{pd}(t)$, over time for both the max-pressure and greedy policies. The same random seed was used so that orders and replenishment were the same in every simulation. For $\pi^\star$, the number of unfulfilled orders increases initially, then remains roughly constant for the remainder of the simulation. Due to the scale of the graph, the fluctuations in the number of unfulfilled orders is difficult to see. However, it fluctuates with an average of 9,110 open orders and a standard deviation of 310. This is characteristic of a stable network per Definition 1: the average number of unfulfilled orders remains bounded. This boundedness means that on average, orders are being fulfilled at the same rate at which they arrive.

However, the number of unfulfilled orders continues to increase over the simulation horizon for π^{gdy} and π^{gdy2}. This demonstrates an unstable network; the system cannot keep up with the rate of customer orders, and the number of open orders grows without bound. π^{gdy} attempts to fulfill orders from the fulfillment center closest to the destination with available inventory and capacity. This causes an inefficient use of capacity: fulfillment centers that are severely capacity-constrained are used to fulfill orders that could have been fulfilled elsewhere. Meanwhile, some unfulfilled orders cannot be fulfilled elsewhere due to lack of inventory, which causes the orders to go unfulfilled. Although π^{gdy2} considers delays caused by queue backlogs in its choice of fulfillment centers, it still fails to achieve stable order fulfillment. It could be possible to improve the performance of the greedy policies by changing the inventory placement. Nevertheless, these results demonstrate how $\pi^\star$ adapts to the given inventory placement in its control decisions.

Figure 3b shows the number of packages in transit. Like the number of open orders, the number of packages in transit remained bounded for $\pi^\star$. It fluctuated with an average of 15,309 and a standard deviation of 874. However, for π^{gdy}, the number of packages in transit increased over time. This indicates that the rate of packages being shipped by the fulfillment centers exceeded the rate of deliveries. Since $\pi^\star$ maintained stability, this demonstrates that π^{gdy} inefficiently used network capacity. The implications for the network are clear. Some nodes have an ever-growing backlog of packages waiting to be shipped onwards, resulting in unbounded delivery times for certain destinations. It appears that π^{gdy2} is able to maintain stability in the number of packages in transit. This may be because ℓ_{ij} is a function of queue lengths, and $\pi^\star$ also routes based on queue lengths. However, the shipping delay of the new greedy policy is much higher than the shipping delay from the max-pressure policy as will be seen in Table 1b.

We also compare the number of orders fulfilled and packages delivered by each policy, and the differences are not large. The number of orders fulfilled is $\sum_{t=1}^{T}\sum_{d\in\mathcal{N}}\sum_{p\in\mathcal{P}}\sum_{i\in\mathcal{F}}\gamma_i^{pd}(t)$, and the number of packages delivered is $\sum_{t=1}^{T}\sum_{d\in\mathcal{N}}\sum_{s\in\mathcal{S}}\sum_{i\in\mathcal{N}}y_{id}^{sd}(t)$. The specific numbers are given in Table 1c. $\pi^\star$ fulfilled around 1.83% more orders than π^{gdy}, and delivered around 5.82% more packages. However, that 1.83% of fulfilled orders is 70,427 orders, which entirely accounts for the



(a) Unfulfilled orders (b) Packages in transit

Figure 3 Performance of max-pressure policy $\pi^\star$ and greedy policy π^{gdy} over the simulation horizon.

difference in orders fulfilled shown in Figures 3a. Similarly, that 5.82% of packages is 213,570 packages. These results indicate that small differences in delivery performance do not tell the entire story. π^{gdy} has significant problems fulfilling and delivering orders of specific product types to specific destinations shown in Figures 3a and 3b. Those problems are corrected by ensuring stability.

5.4. Comparison of performance metrics

We now compare a larger range of scenarios than those discussed in Section 5.3. We solved each scenario using $\pi^\star$ and π^{gdy} to operate the network. The specific realizations of demand and inventory replenishment were identical for both policies due to fixing the random seed. The supply-side network characteristics (node capacities and inventory replenishment) are the same as those in Section 5.3, but we vary the total demand. The variation in these scenarios is a changing gap between the theoretical network capabilities described by the stable region boundaries and the demands placed on the network by customer orders. The demands considered are determined by the equation

$$\sum_{d\in\mathcal{N}} \bar{r}^{pd}\cdot(1+\epsilon) = \sum_{i\in\mathcal{F}} \bar{\sigma}_i^p \qquad \forall p\in\mathcal{P} \tag{49}$$

with $\sum_{p\in\mathcal{P}}\sum_{i\in\mathcal{F}}\bar{\sigma}_i^p = 6000$ units per time step. Our scenarios include $\epsilon\in[0.1, 0.35]$ at increments of 0.05.

Table 1a shows the number of open orders and packages in transit at the end of the simulation, as well as indicating whether the network was stable or unstable. Stability was checked manually; Figure 3 illustrates the difference between stable and unstable networks. Stability is indicated separately for open orders, $\boldsymbol{\chi}(t)$, and for packages in transit, $\mathbf{x}(t)$. It is possible for fulfillment to be stable while package delivery is not, and vice versa. Indeed that is observed for π^{gdy} and π^{gdy2} in some of the scenarios. Overall, $\pi^\star$ stabilizes order fulfillment for significantly larger demands

than π^{gdy} or π^{gdy2}, as is expected. The difference in stability can be observed in the difference between the number of fulfilled orders and the number of orders delivered at the end of the scenario. Instability is also visible in the large numbers of open orders and/or undelivered packages at the end of the simulation. These numbers continue to increase throughout the simulation due to instability. However, for the 4800 demand per time step scenario, π^{gdy} has fewer open orders at the end yet is still unstable; stability is checked based on whether the number of open orders is increasing over time or remains bounded. Table 1c compares the total number of orders fulfilled (converted into a package to be shipped) and delivered. The difference between the max-pressure and greedy policies is not large, relative to the number of orders, but is large enough to make a difference in stability.

Table 1b compares service metrics. These include the average time spent between the order being placed and a corresponding package being shipped from a fulfillment center, τ_{f}, and the average time the package spent in transport, τ_{t}. We refer to τ_{f} as the average fulfillment time and the τ_{t} as the average transportation time. To calculate τ_{f}, we track individual orders in the system, including the time they were created, fulfilled, and delivered. τ_{f} is calculated as the average difference between the fulfillment time and creation time per order. τ_{t} is the average difference between the delivery time and the fulfillment time per order. Orders and shipments were moved in priority of creation time, so earlier orders would be fulfilled or delivered first, given the opportunity. τ_{f} and τ_{t} only count completed orders, so it is important to look at stability as well when interpreting these values.

The number of open (unfulfilled) orders and undelivered packages at the end of the simulation are also recorded. Using Little's Law, the average fulfillment time τ_{f} is related to the average number of orders via

$$\tau_{\text{f}} \cdot \bar{r}^{pd} = \mathbb{E}\left[\chi^{pd}(t)\right] \qquad \forall d \in \mathcal{N}, \forall p \in \mathcal{P} \tag{50}$$

Similarly, the average transportation time τ_{t} is related to the average number of packages via

$$\tau_{\text{t}} \cdot \sum_{i \in \mathcal{N}} \mathbb{E}\left[y_{id}^{sd}(t)\right] = \sum_{i \in \mathcal{N}} \mathbb{E}\left[x_i^{sd}(t)\right] \qquad \forall d \in \mathcal{N}, \forall s \in \mathcal{S} \tag{51}$$

Equations (50)–(51) are useful conceptually because they relate stability, measured by the boundedness of $\mathbb{E}[\chi^{pd}(t)]$ and $\mathbb{E}[x_i^{sd}(t)]$, with τ_{f} and τ_{t}. If the network is stable, then we know that τ_{f} and τ_{t} will be bounded (although their values could still be higher than desired). On the other hand, if the network is unstable, meaning that $\mathbb{E}[\chi^{pd}(t)]$ and/or $\mathbb{E}[x_i^{sd}(t)]$ are unbounded, we expect τ_{f} and/or τ_{t} to be unbounded as well. Therefore, for unstable networks, the numbers given in Table 1b would increase if the time horizon of the simulation was extended. Table 1 indicates whether each scenario was stable or not.

The difference between average fulfillment and transportation times for π^{gdy}, π^{gdy2}, and $\pi^\star$ is small, and π^{gdy} and π^{gdy2} actually have better service times than $\pi^\star$ when they stabilize the network. The benefits of $\pi^\star$ are not readily apparent from these metrics alone. Part of the reason for the similarities is that many orders can be fulfilled at the next time step, depending on inventory availability, and many packages can be delivered quickly by being fulfilled at a fulfillment center near the destination. Given the large volume of orders being handled (see the small differences between total orders delivered in Table 1c) the average values for τ_{f} and τ_{t} are quite small for both policies. The average time in system for unfulfilled orders and undelivered packages are more distinctive metrics, and are also listed in Table 1b. For unstable networks (see Table 1a), these average times start becoming quite large. They indicate that although fulfilled orders and delivered packages are, on average, completed rather quickly, the backlog of unfulfilled orders and undelivered packages has existed for many time steps and is a major issue for customer service.

Overall, Table 1 suggests that π^{gdy} and π^{gdy2} perform slightly better than $\pi^\star$ when they can stabilize the network. However, $\pi^\star$ is a new policy, and modifications to it or its parameters (such as β) could still improve its performance metrics. The main advantage of $\pi^\star$ is that it can stabilize the network under higher demands than π^{gdy} and π^{gdy2}. π^{gdy} requires an ϵ of around 35% excess capacity in $\mathcal{R}$ to achieve stability in both order fulfillment and package delivery. π^{gdy2} appears to stabilize package delivery fairly well but requires an ϵ of around 20% to stabilize order fulfillment. In contrast, the results show that $\pi^\star$ is stable up to an ϵ of 10%, and mathematically should be stable for any $\epsilon > 0$ (although that may be difficult to observe without a very long simulation time).

Table 1 Comparison of performance metrics for different demand scenarios

(a) Stability metrics

Total demand per time step	Open orders at end			Undelivered packages at end			Stable fulfillment?			Stable delivery?		
	$\pi^\star$	π^{gdy}	π^{gdy2}	$\pi^\star$	π^{gdy}	π^{gdy2}	$\pi^\star$	π^{gdy}	π^{gdy2}	$\pi^\star$	π^{gdy}	π^{gdy2}
4444	6,465	4,483	4,483	12,085	11,400	11,400	✓	✓	✓	✓	✓	✓
4615	7,210	4,643	4,643	12,570	14,198	12,291	✓	✓	✓	✓	no	✓
4800	7,781	7,051	6062	13,107	30,269	13,667	✓	no	✓	✓	no	✓
5000	8,196	25,319	18,436	13,700	66,531	15059	✓	no	no	✓	no	✓
5217	8,619	48,485	39174	14,429	107,772	15772	✓	no	no	✓	no	✓
5555	9,337	79,764	67,020	15,321	152,306	16,338	✓	no	no	✓	no	✓

(b) Service metrics

Total demand per time step	Avg. fulfillment time (Δt)			Avg. transport time (Δt)			Fulfillment delay at end (Δt)			Delivery delay at end (Δt)		
	$\pi^\star$	π^{gdy}	π^{gdy2}	$\pi^\star$	π^{gdy}	π^{gdy2}	$\pi^\star$	π^{gdy}	π^{gdy2}	$\pi^\star$	π^{gdy}	π^{gdy2}
4444	1.42	1.00	1.00	3.74	3.61	3.69	6.23	1.00	1.00	1.92	1.84	1.91
4615	1.51	1.00	1.00	3.75	3.67	3.69	7.47	1.00	1.00	1.92	38.09	2.02
4800	1.58	1.06	1.05	3.76	3.78	3.82	8.59	97.50	63.94	1.94	184.31	13.46
5000	1.60	1.12	1.11	3.78	3.83	3.90	8.51	270.36	1.96	236.43	265.45	29.01
5217	1.60	1.20	1.13	3.81	3.82	3.92	8.61	311.03	300.84	1.98	297.01	34.64
5555	1.66	1.21	1.20	3.83	3.84	3.96	8.84	329.27	322.43	2.07	311.44	31.44

Units are Δt — 1 time step

(c) Performance

Total demand per time step	Total orders fulfilled			Total orders delivered		
	$\pi^\star$	π^{gdy}	π^{gdy2}	$\pi^\star$	π^{gdy}	π^{gdy2}
4444	3.19E6 (−0.06%)	3.19E6	3.19E6 (0.00%)	3.17E6 (+0.01%)	3.17E6	3.17E6 (−0.01%) 3
4615	3.31E6 (−0.08%)	3.31E6	3.31E6 (0.00%)	3.29E6 (+0.09%)	3.29E6	3.29E6 (+0.06%)
4800	3.44E6 (−0.02%)	3.44E6	3.44E6 (+0.03%)	3.42E6 (+0.63%)	3.40E6	3.42E6 (+0.52%)
5000	3.58E6 (+0.48%)	3.57E6	3.57E6 (+0.19%)	3.56E6 (+2.17%)	3.48E6	3.54E6 (+1.67%)
5217	3.74E6 (+1.08%)	3.70E6	3.71E6 (+0.25%)	3.71E6 (+3.90%)	3.57E6	3.68E6 (+2.83%)
5555	3.91E6 (+1.83%)	3.84E6	3.85E6 (+0.33%)	3.88E6 (+5.82%)	3.67E6	3.82E6 (+4.06%)

% change is relative to π^{gdy}

Table 2 Computation time

Total demand per time step	Origin CPU time (s)			Node CPU time (s)		
	$\pi^\star$	π^{gdy}	π^{gdy2}	$\pi^\star$	π^{gdy}	π^{gdy2}
4444	1.73	0.99	2.24	8.65E-3	1.05E-3	1.06E-3
4615	1.81	1.02	2.28	8.05E-3	1.03E-3	1.07E-3
4800	2.02	1.09	2.33	7.61E-3	1.04E-3	1.06E-3
5000	2.18	1.16	2.39	7.21E-3	1.04E-3	1.06E-3
5217	2.38	1.23	2.46	6.62E-3	1.06E-3	1.03E-3
5555	2.77	1.32	2.60	6.14E-3	1.01E-3	1.05E-3

5.5. Computation time

The network model was implemented in Java using IBM's CPLEX 22.1 to solve integer programs. The model was solved on a Windows 11 desktop with an Intel i7-12700K processor at 3.60GHz and 32GB of memory. Problem (36) is solved once per timestep, and problem (37) is solved per timestep and per node. Each problem was solved to optimality. We report the average computation times required per solution. Each simulation included 720 time steps, and additional computation time was spent on tracking the movement of individual packages. Therefore, each simulation of $\pi^\star$ required 2–3 hours to complete.

Table 2 reports the average computation times per problem (36) and problem (37). π^{gdy} had significantly lower computation times than $\pi^\star$. However, the difference was less than expected. For problem (36), π^{gdy2} was sometimes slower than $\pi^\star$, which is surprising because π^{gdy2} is solving a polynomial-time greedy algorithm whereas $\pi^\star$ is using CPLEX to solve a knapsack problem to optimality. However, much of the computation time of π^{gdy2} comes from recalculating the shortest path to each destination each time step since ℓ_{ij} varies over time via equation (48). The computation time of π^{gdy2} will scale better with $|\mathcal{P}|$ than $\pi^\star$. If $|\mathcal{P}|$ is large enough, then $\pi^\star$ should require more computation time than π^{gdy}. This comparison suggests that solving problem (36) or an approximation (Section 3.4) may not require much more computational effort than the alternatives. For problem (37), π^{gdy} was 1 order of magnitude faster, but $\pi^\star$ took only around 0.01s anyways. Each time step was intended to represent 1hr of network operations, so the problems were easily solved in "real-time" (computation time of seconds per 1 hr time step), but practical systems will have a much larger scale. Recall that each instance of problems (36) and (37) can be solved on separate computers due to Proposition 2. The computation time of problem (36) appears to be the limiting factor for real-time solutions.

5.6. Effects of β on performance

In Table 1b, we observe that $\pi^\star$ consistently has a higher fulfillment time than π^{gdy}. Although average fulfillment times of 1.66 time steps are usually acceptable, we may be able to improve those fulfillment times by modifying β_1. We suspect that $\pi^\star$ is primarily fulfilling orders from fulfillment

centers near the destination due to the value of β_1 in objective (36a). Therefore, we experiment with β_1 to encourage faster fulfillment from fulfillment centers farther away. A smaller value of β_1 will result in a smaller effect of travel times on objective (36a). For the experiments in Section 5.4, $\beta_1 = \beta_2 = 0.05$. Recall that β_1 is comparing weights of $\omega_i^{pd}(t)$ (in units of numbers of orders) and travel times of $\phi_i^{sd}(t)$ (in units of time steps), so small values are reasonable.

Table 3 compares the effects of varying β_1 on stability and service metrics. The impacts on fulfillment times are immediately noticeable. For instance, when $\beta_1 = 0.01$, the average fulfillment time is 1.00s (similar to the greedy algorithm) whereas when $\beta_1 = 0.05$ the average fulfillment time is 1.66. For all β scenarios, the network remained stable, which is aligned with the predictions of Proposition 3. However, the average number of open orders varied. With $\beta_1 = 0.01$, the number of open orders was 5,445 on average with a standard deviation of 71, and the average fulfillment delay at the end was 1.00 indicating that most orders were fulfilled at the next time step. With $\beta_1 = 0.05$, orders were not fulfilled as quickly, and the average waiting time for open orders at the end of the simulation was much higher at 8.84 time steps. Meanwhile, the average transportation time did not change much. These results do not show any disadvantages to using a small value of β_1. However, in networks with different topology, fulfillment from nearby fulfillment centers, which is prioritized via larger values of β_1, may be more valuable for reducing the transportation time.

We also investigated how variations in β_2 affects performance by varying β_2 from 0.01 to 0.09. However, no significant variations in average fulfillment time or transportation time were observed. This could be due to the network topology, such as overlapping geographical coverage so that packages have multiple routes to their destination with similar travel times. Another possible cause is in how package routing affects transportation times. In problem (37), a lower value of β_2 will increase the weight for packages to be routed along longer paths. When β_2 is higher, packages may have a stronger preference for the shortest path, but the result may be that packages are delayed until capacity is available or $x_{ij}^{sd}(t)$ is sufficiently large to route them on the shortest path. Those delays will also appear to be higher transportation times. In other words, total transportation times on this network may simply not be very sensitive to β_2.

Average fulfillment times (including order fulfillment and transportation) were fairly low despite the geographic size of the network. There are several reasons for this. Last-mile delivery takes considerable time, but was not modeled here. Instead, the last-mile delivery station was used as the destination for this distribution network. Furthermore, fulfillment centers and sort centers were located in major population centers (based on Amazon's facility locations) and the transportation distances were small. Inventory of each product was distributed across the network which reduced the requirements for long-distance shipping. Handling times at each facility were a minimum of 1 time step. Also, the experimental setup ignores some delay due to batching purchase orders, product fulfillment, and package shipping actions over 1 hour time steps.

Table 3 Comparison of performance metrics for varying β_1. The demand is 5555 units per time step.

(a) Stability metrics

β_1	β_2	Open orders at end	Undelivered packages at end	Stable fulfillment?	Stable delivery?
0.01	0.05	5,486	15,246	✓	✓
0.03	0.05	5,933	15,180	✓	✓
0.05	0.05	9,337	15,321	✓	✓
0.07	0.05	12,638	15,464	✓	✓
0.09	0.05	14,553	15,416	✓	✓

(b) Service metrics

β_1	β_2	Avg. fulfillment time (Δt)	Avg. transport time (Δt)	Fulfillment delay at end (Δt)	Delivery delay at end (Δt)
0.01	0.05	1.00	3.82	1.00	1.98
0.03	0.05	1.07	3.82	1.88	1.98
0.05	0.05	1.66	3.83	8.84	2.07
0.07	0.05	2.20	3.83	14.49	2.22
0.09	0.05	2.53	3.82	17.93	2.21

Units are Δt — 1 time step

6. Conclusions

We developed a real-time control policy that is proven to maximize stability. In other words, the policy will serve all customer orders as long as the average rate of orders is within the stable region, which is the superset of the demand rates that can be served by all control policies. Therefore, our policy serves as much demand as any other policy, and achieves maximum throughput for any demand rates in the stable region. In the process, we designed a Markov model to describe the state evolution of the order fulfillment network over time, including the number of open orders, packages in transit, and inventory at each fulfillment center. We also characterized the stable region, or the set of demands that can be served by a given fulfillment network topology. Numerical results demonstrated that alternative policies achieve less stability than the max-pressure policy, and stability is beneficial for order fulfillment and other performance metrics.

Stability is important because failing to serve all customer orders will hurt customer service metrics. However, many other network performance metrics are not directly addressed by stability. These metrics include time for order fulfillment and delivery to customers, and labor and shipping costs for the operator. The max-pressure control does not attempt to achieve the best performance on these other metrics; it only guarantees stability. Additions such as ϕ_i^{sd} to guide packages along the minimum-cost path could help improve performance on other metrics, but it is not clear how close it will be to optimality. After varying β_1 and β_2 in Section 5.6, we obtained service metrics similar to those of the greedy π^{gdy} policy. However, it is not clear how the max-pressure control would perform at larger scale or on other networks. Prior literature indicates that max-pressure control can suffer from long queue backlogs or delays in package movement (Walton 2014, Neely 2022). It may be possible to avoid those by using a fixed-route formulation instead of a dynamic route choice at each node. That would require formulating the logistics network differently, but may improve performance. Due to the complexity in optimally controlling a network under stochasticity, we believe that this tradeoff may be worth it. However, more numerical results using practical systems are necessary to investigate the tradeoff. Alternatively, max-pressure optimizes other performance metrics in certain settings (Stolyar 2004, Ganti, Modiano, and Tsitsiklis 2007), and it may be possible to create a variant of max-pressure control or a different distribution network formulation that admits such performance guarantees on average delay or other metrics.

In the process of proving stability, we gave equations that characterize the stable region, or the set of demands that can be served by any control policy. These equations could be useful for network topology design as discussed in Section 4. This creates a nice connection between topology design and real-time control. We can design the network topology around the average demand rates and inventory replenishment that define the stable region, knowing that the real-time control will adapt to stochasticity and maintain stability whenever the average demand is within the stable

region. However, due to the potentially large number of product indices, the number of variables is probably too large for problem (47) to be solved in practice. Future work should investigate methods of aggregating indices while retaining an accurate characterization of the stable region.

This paper attempts the first use of max-pressure control to achieve maximum-throughput control of a logistics network, and as such there are many open questions that ought to be resolved. Max-pressure is a myopic policy that is suitable for computers or routers, but it lacks planning and predictability which may be cost-effective for humans operating warehouses and package transportation. It is not clear whether the control in this paper is acceptable for practical use. The inventory replenishment model might also be deterministic, or a control variable itself. Rather than a location-based replenishment $\sigma_i^p(t)$, the total replenishment over the network could be allocated to individual nodes based on the inventory storage limits and demand. That would require major model changes and a new max-pressure control. Other constraints such as the limitation on the number of trucks could be adapted to better model practical constraints.

The number of distinct products offered in the simulated system is much smaller than order fulfillment systems used in practice. Therefore, it is not clear how problem (36) will scale to systems of practical size. Integer programs are NP-hard, and the computation times of around 5s will increase exponentially with the number of products. Some options exist to apply these methods to larger problems. Using a smaller time step will reduce the number of new orders and new inventory deliveries per time step, resulting in $\gamma_i^{pd}(t) = 0$ for many indices due to $v_i^p(t) = 0$ or $\chi^{pd}(t) = 0$. We discussed the potential of using polynomial-time approximation algorithms to achieve stability in Section 3.4, and found that an intuitive tradeoff exists between the accuracy of the algorithm and the excess capacity of the network. Further experiments are needed to study how these approximations would perform on large problems in terms of stability and other performance metrics.

Acknowledgments

The authors gratefully acknowledge the support of the National Science Foundation, grant no. 1935514.

Appendix A: Proof of Lemma 2

Before we prove Lemma 2, we prove Lemmas 4 and 5 which compare the average control $(\bar{\boldsymbol{\gamma}}, \bar{\mathbf{y}})$ to $\pi^\star$.

LEMMA 4. *When $\bar{\mathbf{r}} \in \hat{\mathcal{R}}^0$, there exists constants $\kappa < \infty$ and $\epsilon > 0$ such that*

$$\sum_{s\in\mathcal{S}} \sum_{d\in\mathcal{N}} \sum_{(i,j)\in\mathcal{A}} \left(\bar{y}_{ij}^{sd} - y_{ij}^{sd\star}(t)\right) w_{ij}^{sd}(t) \le \kappa - \epsilon|\mathbf{w}(t)| \tag{A.1}$$

Proof. Let $\tilde{\mathbf{y}}(t) = \arg\max_{\mathbf{y}(t)} \{\mathbf{y}(t) \cdot \mathbf{w}(t)\}$ subject to constraints (9)–(10), i.e. the supply constraints without the maximum number of connections or the current state of purchase orders. Since $\bar{\mathbf{y}}$ is derived from

$\bar{\mathbf{r}} \in \hat{\mathcal{R}}^0$, constraints (22)–(24) hold with strict inequality. Therefore, for all $i \in \mathcal{N}$, there exists an $\epsilon > 0$ such that for the $j \in \Gamma_i^+$, $s \in \mathcal{S}$, $d \in \mathcal{N}$ with $\max w_{ij}^{sd}(t)$,

$$\sum_{s\in\mathcal{S}} \sum_{d\in\mathcal{N}} \bar{y}_{ij}^{sd} \leq C_{ij} - \epsilon \tag{A.2}$$

or

$$\sum_{s\in\mathcal{S}} \sum_{d\in\mathcal{N}} \bar{y}_{ij}^{sd} \times s \leq S_{ij} - \epsilon \tag{A.3}$$

If there are multiple $j \in \Gamma_i^+$, $s \in \mathcal{S}$, $d \in \mathcal{N}$ that attain the same value $\max w_{ij}^{sd}(t)$, then

$$\sum_{s\in\mathcal{S}} \sum_{d\in\mathcal{N}} \sum_{j\in\Gamma_i^+} \bar{y}_{ij}^{sd} \leq C_i - \epsilon \tag{A.4}$$

will provide the ϵ gap instead. Because of equations (A.2)–(A.4), for all i

$$\sum_{s\in\mathcal{S}} \sum_{d\in\mathcal{N}} \sum_{j\in\Gamma_i^+} \left(\bar{y}_{ij}^{sd} - \tilde{y}_{ij}^{sd}(t)\right) w_{ij}^{sd}(t) \leq -\epsilon \left(\max_{j\in\Gamma_i^+, s\in\mathcal{S}, d\in\mathcal{N}} w_{ij}^{sd}(t) \right) \tag{A.5}$$

$$\leq -\frac{\epsilon}{|\Gamma_i^+||\mathcal{S}||\mathcal{N}|} \sum_{s\in\mathcal{S}} \sum_{d\in\mathcal{N}} \sum_{j\in\Gamma_i^+} \left| w_{ij}^{sd}(t) \right| \tag{A.6}$$

Also, the difference between $\tilde{\mathbf{y}}(t)$ and $\mathbf{y}^\star(t)$ is bounded. If $\tilde{y}_{ij}^{sd} > y_{ij}^{sd\star}(t) = x_i^{sd}(t)$ because $x_i^{sd}(t)$ is the active constraint, then $x_i^{sd}(t) \leq \tilde{y}_{ij}^{sd}(t) \leq C_{ij}$ so $w_{ij}^{sd}(t) \leq C_{ij}$ Also, $\tilde{y}_{ij}^{sd}(t) - y_{ij}^{sd\star}(t) \leq \tilde{y}_{ij}^{sd} \leq C_{ij}$. Observe that the $\sum_{s\in\mathcal{S}} \sum_{d\in\mathcal{N}} \sum_{(i,j)\in\mathcal{A}} \beta_2 y_{ij}^{sd\star}(t)(\phi_i^d - \phi_j^d)$ term in objective (36a) is bounded by $\sum_{s\in\mathcal{S}} \sum_{d\in\mathcal{N}} \sum_{(i,j)\in\mathcal{A}} \beta_2 C_{ij}(\phi_i^{sd} - \phi_j^{sd})$ and can therefore become part of the constant κ. Therefore $\left(\tilde{y}_{ij}^{sd}(t) - y_{ij}^{sd\star}(t)\right) w_{ij}^{sd}(t) \leq C_{ij} \times C_{ij}$ and

$$\sum_{s\in\mathcal{S}} \sum_{d\in\mathcal{N}} \sum_{(i,j)\in\mathcal{A}} \left(\tilde{y}_{ij}^{sd}(t) - y_{ij}^{sd\star}(t)\right) w_{ij}^{sd}(t) \leq \sum_{i\in\mathcal{N}} \sum_{j\in\Gamma_i^+} (C_{ij})^2 + \sum_{s\in\mathcal{S}} \sum_{d\in\mathcal{N}} \sum_{(i,j)\in\mathcal{A}} C_{ij}\beta_2 \left(\phi_i^{sd} - \phi_j^{sd}\right) \tag{A.7}$$

Combining equations (A.6) and (A.7),

$$\sum_{s\in\mathcal{S}} \sum_{d\in\mathcal{N}} \sum_{(i,j)\in\mathcal{A}} \left(\bar{y}_{ij}^{sd} - y_{ij}^{sd\star}(t)\right)$$

$$\leq \sum_{(i,j)\in\mathcal{A}} \sum_{d\in\mathcal{N}} C_{ij}^2 + \sum_{s\in\mathcal{S}} \sum_{d\in\mathcal{N}} \sum_{(i,j)\in\mathcal{A}} C_{ij}\beta_2(\phi_i^{sd} - \phi_j^{sd}) - \sum_{i\in\mathcal{N}} \frac{\epsilon}{|\Gamma_i^+||\mathcal{S}||\mathcal{N}|} \sum_{s\in\mathcal{S}} \sum_{d\in\mathcal{N}} \sum_{j\in\mathcal{N}} \left| w_{ij}^{sd}(t) \right| \tag{A.8}$$

which gives the κ and ϵ needed for equation (A.1). □

LEMMA 5. *When $\bar{\mathbf{r}} \in \hat{\mathcal{R}}^0$, there exists constants $\kappa < \infty$ and $\epsilon > 0$ such that*

$$\mathbb{E}\left[\sum_{p\in\mathcal{P}} \sum_{d\in\mathcal{N}} \sum_{i\in\mathcal{F}} \left(\bar{\gamma}_i^{pd} - \gamma_i^{pd\star}\right) \omega_i^{pd}(t) \middle| \mathbf{z}(t) \right] \leq \kappa - \epsilon|\boldsymbol{\omega}(t)| \tag{A.9}$$

Proof. Let $\tilde{\boldsymbol{\gamma}}(t)$ be defined as the optimal solution to the problem

$$\max \quad \sum_{p\in\mathcal{P}} \sum_{d\in\mathcal{N}} \sum_{i\in\mathcal{F}} \gamma_i^{pd}(t)\omega_i^{pd}(t) \tag{A.10a}$$

$$\text{s.t.} \quad \sum_{d\in\mathcal{N}} \gamma_i^{pd}(t) \leq \bar{\sigma}_i^p \qquad \forall p \in \mathcal{P}, \forall i \in \mathcal{F} \tag{A.10b}$$

$$\sum_{p\in\mathcal{P}} \sum_{d\in\mathcal{N}} \gamma_i^{pd}(t) \leq C_i \qquad \forall i \in \mathcal{F} \tag{A.10c}$$

$$\gamma_i^{pd}(t) \in \mathbb{Z}_+ \qquad \forall p \in \mathcal{P}, \forall d \in \mathcal{N}, \forall i \in \mathcal{F} \tag{A.10d}$$

$$\sum_{i\in\mathcal{F}} \gamma_i^{pd}(t) \leq \chi^{pd}(t) \qquad \forall p \in \mathcal{P}, \forall d \in \mathcal{N} \tag{A.10e}$$

which substitutes constraint (20) (average inventory deliveries) for constraint (6). We will show that

$$(\tilde{\boldsymbol{\gamma}}(t)-\boldsymbol{\gamma}^{\star}(t))\cdot\boldsymbol{\omega}(t)+(\bar{\boldsymbol{\gamma}}-\tilde{\boldsymbol{\gamma}}(t))\cdot\boldsymbol{\omega}(t)=(\bar{\boldsymbol{\gamma}}(t)-\boldsymbol{\gamma}^{\star}(t))\cdot\boldsymbol{\omega}(t)\leq\kappa-\epsilon|\boldsymbol{\omega}(t)| \tag{A.11}$$

Let $\hat{\boldsymbol{\gamma}}(t)=\arg\max_{\boldsymbol{\gamma}(t)}\{\boldsymbol{\gamma}(t)\cdot\boldsymbol{\omega}(t)\}$ subject to constraints (A.10b)–(A.10d). For all $p\in\mathcal{P}$ and $i\in\mathcal{F}$, there exists an $\epsilon>0$ such that

$$\sum_{d\in\mathcal{N}}\left(\hat{\gamma}_i^{pd}(t)-\bar{\gamma}_i^{pd}\right)\omega_i^{pd}(t)\leq-\epsilon\sum_{d\in\mathcal{N}}\omega_i^{pd}(t) \tag{A.12}$$

or

$$\sum_{p\in\mathcal{P}}\sum_{d\in\mathcal{N}}\left(\hat{\gamma}_i^{pd}(t)-\bar{\gamma}_i^{pd}\right)\omega_i^{pd}(t)\leq-\epsilon\sum_{p\in\mathcal{P}}\sum_{d\in\mathcal{N}}\omega_i^{pd}(t) \tag{A.13}$$

because constraints (20) and (21) are strict inequalities for $\bar{\gamma}$. Therefore

$$\sum_{p\in\mathcal{P}}\sum_{d\in\mathcal{N}}\sum_{i\in\mathcal{N}}\left(\hat{\gamma}_i^{pd}(t)-\bar{\gamma}_i^{pd}\right)\omega_i^{pd}(t)\leq-\epsilon\left(\max_{p\in\mathcal{P},d\in\mathcal{N},i\in\mathcal{F}}\omega_i^{pd}(t)\right) \tag{A.14}$$

$$\leq-\frac{\eta}{|\mathcal{P}||\mathcal{N}||\mathcal{F}|}\sum_{p\in\mathcal{P}}\sum_{d\in\mathcal{N}}\sum_{i\in\mathcal{F}}\left|\omega_i^{pd}(t)\right| \tag{A.15}$$

If

$$\sum_{p\in\mathcal{P}}\sum_{d\in\mathcal{N}}\sum_{i\in\mathcal{F}}\hat{\gamma}_i^{pd}(t)>\chi^{pd}(t)\geq\sum_{i\in\mathcal{F}}\tilde{\gamma}_i^{pd}(t) \tag{A.16}$$

because constraint (A.10e) is active, then

$$\sum_{p\in\mathcal{P}}\sum_{d\in\mathcal{N}}\sum_{i\in\mathcal{F}}\hat{\gamma}_i^{pd}(t)\leq\sum_{i\in\mathcal{F}}C_i \tag{A.17}$$

so

$$\sum_{p\in\mathcal{P}}\sum_{d\in\mathcal{N}}\sum_{i\in\mathcal{F}}\omega_i^{pd}(t)\leq\sum_{p\in\mathcal{P}}\sum_{d\in\mathcal{N}}\chi^{pd}(t)\leq\sum_{i\in\mathcal{F}}C_i \tag{A.18}$$

Therefore,

$$\sum_{p\in\mathcal{P}}\sum_{d\in\mathcal{N}}\sum_{i\in\mathcal{F}}\left(\hat{\gamma}_i^{pd}(t)-\tilde{\gamma}_i^{pd}(t)\right)\omega_i^{pd}(t)\leq C_i\times C_i \tag{A.19}$$

For calculating $\gamma_i^{pd\star}(t)$, constraint (A.10e) is replaced with constraint (6). Because of constraints (4) and (6), and because we assume that $V_i^p>\sigma_i^p(t)$, then

$$v_i^p(t+1)=\min\left\{v_i^p(t)-\sum_{d\in\mathcal{N}}\gamma_i^{pd}(t)+\sigma_i^p(t+1),V_i^p\right\}\geq\min\left\{\sigma_i^p(t+1),V_i\right\}=\sigma_i^p(t+1) \tag{A.20}$$

giving us $v_i^p(t)\geq\sigma_i^p(t)$ and $\mathbb{E}\left[v_i^p(t)\right]\geq\bar{\sigma}_i^p$. Also observe that the $-\sum_{p\in\mathcal{P}}\sum_{d\in\mathcal{N}}\sum_{i\in\mathcal{F}}\beta_1\gamma_i^{pd}(t)\phi_i^{\jmath(p)d}$ term in objective (37a) is bounded above by 0 and below by $-\sum_{p\in\mathcal{P}}\sum_{d\in\mathcal{N}}\sum_{i\in\mathcal{F}}\beta_1C_i\phi_i^{\jmath(p)d}$ and can therefore become part of the constant κ. From Madansky (1960),

$$\mathbb{E}\left[\sum_{p\in\mathcal{P}}\sum_{d\in\mathcal{N}}\sum_{i\in\mathcal{F}}\tilde{\gamma}_i^{pd}(t)\omega_i^{pd}(t)\right]\leq\mathbb{E}\left[\gamma_i^{pd\star}(t)\times\omega_i^{pd}(t)\right] \tag{A.21}$$

since the expected value of the constraint, $\bar{\sigma}_i^p$, is used for $\tilde{\boldsymbol{\gamma}}(t)$ and the actual realized value, $v_i^p(t)\geq\sigma_i^p(t)$, is used for $\boldsymbol{\gamma}^{\star}(t)$. Inequality (A.11) follows from inequalities (A.15), (A.19), and (A.21). □

We are now ready to prove Lemma 2.

Proof of Lemma 2. Let $\mathcal{L}(t)$ be the Lyapunov function defined as

$$\mathcal{L}(t) = \left(\sum_{p\in\mathcal{P}} \sum_{d\in\mathcal{N}} \chi^{pd}(t) + \sum_{i\in\mathcal{N}} \sum_{s\in\mathcal{S}} \sum_{d\in\mathcal{N}} x_i^{pd}(t) \right)^2 \tag{A.22}$$

Because $\boldsymbol{\chi}(t) \geq \mathbf{0}$ and $\mathbf{x}(t) \geq \mathbf{0}$, $\mathcal{L}(t) \geq 0$. We need to show that $\mathcal{L}(t)$ satisfies equation (38). Let $\delta_i^{sd}(t) = x_i^{sd}(t+1) - x_i^{sd}(t)$ and let $\psi^{pd}(t) = \chi^{pd}(t+1) - \chi^{pd}(t)$. Then

$$\mathcal{L}(t+1) - \mathcal{L}(t) = |\mathbf{x}(t) + \boldsymbol{\delta}(t)|^2 - |\mathbf{x}(t)|^2 + |\boldsymbol{\chi}(t) + \boldsymbol{\psi}(t)|^2 - |\boldsymbol{\chi}(t)|^2 \tag{A.23}$$

$$= |\boldsymbol{\delta}(t)|^2 + 2\mathbf{x}(t)\cdot\boldsymbol{\delta}(t) + |\boldsymbol{\psi}(t)|^2 + 2\boldsymbol{\chi}(t)\cdot\boldsymbol{\psi}(t) \tag{A.24}$$

From equation (2), $\psi^{pd}(t) = r^{pd}(t) - \sum_{i\in\mathcal{F}} \gamma_i^{pd}(t)$, so

$$\mathbb{E}\left[(\psi^{pd}(t))^2 \middle| \mathbf{z}(t) \right] \leq \left(\bar{r}^{pd}\right)^2 + (C_i)^2 \tag{A.25}$$

is bounded. From equation (1), when $i \in \mathcal{N}/\mathcal{F}$ then $\delta_j^{sd}(t) = \sum_{k\in\Gamma_j^+} y_{jk}^{sd}(t) + \sum_{i\in\Gamma_j^-} y_{ij}^{sd}(t)$. Then

$$\mathbb{E}\left[\left(\delta_j^{sd}(t)\right)^2 \middle| \mathbf{z}(t) \right] \leq (C_k)^2 + \sum_{i\in\Gamma_j^-} (C_i)^2 \tag{A.26}$$

which is bounded. For $i \in \mathcal{F}$, $\delta_j^{sd}(t) = \sum_{p\in\mathcal{P}^s} \gamma_j^{pd}(t) - \sum_{k\in\Gamma_j^+} y_{jk}^{sd}(t)$ from equation (3). Then

$$\mathbb{E}\left[\left(\delta_j^{sd}(t)\right)^2 \middle| \mathbf{z}(t) \right] \leq (C_i)^2 + \sum_{i\in\Gamma_j^-} (C_j)^2 \tag{A.27}$$

which is bounded. Therefore, equation (A.24) reduces to

$$\mathbb{E}\left[|\boldsymbol{\delta}(t)|^2 + 2\mathbf{x}(t)\cdot\boldsymbol{\delta}(t) + |\boldsymbol{\psi}(t)|^2 + 2\boldsymbol{\chi}(t)\cdot\boldsymbol{\psi}(t) \middle| \mathbf{z}(t) \right] \leq \kappa + 2\mathbb{E}\left[\mathbf{x}(t)\cdot\boldsymbol{\delta}(t) + \boldsymbol{\chi}(t)\cdot\boldsymbol{\psi}(t) \middle| \mathbf{z}(t) \right] \tag{A.28}$$

and we need to show that $\mathbb{E}\left[\mathbf{x}(t)\cdot\boldsymbol{\delta}(t) + \boldsymbol{\chi}(t)\cdot\boldsymbol{\psi}(t) \middle| \mathbf{z}(t) \right] \leq \kappa - \epsilon|\boldsymbol{\chi}(t) + \mathbf{x}(t)|$.

$$\begin{aligned}
\mathbf{x}(t)\cdot\boldsymbol{\delta}(t) &= \sum_{s\in\mathcal{S}} \sum_{d\in\mathcal{N}} \sum_{j\in\mathcal{N}/\mathcal{F}} x_j^{sd}(t) \left(\sum_{i\in\Gamma_j^-} y_{ij}^{sd}(t) - \sum_{k\in\Gamma_j^+} y_{jk}^{sd}(t) \right) \\
&\quad + \sum_{s\in\mathcal{S}} \sum_{d\in\mathcal{N}} \sum_{j\in\mathcal{F}} x_j^{sd}(t) \left(\sum_{p\in\mathcal{P}^s} y_j^{pd}(t) - \sum_{k\in\mathcal{N}} y_{jk}^{sd}(t) \right) && \text{(A.29)} \\
&= \sum_{s\in\mathcal{S}} \sum_{d\in\mathcal{N}} \sum_{k\in\mathcal{N}/\mathcal{F}} \sum_{j\in\Gamma_j^-} x_k^{sd}(t) y_{jk}^{sd}(t) - \sum_{s\in\mathcal{S}} \sum_{d\in\mathcal{N}} \sum_{j\in\mathcal{N}/\mathcal{F}} \sum_{k\in\Gamma_j^+} x_j^{sd}(t) y_{jk}^{sd}(t) \\
&\quad + \sum_{s\in\mathcal{S}} \sum_{d\in\mathcal{N}} \sum_{j\in\mathcal{F}} \sum_{p\in\mathcal{P}^s} x_j^{sd}(t) y_j^{pd}(t) - \sum_{s\in\mathcal{S}} \sum_{d\in\mathcal{N}} \sum_{j\in\mathcal{F}} \sum_{k\in\mathcal{N}} x_j^{sd}(t) y_{jk}^{sd}(t) && \text{(A.30)} \\
&= \sum_{s\in\mathcal{S}} \sum_{d\in\mathcal{N}} \sum_{j\in\mathcal{N}} \sum_{k\in\Gamma_j^+} y_{jk}^{sd}(t) \left(x_k^{sd}(t) - x_j^{sd}(t) \right) + \sum_{s\in\mathcal{S}} \sum_{d\in\mathcal{N}} \sum_{j\in\mathcal{F}} \sum_{p\in\mathcal{P}^s} x_j^{sd}(t) y_j^{pd}(t) && \text{(A.31)}
\end{aligned}$$

where indices (i, j) are replaced by (j, k) in equation (A.30). Next,

$$\begin{aligned}
\boldsymbol{\chi}(t)\cdot\boldsymbol{\psi}(t) &= \sum_{p\in\mathcal{P}} \sum_{d\in\mathcal{N}} \chi^{pd}(t) \left(r^{pd}(t) - \sum_{j\in\mathcal{F}} y_j^{pd}(t) \right) && \text{(A.32)} \\
&= \sum_{p\in\mathcal{P}} \sum_{d\in\mathcal{N}} \chi^{pd}(t) r^{pd}(t) - \sum_{p\in\mathcal{P}} \sum_{d\in\mathcal{N}} \sum_{j\in\mathcal{F}} \chi^{pd}(t) y_j^{pd}(t) && \text{(A.33)}
\end{aligned}$$

Therefore,

$$\sum_{s\in\mathcal{S}}\sum_{d\in\mathcal{N}}\sum_{j\in\mathcal{F}}\sum_{p\in\mathcal{P}^s} x_j^{sd}(t)y_j^{pd}(t)+\boldsymbol{\chi}(t)\cdot\boldsymbol{\psi}(t)=\sum_{s\in\mathcal{S}}\sum_{d\in\mathcal{N}}\sum_{j\in\mathcal{F}}\sum_{p\in\mathcal{P}^s} y_j^{pd}(t)\left(x_j^{sd}(t)-\chi^{pd}(t)\right)+\sum_{p\in\mathcal{P}}\sum_{d\in\mathcal{N}}\chi^{pd}(t)r^{pd}(t) \tag{A.34}$$

Combining equations (A.31) and (A.34),

$$\begin{aligned}\mathbf{x}(t)\cdot\boldsymbol{\delta}(t)+\boldsymbol{\chi}(t)\cdot\boldsymbol{\psi}(t)=&-\sum_{s\in\mathcal{S}}\sum_{d\in\mathcal{N}}\sum_{j\in\mathcal{N}}\sum_{k\in\Gamma_j^+} y_{jk}^{sd}(t)\left(x_j^{sd}(t)-x_k^{sd}(t)\right)\\&-\sum_{s\in\mathcal{S}}\sum_{d\in\mathcal{N}}\sum_{j\in\mathcal{F}}\sum_{p\in\mathcal{P}^s} y_j^{pd}(t)\left(\chi^{pd}(t)-x_j^{sd}(t)\right)+\sum_{p\in\mathcal{P}}\sum_{d\in\mathcal{N}}\chi^{pd}(t)r^{pd}(t) &\text{(A.35)}\\=&-\sum_{s\in\mathcal{S}}\sum_{d\in\mathcal{N}}\sum_{j\in\mathcal{N}}\sum_{k\in\Gamma_j^+} y_{jk}^{sd}(t)w_j^{sd}(t)-\sum_{s\in\mathcal{S}}\sum_{d\in\mathcal{N}}\sum_{j\in\mathcal{F}}\sum_{p\in\mathcal{P}^s} y_j^{pd}(t)\omega_j^{pd}(t)+\sum_{p\in\mathcal{P}}\sum_{d\in\mathcal{N}}\chi^{pd}(t)r^{pd}(t) &\text{(A.36)}\end{aligned}$$

Observe that $\mathbb{E}\left[\chi^{pd}(t)r^{pd}(t)|\mathbf{z}(t)\right]=\chi^{pd}(t)\bar{r}^{pd}$. Next, using equations (17)–(19), we rewrite $\sum_{p\in\mathcal{P}}\sum_{d\in\mathcal{N}}\chi^{pd}(t)\bar{r}^{pd}$ as

$$\begin{aligned}\sum_{p\in\mathcal{P}}\sum_{d\in\mathcal{N}}\chi^{pd}(t)\bar{r}^{pd}=&\sum_{p\in\mathcal{P}}\sum_{d\in\mathcal{N}}\sum_{i\in\mathcal{F}}\chi^{pd}(t)\bar{\gamma_i}^{pd}-\sum_{s\in\mathcal{S}}\sum_{d\in\mathcal{N}}\sum_{i\in\mathcal{F}}\sum_{p\in\mathcal{P}^s} x_i^{sd}(t)\bar{\gamma}_i^{pd}+\sum_{s\in\mathcal{S}}\sum_{d\in\mathcal{N}}\sum_{i\in\mathcal{F}}\sum_{j\in\Gamma_i^+} x_i^{sd}(t)\bar{y}_{ij}^{sd} &\text{(A.37)}\\=&\sum_{s\in\mathcal{S}}\sum_{d\in\mathcal{N}}\sum_{i\in\mathcal{F}}\sum_{p\in\mathcal{P}^s}\bar{\gamma}_i^{pd}\left(\chi^{pd}(t)-x_i^{sd}(t)\right)+\sum_{s\in\mathcal{S}}\sum_{d\in\mathcal{N}}\sum_{i\in\mathcal{N}}\sum_{j\in\Gamma_i^+} x_i^{sd}(t)\bar{y}_{ij}^{sd}-\sum_{s\in\mathcal{S}}\sum_{d\in\mathcal{N}}\sum_{j\in\mathcal{N}/\mathcal{F}}\sum_{k\in\Gamma_j^+} x_j^{sd}(t)\bar{y}_{jk}^{sd} &\text{(A.38)}\\=&\sum_{s\in\mathcal{S}}\sum_{d\in\mathcal{N}}\sum_{i\in\mathcal{F}}\sum_{p\in\mathcal{P}^s}\bar{\gamma}_i^{pd}\left(\chi^{pd}(t)-x_i^{sd}(t)\right)+\sum_{s\in\mathcal{S}}\sum_{d\in\mathcal{N}}\sum_{i\in\mathcal{N}}\sum_{j\in\Gamma_i^+} x_i^{sd}(t)\bar{y}_{ij}^{sd}-\sum_{s\in\mathcal{S}}\sum_{d\in\mathcal{N}}\sum_{j\in\mathcal{N}/\mathcal{F}} x_j^{sd}(t)\left(\sum_{i\in\Gamma_j^-}\bar{y}_{ij}^{sd}\right) &\text{(A.39)}\\=&\sum_{s\in\mathcal{S}}\sum_{d\in\mathcal{N}}\sum_{i\in\mathcal{F}}\sum_{p\in\mathcal{P}^s}\left(\chi^{pd}(t)-x_i^{sd}(t)\right)\bar{\gamma}_i^{pd}+\sum_{s\in\mathcal{S}}\sum_{d\in\mathcal{N}}\sum_{i\in\mathcal{N}}\sum_{j\in\Gamma_i^+}\bar{y}_{ij}^{sd}\left(x_i^{sd}(t)-x_j^{sd}(t)\right) &\text{(A.40)}\\=&\sum_{s\in\mathcal{S}}\sum_{d\in\mathcal{N}}\sum_{i\in\mathcal{F}}\sum_{p\in\mathcal{P}^s}\bar{\gamma}_i^{pd}\omega_i^{pd}(t)+\sum_{s\in\mathcal{S}}\sum_{d\in\mathcal{N}}\sum_{i\in\mathcal{N}}\sum_{j\in\Gamma_i^+}\bar{y}_{ij}^{sd}w_i^{sd}(t) &\text{(A.41)}\end{aligned}$$

where equation (A.38) is formed by adding and substracting $\sum_{s\in\mathcal{S}}\sum_{d\in\mathcal{N}}\sum_{i\in\mathcal{N}/\mathcal{F}}\sum_{j\in\Gamma_i^+} x_i^{sd}(t)\bar{y}_{ij}^{sd}$. Combining equations (A.36) and (A.41) yields

$$\begin{aligned}&\mathbb{E}\left[\mathbf{x}(t)\cdot\boldsymbol{\delta}(t)+\boldsymbol{\chi}(t)\cdot\boldsymbol{\psi}(t)|\mathbf{z}(t)\right]\\&\qquad=\sum_{p\in\mathcal{P}}\sum_{d\in\mathcal{N}}\sum_{i\in\mathcal{F}}\left(\bar{\gamma}_i^{pd}-\gamma_i^{pd}(t)\right)\omega_i^{pd}(t)+\sum_{s\in\mathcal{S}}\sum_{d\in\mathcal{N}}\sum_{i\in\mathcal{N}}\sum_{j\in\Gamma_i^+}\left(\bar{y}_{ij}^{sd}-y_{ij}^{sd}(t)\right)w_i^{sd}(t) &\text{(A.42)}\end{aligned}$$

By Lemmas 4 and 5, when $\pi^\star$ is used

$$\mathbb{E}\left[\mathbf{x}(t)\cdot\boldsymbol{\delta}(t)+\boldsymbol{\chi}(t)\cdot\boldsymbol{\psi}(t)|\mathbf{z}(t)\right]\le\kappa-\epsilon|\mathbf{w}(t)|-\epsilon|\boldsymbol{\omega}(t)| \tag{A.43}$$

Because $\mathbf{w}(t)$ is a linear function of $\mathbf{x}(t)$ and $\boldsymbol{\omega}(t)$ is a linear function of $\mathbf{x}(t)$ and $\boldsymbol{\chi}(t)$, there exists $\eta>0$ such that

$$|\mathbf{x}(t)|+|\boldsymbol{\chi}(t)|\ge\eta\left(|\mathbf{w}(t)|+|\boldsymbol{\omega}(t)|\right) \tag{A.44}$$

Combining equations (A.43) and (A.44) yields

$$\mathbb{E}\left[\mathbf{x}(t)\cdot\boldsymbol{\delta}(t)+\boldsymbol{\chi}(t)\cdot\boldsymbol{\psi}(t)|\mathbf{z}(t)\right]\le\kappa-\epsilon\eta|\mathbf{x}(t)|-\epsilon\eta|\boldsymbol{\chi}(t)| \tag{A.45}$$

which achieves equation (38). □

Appendix B: Proof of Lemma 3

As in the proof of Lemma 2, we first prove bounds on the approximation scheme.

LEMMA 6. *Consider an α approximation scheme to problem* (37)*, and let $\bar{\mathbf{r}} \in \hat{\mathcal{R}}^0$. Let α be sufficiently small so that*

$$\epsilon > \alpha \left(\max_{i \in \mathcal{N}} C_i \right) \tag{B.1}$$

where ϵ is the smallest gap between constraints (22)–(24)*. Then there exists constants $\kappa < \infty$ and $\epsilon' > 0$ such that*

$$\sum_{s \in \mathcal{S}} \sum_{d \in \mathcal{N}} \sum_{(i,j) \in \mathcal{A}} \left(\bar{y}_{ij}^{sd} - y_{ij}^{sd\star}(t) \right) w_{ij}^{sd}(t) \leq \kappa - \epsilon' |\mathbf{w}(t)| \tag{B.2}$$

Proof. Let $\hat{\mathbf{y}}(t)$ be the α approximation scheme to problem (37), which satisfies equation (43) by definition. Consider that

$$\sum_{s \in \mathcal{S}} \sum_{d \in \mathcal{N}} \sum_{i \in \mathcal{N}} \sum_{j \in \mathcal{A}_i \in \Gamma_i^+} y_{ij}^{sd\star}(t) w_{ij}^{sd}(t) \leq \sum_{s \in \mathcal{S}} \sum_{d \in \mathcal{N}} \sum_{i \in \mathcal{N}} \sum_{j \in \mathcal{A}_i \in \Gamma_i^+} C_i w_{ij}^{sd}(t) \leq \left(\max_{i \in \mathcal{N}} C_i \right) \mathbf{w}(t) \tag{B.3}$$

Let ϵ' be defined by

$$\epsilon' = \epsilon - \alpha \left(\max_{i \in \mathcal{N}} C_i \right) \tag{B.4}$$

Using equation (43),

$$\begin{aligned}
\bar{\mathbf{y}} \cdot \mathbf{w}(t) - \hat{\mathbf{y}}(t) \cdot \mathbf{w}(t) &\leq \bar{\mathbf{y}} \cdot \mathbf{w}(t) - (1-\alpha) \mathbf{y}^\star(t) \cdot \mathbf{w}(t) && \text{(B.5)} \\
&\leq \kappa - \epsilon |\mathbf{w}(t)| + \alpha \mathbf{y}^\star(t) \cdot \mathbf{w}(t) && \text{(B.6)} \\
&\leq \kappa - \epsilon |\mathbf{w}(t)| + \alpha \left(\max_{i \in \mathcal{N}} C_i \right) \mathbf{w}(t) && \text{(B.7)} \\
&= \kappa - \left[\epsilon - \alpha \left(\max_{i \in \mathcal{N}} C_i \right) \right] |\mathbf{w}(t)| && \text{(B.8)} \\
&= \kappa - \epsilon' |\mathbf{w}(t)| && \text{(B.9)}
\end{aligned}$$

where equation (B.6) follows from Lemma 4. Based on the assumption of inequality (B.1), $\epsilon' > 0$. □

LEMMA 7. *Consider an α approximation scheme to problem* (37)*, and let $\bar{\mathbf{r}} \in \hat{\mathcal{R}}^0$. Let α be sufficiently small so that*

$$\epsilon > \alpha \left(\max_{i \in \mathcal{N}} C_i \right) \tag{B.10}$$

where ϵ is the smallest gap between constraints (21)*. Then there exists constants $\kappa < \infty$ and $\epsilon' > 0$ such that*

$$\sum_{p \in \mathcal{P}} \sum_{d \in \mathcal{N}} \sum_{(i,j) \in \mathcal{A}} \left(\bar{\gamma}_i^{pd} - \gamma_i^{pd\star}(t) \right) \omega_i^{pd}(t) \leq \kappa - \epsilon' |\boldsymbol{\omega}(t)| \tag{B.11}$$

The proof of Lemma 7 follows similarly to the proof of Lemma 6. Now we can prove Lemma 3.

Proof of Lemma 3. By Lemmas 6 and 7, when $\pi^\star$ is used, there exists an $\alpha > 0$ such that $\epsilon' > 0$ and

$$\mathbb{E}\left[\mathbf{x}(t) \cdot \boldsymbol{\delta}(t) + \boldsymbol{\chi}(t) \cdot \boldsymbol{\psi}(t) | \mathbf{z}(t) \right] \leq \kappa - \epsilon' |\mathbf{w}(t)| - \epsilon' |\boldsymbol{\omega}(t)| \tag{B.12}$$

which replaces inequality (A.43). The remainder of the proof is the same as the proof of Lemma 2. □

Appendix C: Greedy fulfillment policies π^{gdy} and π^{gdy2} used for numerical results

for $t \in T$ **do**
  for $p \in \mathcal{P}$, $d \in \mathcal{N}$ **do**
    For all s, $\phi_i^{sd}(t)$ is the minimum cost-to-go to d with link length of 1 (for π^{gdy}) or ℓ_{ij} (for π^{gdy2})
    while $\chi^{pd}(t) > 0$ **do**
      Find the i with $v_i^p(t) > 0$ with minimum $\phi_i^{s(p)d}$
      if i does not exist **then**
        break
      end if
      $\gamma_i^{pd}(t) \leftarrow \gamma_i^{pd}(t) + \min\{\chi^{pd}(t), v_i^p(t)\}$ but respect constraint (7)
    end while
  end for
  Using $\gamma_i^{pd}(t)$, fulfill orders according to equation (2)
  for $i \in \mathcal{N}$ **do**
    for $d \in \mathcal{N}$, $s \in \mathcal{N} : x_i^{sd}(t) > 0$ **do**
      Find the j with minimum ϕ_j^{sd}
      $y_{ij}^{sd}(t) \leftarrow x_i^{sd}(t)$ but respect constraints (8)–(13)
    end for
    Using $y_{ij}^{sd}(t)$, move packages according to equation (3)
  end for
  for $i \in \mathcal{F}$ **do**
    Update inventory stock using equation (4)
  end for
end for

References

Abouelrous A, Gabor AF, Zhang Y, 2022 *Joint inventory and fulfilment optimization for an omnichannel retailer: A stochastic optimization approach. Computers & Industrial Engineering* 108723.

Acimovic J, Farias VF, 2019 *The fulfillment-optimization problem. Operations Research & Management Science in the age of analytics*, 218–237 (INFORMS).

Acimovic J, Graves SC, 2015 *Making better fulfillment decisions on the fly in an online retail environment. Manufacturing & Service Operations Management* 17(1):34–51.

Agatz N, Campbell A, Fleischmann M, Savelsbergh M, 2011 *Time slot management in attended home delivery. Transportation Science* 45(3):435–449.

Agatz NA, Fleischmann M, Van Nunen JA, 2008 *E-fulfillment and multi-channel distribution–a review. European Journal of Operational Research* 187(2):339–356.

Alibeiki H, Hassanlou M, Jorjani S, 2020 *Fulfillment optimization for online retailers under stochastic demands. Journal of Supply Chain and Operations Management* 18(2):194.

Andrews JM, Farias VF, Khojandi AI, Yan CM, 2019 *Primal–dual algorithms for order fulfillment at urban outfitters, inc. INFORMS Journal on Applied Analytics* 49(5):355–370.

Barman S, Levin MW, 2022 *Performance evaluation of modified cyclic max-pressure controlled intersections in realistic corridors. Transportation Research Record* 03611981211072807.

Bayram A, Cesaret B, 2021 *Order fulfillment policies for ship-from-store implementation in omni-channel retailing. European Journal of Operational Research* 294(3):987–1002.

Chekuri C, Khanna S, 2005 *A polynomial time approximation scheme for the multiple knapsack problem. SIAM Journal on Computing* 35(3):713–728.

Ganti A, Modiano E, Tsitsiklis JN, 2007 *Optimal transmission scheduling in symmetric communication models with intermittent connectivity. IEEE Transactions on Information Theory* 53(3):998–1008.

Govindarajan A, Sinha A, Uichanco J, 2021 *Joint inventory and fulfillment decisions for omnichannel retail networks. Naval Research Logistics (NRL)* 68(6):779–794.

Jasin S, Sinha A, 2015 *An lp-based correlated rounding scheme for multi-item ecommerce order fulfillment. Operations Research* 63(6):1336–1351.

Jiu S, 2022 *Robust omnichannel retail operations with the implementation of ship-from-store. Transportation Research Part E: Logistics and Transportation Review* 157:102550.

Kang D, Levin MW, 2021 *Maximum-stability dispatch policy for shared autonomous vehicles. Transportation Research Part B: Methodological* 148:132–151.

Lei Y, Jasin S, Sinha A, 2018 *Joint dynamic pricing and order fulfillment for e-commerce retailers. Manufacturing & Service Operations Management* 20(2):269–284.

Li L, Pantelidis T, Chow JY, Jabari SE, 2021 *A real-time dispatching strategy for shared automated electric vehicles with performance guarantees. Transportation Research Part E: Logistics and Transportation Review* 152:102392.

Lim YF, Jiu S, Ang M, 2021 *Integrating anticipative replenishment allocation with reactive fulfillment for online retailing using robust optimization. Manufacturing & Service Operations Management* 23(6):1616–1633.

Lin YH, Wang Y, Lee LH, Chew EP, 2022 *Omnichannel facility location and fulfillment optimization. Transportation Research Part B: Methodological* 163:187–209.

Madansky A, 1960 *Inequalities for stochastic linear programming problems. Management science* 6(2):197–204.

Neely M, 2022 *Stochastic network optimization with application to communication and queueing systems* (Springer Nature).

Sarimveis H, Patrinos P, Tarantilis CD, Kiranoudis CT, 2008 *Dynamic modeling and control of supply chain systems: A review. Computers & Operations Research* 35(11):3530–3561.

Sluijk N, Florio AM, Kinable J, Dellaert N, Van Woensel T, 2022 *Two-echelon vehicle routing problems: A literature review. European Journal of Operational Research* .

Stolyar AL, 2004 *Maxweight scheduling in a generalized switch: State space collapse and workload minimization in heavy traffic. The Annals of Applied Probability* 14(1):1–53.

Sun X, Yin Y, 2018 *A simulation study on max pressure control of signalized intersections. Transportation Research Record* 2672(18):117–127.

Tassiulas L, Ephremides A, 1990 *Stability properties of constrained queueing systems and scheduling policies for maximum throughput in multihop radio networks. 29th IEEE Conference on Decision and Control*, 2130–2132 (IEEE).

Taylor D, Brockhaus S, Knemeyer AM, Murphy P, 2019 *Omnichannel fulfillment strategies: defining the concept and building an agenda for future inquiry. The International Journal of Logistics Management* .

Varaiya P, 2013 *Max pressure control of a network of signalized intersections. Transportation Research Part C: Emerging Technologies* 36:177–195.

Walton NS, 2014 *Concave switching in single and multihop networks. ACM SIGMETRICS Performance Evaluation Review* 42(1):139–151.

Wongpiromsarn T, Uthaicharoenpong T, Wang Y, Frazzoli E, Wang D, 2012 *Distributed traffic signal control for maximum network throughput. 2012 15th international IEEE conference on intelligent transportation systems*, 588–595 (IEEE).

Xu PJ, Allgor R, Graves SC, 2009 *Benefits of reevaluating real-time order fulfillment decisions. Manufacturing & Service Operations Management* 11(2):340–355.

Xu Z, Zhang H, Zhang J, Zhang RQ, 2020 *Online demand fulfillment under limited flexibility. Management Science* 66(10):4667–4685.