Submitted to *Management Science*
manuscript MS-17-01745

Capacity Allocation in Flexible Production Networks: Theory and Applications

Guodong Lyu
National University of Defense Technology, Science and Technology on Information Systems Engineering Laboratory, Changsha, Hunan, CN 410073
National University of Singapore, NUS Business School, Department of Analytics & Operations, Singapore, SG 119245, guodong.lyu@u.nus.edu

Wang-Chi Cheung
National University of Singapore, NUS Engineering, Department of Industrial Systems Engineering and Management, Singapore, SG 117576, wangchimit@gmail.com

Mabel C. Chou
National University of Singapore, NUS Business School, Department of Analytics & Operations, Singapore, SG 119245, bizchoum@nus.edu.sg

Chung-Piaw Teo
National University of Singapore, Institute of Operations Research and Analytics, Singapore, SG 117602
National University of Singapore, NUS Business School, Department of Analytics & Operations, Singapore, SG 117592, bizteocp@nus.edu.sg

Zhichao Zheng
Singapore Management University, Lee Kong Chian School of Business, Singapore, SG 178899, danielzheng@smu.edu.sg

Yuanguang Zhong
South China University of Technology, School of Business Administration, Guangzhou, Guangdong, CN 510640, bmygzhong@scut.edu.cn

In many production environments, a fixed network of capacity is shared flexibly between multiple products with random demands. What is the best way to configure the capacity of the production network and to allocate the available capacity, to meet pre-determined fill rate requirements? We develop a new approach for network capacity configuration and allocation, and characterize the relationship between the capacity of the network and the attainable fill rate levels for the products, taking into account the flexibility structure of the network. This builds on a new randomized allocation mechanism to deliver the desired services.

We use this theory to investigate the connection between the flexibility structure and capacity configuration. We provide a new perspective to the well-known phenomenon that "long chain is almost as good as the fully flexible network": For given target fill rates, the required capacity level in a long-chain network is close to that in a fully flexible network, and is much lower than a dedicated system. We apply these insights and techniques on problems arising in the design of last mile delivery operations, and in semi-conductor production planning, using real data from two companies.

Key words: Production Networks; Capacity Configuration; Process Flexibility; Fill Rate Targets
History: Received August 8, 2017; Revisions received April 2, 2018, June 25, 2018; Accepted July 5, 2018

1. Introduction

The fundamental problem in any supply chain system is to match available supply or capacity with (random) demand, using various buffers such as safety inventory, excess capacity, or flexible production and routing techniques, to facilitate service deliveries. In their classic work on factory physics, Hopp and Spearman (2008) postulated that flexibility can reduce the amount of buffers needed to mitigate the effects of demand variability. This follows a spate of activities in the industry to embrace flexible production techniques in its manufacturing system.[1]

Unfortunately, the ability to allocate capacity in a flexible manner leads to challenging control and costly coordination problems. In a survey among almost 250 manufacturers over 10 years (2005–2015),[2] 78% of the companies indicated that while they have enhanced their ability to adjust production to respond to market changes, they have also lost control on the production costs. Software giants like SAP and IBM have seized this opportunity to offer tools to help make better trade-offs in capacity/inventory allocation. For instance, SAP featured "service level optimization" as the next wave of innovations, beyond enterprise inventory optimization, to transform supply chain planning from cost containment into revenue optimization.[3] This is a complicated problem because multiple stakeholders in finance, marketing and operations etc. need to agree on the target service levels set for each product in the system, so that proper resource allocation can follow. To this end, we must address the following core question: **Do we have enough capacity in the flexible production system to achieve the target service levels for different products?**

Among various types of flexible manufacturing techniques, one particular stream—process flexibility—has been extensively studied in the operations management literature. Process flexibility entails the ability to reallocate fixed resources to produce different types of products (Jordan and Graves 1995, Goyal and Netessine 2011). In their seminal paper, Jordan and Graves (1995) showed that systems with well-designed limited flexibility can achieve almost the same performance as the full-flexibility system, in which all types of products can be produced at any plants. This has motivated a series of follow-up works on process flexibility to investigate the benefits of sparse flexibility design (e.g., Chou et al. 2010, 2011, 2014, Simchi-Levi and Wei 2015, Wang and Zhang 2015, Asadpour et al. 2017). The techniques developed thus far, however, ignore the connection between capacity configuration and service levels attainable, which is still an open question prior to this work to the best of our knowledge.

[1] For instance, Ford celebrates 100th anniversary of the moving assembly line with new goals for advanced, flexible manufacturing (October 7, 2013); retrieved from `https://media.ford.com/content/fordmedia-mobile/fna/us/en/news/2013/10/07/ford-celebrates-100th-anniversary-of-the-moving-assembly-line-wi` on March 31, 2017.

[2] How manufacturers can get faster, more flexible, and cheaper (February 27, 2017); retrieved from `https://hbr.org/2017/02/how-manufacturers-can-get-faster-more-flexible-and-cheaper` on March 31, 2017.

[3] SAP service level optimization (September 19, 2016); retrieved from `https://wiki.scn.sap.com/wiki/display/SCM/Service+Level+Optimization+-+Introduction+and+Fundamentals` on August 6, 2017.

We can solve the capacity configuration problem using a numerical approach, by first generating a large number of samples from the demand distribution, and solving a related optimization problem to find the minimum level of capacity needed to attain the desired service levels, based on the samples generated. This traditional approach—sampling average approximation (SAA)—suppresses the issue of allocation logic, and only constructs via brute force an allocation decision for each scenario represented in the samples generated; it is otherwise silent on the allocation policy for any new scenarios not among the samples generated.

We patch up the theory of SAA in this paper, and show that it remains a viable approach for capacity configuration under fill rate constraints, by explicitly constructing the allocation rule. Our approach solves a stochastic programming problem using a finite set of samples, but we explicitly construct the priority rule used in the allocation logic for each scenario. In this way, we obtain a randomized priority allocation rule for this class of problem that can be used on any new sample generated from the same demand distribution.

More specifically, we consider a supply chain network with flexible plants configured to meet the demands of multiple products, where the demand distribution of each product is known to the supply chain planner. We need to construct an allocation policy to ensure that the target fill rate level of each product is attainable. We focus on the class of randomized priority rules constructed in the following way: Let X_i and S_j denote the random demand and capacity for product i and plant j, respectively. We find a set of weights $w_i(\boldsymbol{X}, \boldsymbol{S})$'s after demand realization, and solve the following assignment problem to determine the allocation $\boldsymbol{D}$:

$$\begin{aligned} \max \ & \sum_{j,i} w_i(\boldsymbol{X}, \boldsymbol{S}) D_{j,i} \\ \text{s.t.} \ & \sum_{j} D_{j,i} \le X_i, \forall i \\ & \sum_{i} D_{j,i} \le S_j, \forall j \\ & D_{j,i} \ge 0, \forall i, j \end{aligned}$$

Here $D_{j,i}$ represents the resource allocated from plant j to serve the demand of product i. For ease of exposition, we suppress the dependence on S from $w_i(\cdot)$ in subsequent discussion.

By varying the weight $w_i(\boldsymbol{X})$ for different demand scenario $\boldsymbol{X}$, we can fine tune the service level delivered to product i to satisfy the target fill rate level. For instance, if $w_i(\boldsymbol{X}) = X_i$, the allocation rule gives priority to product with higher realized demand value. If $w_i(\boldsymbol{X}) =$ "the revenue of product i", then the priority is given to products with higher revenues. We show that for every attainable service level targets, β_i's, there is a corresponding set of random weights, $w_i(\boldsymbol{X})$'s, that will allow the above allocation method to deliver the desired service level to each

product. Interestingly, we show that $w_i(\boldsymbol{X})$ can be constructed in a dynamic manner and recast using a randomized policy that does not depend on $\boldsymbol{X}$. In this way, our random priority allocation rule does not depend on the realized product demands.

Our key contributions in this paper are as follows:

(1) We construct a randomized priority rule to obtain an allocation mechanism that can satisfy the service level requirements for all products, whenever the capacity in the network is sufficient. Furthermore, we develop a set of necessary and sufficient conditions on the optimal capacity configuration to ensure that the target fill rate levels are attainable.

(2) We present an application of this theory on multi-item newsvendor problem with upgrading. Interestingly, we show that the set of conditions obtained generalizes the critical fractile condition for the classical single item newsvendor model.

(3) We develop a two-stage online gradient descent algorithm to solve the capacity configuration problem. Using the online convex optimization techniques and "online to batch" conversions, we establish that the capacity values obtained converge (almost surely) to the optimal solution.

(4) For given target fill rate levels, we observe from our numerical results that the optimal capacity level needed for a long-chain network (sparse process flexibility) is already close to that for the fully flexible network. This provides a new twist to the well-known phenomenon that long chain is almost as good as the fully flexible network in terms of demand fulfillment.

(5) We evaluate the performance of our capacity configuration techniques on two problems from the industry, and demonstrate a range of improvements and insights for these problems, using real data provided by two companies. In particular, we test the effectiveness of long-chain structure in a last mile delivery operations, and scrutinize the strategic role of subcontractors in supporting production in a flexible manufacturing network.

The rest of this paper is organized as follows. We review the relevant literature in Section 2. In Section 3, we formally introduce our model and address the capacity allocation problem by explicitly constructing an optimal allocation policy. We also demonstrate an application of our key results in a multi-item newsvendor problem with product upgrades. In Section 4, we address the capacity configuration problem by developing a two-stage online gradient descent algorithm. In Section 5, we numerically demonstrate the effectiveness of long-chain network in mitigating demand uncertainty from the perspective of capacity configuration and service-level guarantee. We further apply these insights on the design of last mile delivery operations. In Section 6, we discuss how these techniques can be used to support production planning in a flexible network. We conclude this work in Section 7. All technical proofs are relegated to Appendix B.

2. Literature Review

Our research is closely related to the stream of literatures on process flexibility. The concepts of chaining structure and limited flexibility proposed by Jordan and Graves (1995) are arguably the most influential ideas. Jordan and Graves (1995) demonstrated that (1) limited flexibility, if configured in the right way, can accrue most of benefits obtained from a full-flexibility system (in terms of demand fulfilled or lost sales); and (2) limited flexibility has the greatest benefits in a network with long chain. Motivated by these findings, Chou et al. (2010) proved that the asymptotic expected fulfilled demand in a long-chain system is close to that in a full-flexibility system using a random walk approach. Chou et al. (2011) argued that good production networks are essentially highly connected graphs and they showed that the graph expander structure with $O(n/\epsilon)$ arcs is able to achieve $(1-\epsilon)$ optimality of full-flexibility networks. In a similar spirit, Chen et al. (2015) used probabilistic graph expanders with $O(n\ln(1/\epsilon))$ arcs to demonstrate the same $(1-\epsilon)$-optimality with high probability. Beyond the balanced and symmetrical assumptions, Chen et al. (2018) generalized this result to a general class of production networks. In comparison with other designs with limited-flexibility, Simchi-Levi and Wei (2012) showed that the long-chain structure is optimal among all 2-flexibility systems in terms of the expected fulfilled demand, for a balanced and symmetrical system with exchangeable demand. Désir et al. (2016) further demonstrated that the long-chain design is optimal among all those connected networks with at most $2n$ arcs over n supply and n demand nodes, but there exists a disconnected network with $2n$ edges performing better than the long-chain under a special demand case. The effectiveness of long-chain design has also been analyzed in other setting, including Chou et al. (2014), Simchi-Levi and Wei (2015). Interestingly, Wang and Zhang (2015) developed a semidefinite programming formulation to calculate the asymptotic performance ratio of long-chain and derived a closed-form lower bound when the capacity level is identical to the demand mean value in the framework of robustness. Besides the analytical investigation, the benefits of process flexibility have also been verified empirically in the literature, such as Suarez et al. (1996) and Hallgren and Olhager (2009).

Different from these theoretical works that were studied in a single-period setting, Shi et al. (2015) studied the design of process flexibility problem in a multi-period setting. We note that Shi et al. (2015) also provided a max-weight policy for resource allocation, but their objective is to minimize the total expected backlogging cost while we aim to achieve the fill rate target with the minimal capacity profile. In addition, Shi et al. (2015) allowed the unsatisfied demand to be backlogged to next period, but we study the lost-sale model. These two differences lead to significantly different analysis. Asadpour et al. (2017) investigated the benefits of long-chain flexibility design in an online environment in which stochastic demands arrive over time and the resource allocation decisions must be made immediately when new demand arrives. They derived

an upper bound on the expected total number of lost sales that is independent of the market size. It is interesting to observe that their myopic online fulfillment policy is similar to the allocation policy we develop in this paper. However, the problem we study is inherently different from the current literature on process flexibility: We consider the capacity configuration problem given any pre-determined flexibility structure and service level requirements, rather than trying to bound the gap between lost sales of specific flexibility designs given capacity levels.

In closing, we briefly review the literatures related to online convex optimization (OCO) and stochastic convex optimization. Zinkevich (2003) developed an online gradient descent algorithm to address the sequential decision problem over T periods. Compared to the offline optimal decision, this online algorithm achieves a regret bound in the order of $O(\sqrt{T})$. Several other OCO algorithms have also been developed to guarantee sub-linear regrets so that the average performance gap between the online solution and the optimal offline solution is negligible when T is sufficiently large. We refer the readers to Cesa-Bianchi and Lugosi (2006), Shalev-Shwartz (2011), and the references therein, for the technical details. These online convex optimization algorithms have been generalized to study the stochastic convex optimization problems. Under the "online to batch" conversions, the empirical solution to a sequence of online functions still provides a sub-linear regret bound guarantee to the stochastic problem with high probability (Cesa-Bianchi et al. 2002, Kakade and Tewari 2009, Shalev-Shwartz et al. 2009). In this paper, we show that the capacity configuration problem can be suitably recast as an OCO problem, after introducing dual multipliers to service level constraints. We notice that similar primal-dual algorithms were also used to solve OCO problems with long-term constraints (e.g., Mahdavi et al. 2012) and stochastic constraints (e.g., Mahdavi et al. 2013). Different from these algorithms, our approach is built on the online gradient descent algorithm (Zinkevich 2003) and the "online to batch" conversions (Shalev-Shwartz et al. 2009) directly and hence the capacity profile, which converges (almost surely) to the optimal one, can be computed easily.

3. Model Formulation and Capacity Allocation

In this section, we first formally define our problem and then address one critical decision in the model—capacity allocation—by developing an optimal allocation mechanism. We turn to the other decision—capacity configuration—in the next section.

3.1. Problem Definition and Transformation

We use a bipartite graph $\mathcal{G}$ to represent a production network of multiple plants and heterogeneous products. On one side is a set $\mathcal{J}$ of plant nodes, whereas on the other side is a set $\mathcal{I}$ of product nodes. A link connecting product node i to plant node j means that plant j is configured to produce product i. With a little abuse of notation, we also use $\mathcal{G}$ to denote the set of links in the graph. Each

product i faces a nonnegative random demand X_i and the demands $(X_i, \forall i \in \mathcal{I})$ are not necessarily i.i.d. Let $\boldsymbol{X} := (X_1, X_2, \ldots, X_{|\mathcal{I}|})$ denote the demand vector. Similarly, $\boldsymbol{S} := (S_1, S_2, \ldots, S_{|\mathcal{J}|})$ denote the capacities of the plants, and $\boldsymbol{c} := (c_1, c_2, \ldots, c_{|\mathcal{J}|})$ denote the configuration costs of the plants. Let $\beta_i \in (0,1)$ denote the fill rate required by product i, $\forall i \in \mathcal{I}$, and $\boldsymbol{\beta} := (\beta_1, \beta_2, \ldots, \beta_{|\mathcal{I}|})$.

For any subset of products $U \subseteq \mathcal{I}$, let $\Gamma(U) \subseteq \mathcal{J}$ denote the the set of production sites that support U: Plant $j \in \Gamma(U)$ if and only if j is configured to produce at least one product in the set U. We let $\mathcal{G}(U, \Gamma(U))$ denote the subgraph with nodes U and $\Gamma(U)$, and all the links between U and $\Gamma(U)$. Similarly, for any subset of plants $V \subseteq \mathcal{J}$, we denote $\Gamma(V)$ as the subset of products that can be produced by plants in V.

The supply chain planner needs to determine the capacity levels of all plants in anticipation of the product demands and in view of the fill rate targets. After demands are realized, the product managers report the requirements to the planner, who will then allocate $D_{j,i}(\boldsymbol{X}, \boldsymbol{S})$ units of capacity from plant j to product i, for all $i \in \mathcal{I}$ and $j \in \mathcal{J}$. Therefore, for any feasible capacity level and allocation mechanism, the total capacity allocated to product i must satisfy the following fill rate conditions:

$$\mathbf{E}\left[\sum_{j \in \Gamma(\{i\})} D_{j,i}(\boldsymbol{X}, \boldsymbol{S})\right] \geq \beta_i \mathbf{E}[X_i], \forall i \in \mathcal{I}. \tag{1}$$

This condition is commonly used to measure the expected fill rate, which is also known as Type-II service level, in the existing inventory literature (e.g., Chen et al. 2003, Zhong et al. 2018). Given demand information $\boldsymbol{X}$ and fill rate target $\boldsymbol{\beta}$, the feasible region for the capacity profile can be characterized as $\mathcal{S}(\boldsymbol{X}, \boldsymbol{\beta})$:

$$\mathcal{S}(\boldsymbol{X}, \boldsymbol{\beta}) := \left\{ \boldsymbol{S} \in \mathbb{R}_+^{|\mathcal{J}|} \;\middle|\; \begin{array}{l} \mathbf{E}\left[\sum_{j \in \Gamma(\{i\})} D_{j,i}(\boldsymbol{X}, \boldsymbol{S})\right] \geq \beta_i \mathbf{E}[X_i], \forall i \in \mathcal{I} \\ \sum_{j \in \Gamma(\{i\})} D_{j,i}(\boldsymbol{X}, \boldsymbol{S}) \leq X_i, \forall i \in \mathcal{I}, \forall \boldsymbol{X} \in \Omega \\ \sum_{i \in \Gamma(\{j\})} D_{j,i}(\boldsymbol{X}, \boldsymbol{S}) \leq S_j, \forall j \in \mathcal{J}, \forall \boldsymbol{X} \in \Omega \\ D_{j,i}(\boldsymbol{X}, \boldsymbol{S}) \geq 0, \forall (j,i) \in \mathcal{G}, \forall \boldsymbol{X} \in \Omega \end{array} \right\},$$

where Ω represents the set of all possible realizations for demand $\boldsymbol{X}$ and $\mathbb{R}_+^{|\mathcal{J}|}$ denotes the nonnegative orthant in the $|\mathcal{J}|$-dimensional space. The first set of constraints states that the service level requirements for all the products need to be met. The second and third set of constraints state that the allocated capacity to product i should not be more than the required amount X_i and the allocated capacity from plant j cannot exceed the available capacity level S_j.

More formally, the capacity configuration and allocation problem can be formulated as follows:

$$\begin{aligned} \text{(P1)} \quad & \min_{\boldsymbol{S}, \boldsymbol{D}(\boldsymbol{X},\boldsymbol{S})} \sum_{j\in\mathcal{J}} c_j S_j \\ & \text{s.t.} \quad \boldsymbol{S} \in \mathcal{S}(\boldsymbol{X}, \boldsymbol{\beta}) \end{aligned}$$

Note that the characterization of $\mathcal{S}(\boldsymbol{X}, \boldsymbol{\beta})$ requires specifying feasible resource allocation decision $\boldsymbol{D}(\boldsymbol{X}, \boldsymbol{S})$. Therefore, two sets of decision variables—$\boldsymbol{S}$ and $\boldsymbol{D}(\boldsymbol{X}, \boldsymbol{S})$—should be jointly optimized in Problem (P1). Since there are infinitely many ways to determine the allocation decision $\boldsymbol{D}(\boldsymbol{X}, \boldsymbol{S})$, it is generally intractable to solve Problem (P1) directly. An alternative approach is to construct an explicit allocation mechanism such that the capacity in the network is sufficient to attain the service level targets, i.e., the capacity is in the feasible region $\mathcal{S}(\boldsymbol{X}, \boldsymbol{\beta})$. However, finding an optimal allocation mechanism is a challenging problem. To do this, we first project out the allocation decisions to a set of conditions that only involves the capacity $\boldsymbol{S}$, the service targets $\boldsymbol{\beta}$, and the random demands $\boldsymbol{X}$. In this section, we focus on developing feasible capacity allocation mechanism for Problem (P1) while we address the optimal capacity configuration issue in Section 4.

3.2. Necessary and Sufficient Conditions

If there exists a feasible allocation solution $\boldsymbol{D}(\cdot,\cdot) = D_{j,i}(\cdot,\cdot), \forall (j,i) \in \mathcal{G}$, with capacity $\boldsymbol{S}$ that can achieve the fill rate requirements for all products, then for any subset of products $U \subseteq \mathcal{I}$, the expected total capacity allocated to products in U would be enough to achieve the fill rate requirements for all the products in U. That is, the following inequality must be satisfied:

$$\mathbf{E}\left[\sum_{(j,i)\in\mathcal{G}(U,\Gamma(U))} D_{j,i}(\boldsymbol{X}, \boldsymbol{S})\right] \geq \sum_{i\in U} \beta_i \mathbf{E}[X_i], \forall U \subseteq \mathcal{I}. \tag{2}$$

We can view $D_{j,i}(\boldsymbol{X}, \boldsymbol{S})$ as a flow from plant node j to product node i, given the demands $\boldsymbol{X}$ and capacity profile $\boldsymbol{S}$. According to the max-flow min-cut theorem on the subnetwork $\mathcal{G}(U, \Gamma(U))$, we have

$$\text{min-cut} = \min_{V\subseteq U}\left\{\sum_{j\in\Gamma(V)} S_j + \sum_{k\in U\setminus V} X_k\right\} = \text{max-flow} \geq \sum_{(j,i)\in\mathcal{G}(U,\Gamma(U))} D_{j,i}(\boldsymbol{X}, \boldsymbol{S}), \forall U \subseteq \mathcal{I}. \tag{3}$$

Combining conditions (2)-(3), we have

$$\mathbf{E}\left[\min_{V\subseteq U}\left\{\sum_{j\in\Gamma(V)} S_j + \sum_{k\in U\setminus V} X_k\right\}\right] \geq \sum_{i\in U} \beta_i \mathbf{E}\left[X_i\right], \forall U \subseteq \mathcal{I}. \tag{4}$$

To sum up, the set of conditions (4) holds for any feasible capacity profile $\boldsymbol{S}$ such that the desired fill rate levels can be attained. In other words, (4) is a set of *necessary* conditions for capacity

profile $\boldsymbol{S}$ to provide a feasible solution for Problem (P1). Furthermore, we can show that these necessary conditions are also *sufficient* to ensure that the service level requirements in Problem (P1) can be met for all products. We first present this result as Theorem 1 below, and then lay out the scheme of proof by explicitly constructing an allocation policy after the theorem.

THEOREM 1. *(i) The set of conditions (4) is necessary for all feasible capacity profile $\boldsymbol{S}$ to Problem (P1). (ii) If the capacity level $\boldsymbol{S}$ satisfies conditions (4), then there exists an allocation policy for Problem (P1) such that the service level requirement $\boldsymbol{\beta}$ can be attained for each and every product.*

Part (i) of Theorem 1 has been established above. In the remainder of this section, we focus on proving the second part. The proof is constructive in the sense that we explicitly construct an optimal allocation policy that can deliver the required fill rates for Problem (P1).

Note that it is challenging to check the feasibility of the fill rate constraint without a closed-form expression for Equation (1). To tackle this difficulty, a standard approach is to reformulate the expectation term in Equation (1) by sampling a set of demand scenarios $\{X_i(t)\}_{t=1}^T \sim X_i$ over T samples, where T is a sufficiently large number. For simplicity of notation, we denote the capacity that the manufacturer allocates from plant j to product i by $D_{j,i}(t)$ when it is clear that the demand vector is $\boldsymbol{X}(t)$ and the capacity vector is $\boldsymbol{S}$. We can then reformulate the fill rate constraint by the following:[4]

$$\liminf_{T\to\infty} \frac{\sum_{t=1}^T \left[\sum_{j\in\Gamma(\{i\})} D_{j,i}(t)\right]}{\sum_{t=1}^T [X_i(t)]} \geq \beta_i,\ a.s.,\ \forall i \in \mathcal{I}$$

In this way, we can cast Problem (P1) into an infinite stochastic linear programming Problem (P2), as follows:

$$\begin{aligned}
(\text{P2}) \quad \min_{\boldsymbol{S},\boldsymbol{D}(\boldsymbol{S},\boldsymbol{X}(t))} \quad & \sum_{j\in\mathcal{J}} c_j S_j \\
\text{s.t.} \quad & \liminf_{T\to\infty} \frac{\sum_{t=1}^T \left[\sum_{j\in\Gamma(\{i\})} D_{j,i}(t)\right]}{\sum_{t=1}^T [X_i(t)]} \geq \beta_i,\ a.s.,\ \forall i \in \mathcal{I} \\
& \sum_{j\in\Gamma(\{i\})} D_{j,i}(t) \leq X_i(t),\ \forall i \in \mathcal{I}, \forall t = 1,2,\ldots \\
& \sum_{i\in\Gamma(\{j\})} D_{j,i}(t) \leq S_j,\ \forall j \in \mathcal{J}, \forall t = 1,2,\ldots \\
& S_j \geq 0,\ \forall j \in \mathcal{J} \\
& D_{j,i}(t) \geq 0,\ \forall (j,i) \in \mathcal{G}, \forall t = 1,2,\ldots
\end{aligned}$$

[4] Note that the allocation policy can be sample-dependent, thus the existence of $\lim_{T\to\infty} \left\{\sum_{t=1}^T \left[\sum_{j\in\Gamma(\{i\})} D_{j,i}(t)\right] \Big/ \sum_{t=1}^T [X_i(t)]\right\}$ may not be guaranteed. Therefore, we introduce the operator $\liminf$ and require β_i to be satisfied almost surely (a.s.) in the stochastic formulation.

Note that Problems (P1) and (P2) are equivalent in the sense that their optimal capacity profiles are the same. However, they require inherently different allocation policies to deliver the required fill rates, since (P1) is a single-period problem but (P2) can be viewed as an infinite-period problem by treating each sample as an independent period. In fact, we can now solve Problem (P2) numerically via the SAA approach by choosing a fixed T,[5] to derive a near optimal capacity profile and obtain the allocation solution for the generated demand scenarios. However, the SAA approach does not explicitly construct the allocation policy, and is silent on how the capacity can be allocated on new scenarios generated from the same demand distribution. To this end, we first introduce the *max-flow debt* (MFD) policy to address the capacity allocation issue in Problem (P2),[6] and then convert the MFD policy to a feasible allocation mechanism for the single-period Problem (P1).

We present our MFD policy below. The policy involves the notion of a *priority list* $\mathcal{L}$ and its associated *lexi-cographical maximum flow* $\boldsymbol{D}(\cdot, \boldsymbol{S})$, which are detailed after the policy description. Under this policy, we formulate explicitly a priority allocation rule for each sample, using the allocation decisions for all the previous samples generated.

Max-Flow Debt (MFD) Policy for Problem (P2) with capacity profile $\boldsymbol{S}$

* *Input: T demand samples $\boldsymbol{X}(t)$, $t = 1, 2, \ldots, T$; capacity $\boldsymbol{S}$; fill rate requirement $\boldsymbol{\beta}$.*

1. Arbitrarily sort the T samples into a sequence and choose an arbitrary priority list to allocate capacity for the first sample.
2. For the $(t+1)^{\text{th}}$ sample, $t = 1, 2, \ldots, T-1$, compute weights
$$\rho_i(t+1) := \frac{\sum_{s=1}^{t} R_i(s)}{t}, \text{ where } R_i(s) := \beta_i \mathbf{E}[X_i] - \sum_{j \in \Gamma(\{i\})} D_{j,i}(s). \tag{5}$$
$R_i(s)$ is the debt associated with demand node i in the allocation for the s^{th} sample. Note that $\rho_i(t+1)$ is the average of the debts $R_i(1), R_i(2), \ldots, R_i(t)$, for the first t samples. Moreover, these weights, $\rho_i(t+1)$'s, do not depend on the demand in the current sample, $\boldsymbol{X}(t+1)$.
3. To determine the allocation for the $(t+1)^{\text{th}}$ sample, we compute a weighted max flow solution. For some arbitrarily small positive constant ϵ, let $\hat{\rho}_i(t+1) := \max\{\rho_i(t+1), \epsilon\}$ and solve
$$\begin{aligned} \text{(Q)} \max \quad & \sum_{(j,i) \in \mathcal{G}} \hat{\rho}_i(t+1) D_{j,i}(t+1) \\ \text{s.t.} \quad & \sum_{j \in \Gamma(\{i\})} D_{j,i}(t+1) \le X_i(t+1), \ \forall\ i \in \mathcal{I} \\ & \sum_{i \in \Gamma(\{j\})} D_{j,i}(t+1) \le S_j, \ \forall\ j \in \mathcal{J} \\ & D_{j,i}(t+1) \ge 0, \ \forall\ (j,i) \in \mathcal{G} \end{aligned}$$

[5] In this case, we can drop the lim inf operator in the service level constraints and replace them with sample averages.

[6] We claim that the MFD policy provides feasible resource allocation solution to Problem (P2) since the attained fill rates can meet their targets. Furthermore, the MFD policy is an optimal policy given the minimal capacity profile and the fill rate targets, but there may exist multiple optimal allocation policies.

Note that if we use $\rho_i(t+1)$ in place of the weights in the formulation, there may be multiple optimal solutions to (Q), especially when $\rho_i(t+1) \le 0$. We use $\epsilon > 0$ to break ties in this case, to ensure that the solution obtained from solving (Q) always maximizes the total flow in the network. We can choose any ϵ such that $0 < \epsilon < \min\{\rho_i(t+1) : \rho_i(t+1) > 0\}$ for this purpose. We adopt such transformation to ensure that the allocation entails a maximum flow solution, for ease of exposition, but the results in this paper do not depend on the transformation, i.e., we can simply leave the weights as $\rho_i(t+1)$'s.

4. Repeat Steps 2 and 3 until all sample allocations are found.

Let $\mathcal{L}(t+1)$ denote the priority list that ranks the products in non-increasing order of $\hat{\rho}_i(t+1)$'s, breaking ties arbitrarily if necessary, in which we assume that $\hat{\rho}_i(t+1)$'s are also modified to reflect strict ordering implied by the priority list. Suppose $\mathcal{L}(t+1) = (\mathcal{L}_1, \mathcal{L}_2, \ldots, \mathcal{L}_{|\mathcal{I}|})$, where $\mathcal{L}_l$ represents the l^{th} node in the priority list $\mathcal{L}(t+1)$. Note that a priority list is simply a permutation of the nodes in $\mathcal{I}$. In the following proposition, we show that the weighted maximum flow solution in Problem (Q) delivers the *lexi-cographical maximum flow*[7] $\boldsymbol{D}^{\mathcal{L}(t+1)}(\boldsymbol{X}(t+1), \boldsymbol{S})$ associated with the priority list $\mathcal{L}(t+1)$—i.e., $\boldsymbol{D}^{\mathcal{L}(t+1)}(\boldsymbol{X}(t+1), \boldsymbol{S})$ delivers the largest possible amount of resources to $\mathcal{L}_1$ (the first demand node in $\mathcal{L}$) until $X_{\mathcal{L}_1}(t+1)$ is fully served or the resource is depleted, followed by $\mathcal{L}_2$, and so on. A rigorous proof of the proposition is provided in Appendix B.9.

PROPOSITION 1. *In a bipartite network, the optimal weighted maximum flow derived from Problem (Q) is not only a max-flow solution to the whole network, but also a lexi-cographical maximum flow associated with the priority list* $\mathcal{L}(t+1)$.

Proposition 1 establishes a key property of the allocation rule as a result of solving Problem (Q). In this way, our MFD policy delivers the maximum fill rate to product $\mathcal{L}_1$, followed by $\mathcal{L}_2$ etc., based on available capacity in the network. In what follows, we first argue that the MFD policy, applied repeatedly on a large number of samples, provides a feasible allocation to the stochastic programming Problem (P2).

Next, we state Theorem 2, which establishes the feasibility of the MFD policy for Problem (P2). The Theorem shows that the average debt $\rho_i(T+1) \le 0$ almost surely for each and every product i, when T goes to infinity. We focus on analyzing $\rho_i^+(T+1) := \max\{\rho_i(T+1), 0\}$ for every $i \in \mathcal{I}$, and consider the penalty function

$$\|\boldsymbol{\rho}^+\|_2^2 = \sum_{i=1}^{|\mathcal{I}|} \rho_i^2 1(\rho_i > 0).$$

[7] Note that this is not the same as the classical lexi-cographical maximum flow problem solved in Megiddo (1977), which defines the lexi-cographic orders for both source and sink nodes in general networks. In this paper, the lexi-cographic order refers to the demand side in bipartite networks.

To state the Theorem, we define the following four constants, which are independent of T:

$$\Lambda := \sum_{j\in\mathcal{J}} S_j,\ \Upsilon := \max_{i\in\mathcal{I}} \left\{ \sum_{j\in\Gamma(\{i\})} S_j \right\},$$
$$\Lambda' := \max\left\{ \sum_{i\in\mathcal{I}} \beta_i \mathbf{E}[X_i], \sum_{j\in\mathcal{J}} S_j \right\},\ \Upsilon' := \max_{i\in\mathcal{I}} \left\{ \max\left\{ \beta_i \mathbf{E}[X_i], \sum_{j\in\Gamma(\{i\})} S_j \right\} \right\}. \tag{6}$$

THEOREM 2. *Suppose we are given a vector of capacities $\boldsymbol{S}$ that satisfies conditions (4). Then the average debt $\boldsymbol{\rho}(T+1)$ under the MFD policy satisfies the following non-asymptotic convergence guarantees in expectation:*

$$\mathbf{E}\left[\|\boldsymbol{\rho}^+(T+1)\|_2^2\right] \le \frac{\Upsilon\Lambda(1+\log T)}{T}. \tag{7}$$

The policy also satisfies the following non-asymptotic convergence with high probability: For any fixed $\delta \in (0,1)$, we have

$$\mathbf{P}\left(\|\boldsymbol{\rho}^+(T+1)\|_2^2 \le \frac{1}{\sqrt{T}} \left\{ \frac{\Upsilon\Lambda(1+\log T)}{\sqrt{T}} + 4\Upsilon'\Lambda' \sqrt{2\log\left(\frac{1}{\delta}\right)} \right\} \right) \ge 1-\delta, \tag{8}$$

where constants $\Upsilon, \Lambda, \Upsilon', $ and Λ' are independent of T.

Assuming that the array of capacity levels satisfies conditions (4), Theorem 2 asserts that the positive part $\rho_i^+(T)$ of the debt under MFD policy converges to zero for each and every product i, as T tends to infinity.

The proof of Theorem 2 involves tracking the rate of decrease in $\|\boldsymbol{\rho}^+(T)\|_2^2$, as T increases. Observe that, for each sample $T+1$, the MFD policy prioritizes the allocation of capacity to the demand node with the highest debt incurred in the previous samples $1, 2, \ldots, T$. The prioritization serves to compensate for the demand nodes that are still under-supplied after T samples. Harnessing the property of the optimization problem (Q) and the strong convexity of the squared Euclidean norm, we demonstrate the following non-asymptotic convergence:

$$\|\boldsymbol{\rho}^+(T+1)\|_2^2 \le \left(1-\frac{1}{T}\right)\|\boldsymbol{\rho}^+(T)\|_2^2 + O\left(\frac{1}{T^2}\right) + \mathrm{Err}_T, \tag{9}$$

where Err_T is a stochastic error term that has mean zero and converges to 0 almost surely as T tends to infinity, and the $O(\cdot)$ notation hides a constant independent of T. Theorem 2 is then proved by unraveling the recursion (9) for an upper bound on $\|\boldsymbol{\rho}^+(T+1)\|_2^2$.

After highlighting the analysis of Theorem 2, we provide the following remarks on the theorem. Theorem 2 implies that the fill rate targets for all products in Problem (P2) can be met by performing the MFD policy, for any feasible capacity profiles. In this way, the MFD policy can be used to check the feasibility of any given capacity profile for Problem (P2).

In addition, we remark that Theorem 2 generalizes the results in Zhong et al. (2018) in two ways. First, the capacity allocation policy in Zhong et al. (2018) only applies for single-supplier settings, while Theorem 2 applies for multiple-supplier settings. Second, we establish non-asymptotic rates of convergence in expectation and with high probability for $\boldsymbol{\rho}(T)$ as T increases, and such convergence results generalize the asymptotic convergence established in Zhong et al. (2018).

Finally, in the case when the capacity profile is insufficient to deliver the required fill-rate targets to all customers, the MFD policy can still be executed, but converges to a different set of fill rate performances for the customers. In Appendix A, we characterize the performance of the MFD policy in this environment.

3.3. Randomized Anticipative Allocation Policy

In this subsection, we develop an optimal allocation mechanism to solve the single-period problem (P1) by sampling a priority list used in the MFD policy randomly. A crucial property we use in the analysis is that the allocation priority $\mathcal{L}(t)$ used in Problem (Q) in the MFD policy does not depend on the demand $\boldsymbol{X}(t)$.

In the capacity pooling literatures, Alptekindğlu et al. (2013) defined two classes of priority allocation policies to address the capacity pooling problem under Type-I service level constraint: 1) ***Anticipative priority policy:*** The priority list for a demand scenario is constructed without using the realized demand information in that period. For example, a straightforward anticipative policy is to prioritize products according to their service level requirements; 2) ***Responsive priority policy:*** The priority list for a demand scenario is formed using the demand information realized in that period. For example, the *small-demand-high-priority* policy is a commonly used responsive policy—i.e., the product with smaller demand is filled first and then the one with larger demand can be served if there are any remaining resources. We refer to Zhong et al. (2018) and Lyu et al. (2017) for detailed comparisons of anticipative policy and responsive policy.

Intuitively, these two types of policies differ in whether the demand realization is used to prioritize the allocation order or not. Since responsive policy requires more demand information, such policy may perform better than the anticipative policy. In the MFD policy, the allocation list obtained for the t^{th} sample is constructed purely using the allocation decisions on the previous $(t-1)$ samples, but does not rely on the demand in the t^{th} sample, $\boldsymbol{X}(t)$. Therefore, the MFD policy is indeed *anticipative*. This property is crucial to derive the randomized allocation policy for the single-period Problem (P1) given a capacity profile $\boldsymbol{S}$, i.e., the allocation priority list is randomly selected from a priority list pool, obtained from the MFD policy on T random samples. We describe the construction in the following Algorithm 1.

We describe a concrete example below to illustrate how Algorithm 1 works.

Algorithm 1 Randomized Allocation Policy for Problem (P1) with capacity profile $\boldsymbol{S}$

* *Input: Distribution of demand $\boldsymbol{X}$; capacity $\boldsymbol{S}$; fill rate requirement $\boldsymbol{\beta}$.*

1. Generate T demand samples $\boldsymbol{X}(t)$ independently according to the distribution of demand $\boldsymbol{X}$, where $t = 1, 2, \ldots, T$, and T is a sufficiently large number.
2. Input the generated T demand samples $\{\boldsymbol{X}(t)\}_{t=1}^{T}$, capacity $\boldsymbol{S}$ and fill rate requirement $\boldsymbol{\beta}$ into the MFD policy to generate T allocation priority lists. Denote the allocation priority list for sample t as $\mathcal{L}(t)$. Since the allocation rule is anticipative, $\mathcal{L}(t)$ is a randomized list that is independent of the demand information $\boldsymbol{X}(t)$.
3. To solve the single-period Problem (P1) with given $\boldsymbol{S}$, pick a priority list from the pool of priority lists $\{\mathcal{L}(t) : t = 1, 2, \ldots, T\}$, each with equal probability, and solve a lexi-cographical max flow problem in the network, based on the realization of $\boldsymbol{X}$.

Example 1. Consider a "Z" production network, with two plants $\{P, Q\}$—each having capacity 50 and 80, respectively—and two products $\{A, B\}$—each facing i.i.d. uniform demands on $[0, 100]$. Product A can be produced at both plants, but Product B can be produced only at Plant Q. Suppose that A is a critical product for the customer, and has an expected fill rate target of $\beta_A = 0.96$; while Product B requires a lower expected fill rate of $\beta_B = 0.90$.

If we serve demand solely following the priority order of (A, B), although we can guarantee a 100% fill rate to Product A, the expected fill rate delivered to B is less than 0.87, not meeting the requirement. If we follow another fixed priority order of (B, A), we certainly fail to serve Product A, which can only receive an expected fill rate of 0.908, while B is over-served with an expected fill rate of 0.960. We can see that in this case, fixed priority rules fail to deliver the fill rate targets.

Following the MFD policy with $T = 10^6$ and choosing the priority order (A, B) in the first sample, we observe that the priority order of (A, B) is used in 60.73% samples, and the priority order of (B, A) is used in the remaining samples. Therefore, based on Algorithm 1, we obtain a randomized allocation policy, which serves the demands in the order (A, B) with probability 0.6073, and (B, A) with probability 0.3927. In this way, we can satisfy the expected fill rate requirements for both products—Product A receives an expected fill rate of 0.964 and Product B gets 0.904. ■

Based on Theorem 2 and Algorithm 1, we can conclude the following proposition.

Proposition 2. *If conditions* (4) *are satisfied by the capacity $\boldsymbol{S}$, Algorithm 1 provides a feasible capacity allocation solution to the single-period Problem (P1).*

To this end, we claim that the sufficiency of conditions (4) for all feasible capacity profiles to Problem (P1) is demonstrated and hence Part (ii) of Theorem 1 is proved. Meanwhile, given any

capacity profile, our randomized allocation policy can be used to determine whether the capacity profile falls in the feasible region $\mathcal{S}(\boldsymbol{X}, \boldsymbol{\beta})$ or not. We note that although we have explicitly constructed an optimal allocation policy, Algorithm 1, there may exist multiple optimal allocation policies for Problem (P1).

The anticipative property of our allocation policy refers to the fact that the priority list on which product to serve first is constructed without using the realized demand information. In fact, the actual allocation quantities depend on the demand realization. Given the priority list and demand information, we solve the optimization model (Q) to determine the allocation quantities.

The observation that the allocation mechanism is anticipative has other important ramifications. For instance, a related interesting observation is that our allocation mechanism is strategy-proof in the following sense: In a more general formulation in which each product manager is privately informed of the demand realization for her product, she does not have incentive to misreport her demand since the randomized allocation priority list is constructed independent of the reported (realized) demand values. Therefore, truth telling (i.e., reporting the actual demand) is a dominant strategy in this setting. We summarize this result as follows and provide more details in Appendix B.10.

PROPOSITION 3. *The randomized allocation mechanism is strategy-proof in the sense that truth telling is a dominant strategy for each product manager.*

3.4. An Application to Product Upgrades

Before proceeding to address the capacity configuration problem in the next section, we first demonstrate an immediate application of the necessary and sufficient conditions to a multi-item newsvendor problem in this subsection.

As a result of technological upgrading over time, multiple versions of a single product are commonly seen in practice, especially in spare parts management. For example, a hard disk manufacturer can use a hard disk with larger space (1 TB) and higher-speed port (USB 3.0) to replace the faulty old hard disk (320 GB and USB 2.0) with the same physical size, if it is still in warranty period. In general, backward compatibility is allowed while forward compatibility is not, i.e., newer-version (upgraded) items can be used to fulfill the demand for lower-version items but the reverse is either infeasible or not acceptable. A relevant and important question is then how product substitutions affect the revenue streams and capacity configuration in the supply chain. We address this problem using the main results derived in Theorem 1.

We consider a system with N product and demand types indexed by $\{1, 2, \ldots, N\}$. Each product j can be used to meet demand of type i provided $j \leq i$. Products with lower indices are of higher quality or from newer version and thus can be used to satisfy the demand for those with higher

indices. The supply for each product j is denoted by S_j, with unit cost c_j and $c_1 > c_2 > \cdots > c_N$. Similarly, the profit from demand type i is p_i, and $p_1 > p_2 > \cdots > p_N$.

What is the optimal S_j to be installed in the system, if product substitution with upgrades are allowed? This is a difficult question to address in general, since the performance also depends on the product substitution (i.e., resource allocation) decisions. Instead of answering this question directly, we first fix S_j, and find the optimal fill rates that the system can support to maximize the profits accrued. We find the optimal S_j next after understanding this trade-off between the capacity configuration and the optimal fill rate performance.

Suppose that the fill rate target for each type i is β_i. Our objective function can be written as

$$\max_{\boldsymbol{\beta},\boldsymbol{S}} \left\{ -\sum_{j=1}^{N} c_j S_j + \sum_{i=1}^{N} p_i \beta_i \mathbf{E}\left[X_i\right] \right\}, \tag{10}$$

where X_i denotes the stochastic demand of type i product. Let $U = \{i_1, i_2, \ldots, i_{|U|}\} \subseteq \{1, 2, \ldots, N\}$ be a subset of demand types, and assume $i_1 < i_2 < \cdots < i_{|U|}$ without loss of generality. Denote $\{i_0\} = \emptyset$. Let $\Gamma(\{i\})$ denote the product set that can be used to meet the demand of type i. Note that $\Gamma(\{i\}) = \{1, 2, \ldots, i\} \subset \Gamma(\{k\})$ if $i < k$. This allows us to rewrite conditions (4) to

$$\mathbf{E}\left[\min_{m \in \{0,1,\ldots,|U|\}} \left\{ \sum_{j \in \Gamma(\{i_m\})} S_j + \sum_{k=m+1}^{|U|} X_{i_k} \right\} \right] \geq \sum_{i \in U} \beta_i \mathbf{E}\left[X_i\right], \forall U = \{i_1, i_2, \ldots, i_{|U|}\} \subseteq \{1, 2, \ldots, N\}.$$

Although there are exponentially many such constraints, optimizing over β_i's, while keeping S_j's fixed, is an easy problem, since the function $\mathbf{E}[\min_{m \in \{0,1,\ldots,|U|\}} \{\sum_{j \in \Gamma(i_m)} S_j + \sum_{k=m+1}^{|U|} X_{i_k}\}]$ is a monotone submodular function in U.[8] Therefore, the optimal $\beta^*(\boldsymbol{S})$ can be obtained by solving the following system of equations:

$$\mathbf{E}\left[\min_{m \in \{0,1,\ldots,k\}} \left\{ \sum_{j=1}^{m} S_j + \sum_{i=m+1}^{k} X_i \right\} \right] = \sum_{i=1}^{k} \beta_i^*(\boldsymbol{S}) \mathbf{E}\left[X_i\right], \forall k = 1, 2, \ldots, N, \tag{11}$$

This is equivalent to

$$\mathbf{E}\left[\max_{m \in \{0,1,\ldots,k\}} \left\{ \sum_{i=1}^{m} X_i - \sum_{j=1}^{m} S_j \right\} \right] = \sum_{i=1}^{k} (1 - \beta_i^*(\boldsymbol{S})) \mathbf{E}\left[X_i\right], \forall k = 1, 2, \ldots, N. \tag{12}$$

This reduces conditions (4) to essentially N constraints on a set of maximum partial sums.

[8] This is a classical result in optimization. For any two subsets U and V, consider a max flow $x(U \cap V)$ in $G(\Gamma(U \cap V), U \cap V)$. This max flow solution can be modified with augmenting flows to arrive at a max flow solution $x(U \cup V)$ to $G(\Gamma(U \cup V), U \cup V)$. The total flows out of nodes in $U \cap V$ must be the same in both $x(U \cap V)$ and $x(U \cup V)$, since the initial flow $x(U \cap V)$ is optimal for the network $G(\Gamma(U \cap V), U \cap V)$. Let $x(U)$ and $x(V)$ be the solution obtained from $x(U \cup V)$ by restricting the flows to arcs in $G(\Gamma(U), U)$ and $G(\Gamma(V), V)$ respectively. They overlap on a common set of flows with values at least as large as the flows in $x(U \cap V)$. A complete proof of the submodularity property mentioned above can be found in Appendix B.8.

Let $p_{N+1}=0$. Problem (10) can be written in the following way:

$$\begin{aligned}
&\max_{\boldsymbol{S}}\left\{-\sum_{j=1}^{N}c_jS_j+\sum_{i=1}^{N}p_i\mathbf{E}\left[X_i\right]-\sum_{i=1}^{N}p_i(1-\beta_i^*(\boldsymbol{S}))\mathbf{E}\left[X_i\right]\right\}\\
=&\max_{\boldsymbol{S}}\left\{-\sum_{j=1}^{N}c_jS_j+\sum_{i=1}^{N}p_i\mathbf{E}\left[X_i\right]-\sum_{k=1}^{N}(p_k-p_{k+1})\sum_{i=1}^{k}(1-\beta_i^*(\boldsymbol{S}))\mathbf{E}\left[X_i\right]\right\}\\
=&\max_{\boldsymbol{S}}\left\{-\sum_{j=1}^{N}c_jS_j+\sum_{i=1}^{N}p_i\mathbf{E}\left[X_i\right]-\sum_{k=1}^{N}(p_k-p_{k+1})\mathbf{E}\left[\max_{m\in\{0,1,\ldots,k\}}\left\{\sum_{j=1}^{m}(X_j-S_j)\right\}\right]\right\}
\end{aligned}$$

We can then use the first order conditions to pin down the optimal value of S_j.

PROPOSITION 4. *In a flexible production system with upgrades, if $S_j^*>0,\forall j=1,2,\ldots,N$, then the following condition holds:*

$$c_j=\sum_{i=j}^{N}(p_i-p_{i+1})\mathbf{P}\left(\max_{m:j\le m\le i}\left\{\sum_{k=1}^{m}(X_k-S_k^*)\right\}=\max_{m:m\le i}\left\{\sum_{k=1}^{m}(X_k-S_k^*)\right\}\right),\forall j=1,2,\ldots,N \quad (13)$$

For $N=2$, the above conditions can be written more succinctly as:

$$\begin{aligned}
c_2&=p_2\mathbf{P}\left(X_1+X_2-S_1^*-S_2^*=\max\left\{0,X_1-S_1^*,X_1+X_2-S_1^*-S_2^*\right\}\right)\\
&=p_2\mathbf{P}\left(X_2\ge S_2^*,X_1+X_2\ge S_1^*+S_2^*\right),\text{ and}\\
c_1&=(p_1-p_2)\mathbf{P}\left(X_1\ge S_1^*\right)+p_2\mathbf{P}\left(\left\{X_1\ge S_1^*\right\}\text{ or }\left\{X_1+X_2\ge S_1^*+S_2^*\right\}\right)\\
&=p_1\mathbf{P}\left(X_1\ge S_1^*\right)+p_2\mathbf{P}\left(X_1<S_1^*,X_1+X_2\ge S_1^*+S_2^*\right).
\end{aligned}$$

These conditions generalize the critical fractile condition for the classical newsvendor model (cf. Porteus 1990).

4. Capacity Configuration

In the previous section, we have addressed the capacity allocation problem in our setting: Given sufficient capacity levels $\boldsymbol{S}$, how to allocate them to achieve the expected fill rate targets? This can be used to construct a search procedure to determine the appropriate capacity configuration, if the search space is not too large or low dimensional. In this section, we develop an efficient method to compute the minimum capacity levels $\boldsymbol{S}$ given the expected fill rate targets. To this end, we develop a two-stage online gradient descent algorithm to solve the capacity configuration problem. Leveraging on the online gradient descent algorithm (cf. Zinkevich 2003) and the "online to batch" conversions (cf. Shalev-Shwartz et al. 2009), we demonstrate that the capacity profile obtained by this approach converges (almost surely) to the optimal solution.

For the single-period Model (P1), we consider its Lagrangian dual formulation by introducing the Lagrangian dual multiplier $\boldsymbol{\lambda}$ in correspondence to the expected fill rate constraints, which can be formulated as:

$$\begin{aligned}
\text{(P3)}\quad \max_{\boldsymbol{\lambda}\geq \mathbf{0}} \min_{\boldsymbol{S},\boldsymbol{D}(\boldsymbol{X},\boldsymbol{S})} \; & \sum_{j\in\mathcal{J}} c_j S_j + \sum_{i\in\mathcal{I}} \lambda_i \left\{ \beta_i \mathbf{E}[X_i] - \mathbf{E}\left[\sum_{j\in\Gamma(\{i\})} D_{j,i}(\boldsymbol{X},\boldsymbol{S}) \right] \right\} \\
\text{s.t.} \quad & \sum_{j\in\Gamma(\{i\})} D_{j,i}(\boldsymbol{X},\boldsymbol{S}) \leq X_i,\ \forall i\in\mathcal{I}, \forall \boldsymbol{X}\in\Omega \\
& \sum_{i\in\Gamma(\{j\})} D_{j,i}(\boldsymbol{X},\boldsymbol{S}) \leq S_j,\ \forall j\in\mathcal{J}, \forall \boldsymbol{X}\in\Omega \\
& S_j \geq 0,\ \forall j\in\mathcal{J} \\
& D_{j,i}(\boldsymbol{X},\boldsymbol{S}) \geq 0,\ \forall (j,i)\in\mathcal{G}, \forall \boldsymbol{X}\in\Omega
\end{aligned}$$

To solve this model, one classic approach is to alternatively minimize the inner optimization problem with respect to $\boldsymbol{S}$ and to maximize the outer optimization problem with respect to $\boldsymbol{\lambda}$ (Nocedal and Wright 2006). Stemming from this idea, we provide a two-stage algorithm to update the dual multiplier $\boldsymbol{\lambda}$ and capacity profile $\boldsymbol{S}$ iteratively until the capacity profile converges. Specifically, at the $(k)^{\text{th}}$ iteration, we (1) use an OCO algorithm to derive the capacity profile $\boldsymbol{S}^k$ given $\boldsymbol{\lambda}^k \in \mathbb{R}_+^{|\mathcal{I}|}$; and (2) use gradient ascent algorithm[9] to update the dual multiplier for the $(k+1)^{\text{th}}$ iteration.

More concretely, for the above step (1), we address the expectation terms in the inner minimization problem of Problem (P3) by sampling T independent demand scenarios $\{X_i(t)\}_{t=1}^T \sim X_i$ for a sufficiently large T. In this way, at the k^{th} iteration when the dual multipliers are λ_i^k, we formulate a new stochastic programming problem as follows:

$$\begin{aligned}
\text{(P4)}\quad \min_{\boldsymbol{S}\geq\mathbf{0}} \sum_{j\in\mathcal{J}} c_j S_j + \min_{\boldsymbol{D}(t)} \; & \sum_{i\in\mathcal{I}} \lambda_i^k \left[\frac{1}{T} \sum_{t=1}^{T} \left(\beta_i X_i(t) - \sum_{j\in\Gamma(\{i\})} D_{j,i}(t) \right) \right] \\
\text{s.t.} \quad & \sum_{j\in\Gamma(\{i\})} D_{j,i}(t) \leq X_i(t),\ \forall i\in\mathcal{I}, \forall t = 1,2,\ldots,T \\
& \sum_{i\in\Gamma(\{j\})} D_{j,i}(t) \leq S_j,\ \forall j\in\mathcal{J}, \forall t = 1,2,\ldots,T \\
& D_{j,i}(t) \geq 0,\ \forall (j,i)\in\mathcal{G}, \forall t = 1,2,\ldots,T
\end{aligned}$$

Let $\hat{\boldsymbol{S}}^{k*}$ denote the optimal $\boldsymbol{S}$ for the above problem, which approximates the true optimal solution to the inner minimization problem of Problem (P3), denoted as $\boldsymbol{S}^{k*}$, using T samples. However, we note that solving Problem (P4) requires addressing all the demand scenarios simultaneously in a large-scale linear programming problem for a large T. To circumvent this computational difficulty, we adopt some OCO techniques, using the observation that Problem (P4) is convex in $\boldsymbol{S}$.

[9] To avoid confusion, we use gradient "ascent" algorithm to update the dual multiplier for the maximization problem while we use gradient "descent" algorithm to update the primal decision variable for the minimization problem.

Observe that given $\boldsymbol{S}^k$, the inner minimization over $\boldsymbol{D}(t)$ in Problem (P4) is separable for each demand scenario. Define

$$\begin{aligned}(\text{P5})\quad f_t^k(\boldsymbol{S},\boldsymbol{X}(t)) := \sum_{j\in\mathcal{J}} c_j S_j + \min_{\boldsymbol{D}(t)} \sum_{i\in\mathcal{I}} \lambda_i^k \left(\beta_i X_i(t) - \sum_{j\in\Gamma(\{i\})} D_{j,i}(t)\right)\\ \text{s.t.} \sum_{j\in\Gamma(\{i\})} D_{j,i}(t) \le X_i(t),\ \forall i\in\mathcal{I}\\ \sum_{i\in\Gamma(\{j\})} D_{j,i}(t) \le S_j,\ \forall j\in\mathcal{J}\\ D_{j,i}(t)\ge 0,\ \forall (j,i)\in\mathcal{G}\end{aligned}$$

Then $\hat{\boldsymbol{S}}^{k*} = \arg\min_{\boldsymbol{S}} \sum_{t=1}^T f_t^k(\boldsymbol{S},\boldsymbol{X}(t))$. Instead of finding $\hat{\boldsymbol{S}}^{k*}$ directly, we use an OCO algorithm to develop a feasible capacity profile $\boldsymbol{S}^k(t-1)$ to Problem (P5), for each scenario $t=1,2,\ldots,T$, and use these to construct a near optimal approximation for $\hat{\boldsymbol{S}}^{k*}$. More formally, we aim to minimize the cumulative regret from using $\{\boldsymbol{S}^k(t)\}_{t=0}^{T-1}$ over T scenarios, defined as:

$$\text{Regret}(T) := \sum_{t=1}^T f_t^k(\boldsymbol{S}^k(t-1),\boldsymbol{X}(t)) - \sum_{t=1}^T f_t^k(\hat{\boldsymbol{S}}^{k*},\boldsymbol{X}(t)).$$

A classical result by Zinkevich (2003) is that the cumulative regret of the online gradient descent algorithm satisfies the following non-asymptotic bound:

$$\sum_{t=1}^T f_t^k(\boldsymbol{S}^k(t-1),\boldsymbol{X}(t)) \le \sum_{t=1}^T f_t^k(\hat{\boldsymbol{S}}^{k*},\boldsymbol{X}(t)) + O(\sqrt{T}). \tag{14}$$

Furthermore, let $g^k(\boldsymbol{S}) := \mathbf{E}_{X(t)}\left[f_t^k(\boldsymbol{S},\boldsymbol{X}(t))\right]$. Then $\boldsymbol{S}^{k*} = \arg\min_{\boldsymbol{S}} g^k(\boldsymbol{S})$. Define the average solution to the sequence of online problems as $\bar{\boldsymbol{S}}^k := (1/T)\sum_{t=1}^T \boldsymbol{S}^k(t-1)$. Note that $g^k(\boldsymbol{S})$ is independent of t since $X(t)$'s are identically distributed. Theorem 2 in Cesa-Bianchi et al. (2002) asserts that for all $\delta\in(0,1)$, the inequality

$$g^k(\bar{\boldsymbol{S}}^k) - \frac{\sum_{t=1}^T f_t^k(\boldsymbol{S}^k(t-1),\boldsymbol{X}(t))}{T} \le O\left(\sqrt{\frac{1}{T}\log\left(\frac{1}{\delta}\right)}\right) \tag{15}$$

holds with probability at least $(1-\delta)$.

Note that the demand for each product i is i.i.d. across different scenarios (i.e.,$\{X_i(t)\}_{t=1}^T \sim X_i$), it follows that the SAA solution $\hat{\boldsymbol{S}}^{k*}$ coverages to the optimal solution $\boldsymbol{S}^{k*}$ (cf. Shapiro et al. 2009, Proposition 5.2).

Proposition 5. *Under the conditions that the demand sequences ($\{X_i(t)\}_{t=1}^T \sim X_i$) for each product i are i.i.d. and $|g^k(\boldsymbol{S})|<\infty$ for any feasible and finite capacity profile $\boldsymbol{S}$, we have*

$$\lim_{T\to\infty} \frac{1}{T}\sum_{t=1}^T f_t^k(\hat{\boldsymbol{S}}^{k*},\boldsymbol{X}(t)) \to g^k(\boldsymbol{S}^{k*}),\ a.s.$$

Combining Equation (14) and (15) with Proposition 5, we can claim that the average capacity $\bar{\boldsymbol{S}}^k$ generated from the OCO problem converges (almost surely) to the optimal solution to the original stochastic problem—the inner minimization problem of Problem (P3). In addition, Equation (14) and (15) also imply that the average capacity $\bar{\boldsymbol{S}}^k$ converges to the SAA solution $\hat{\boldsymbol{S}}^{k*}$ with a sub-linear regret guarantee.

Let $D^k_{j,i}(t)$'s denote the optimal allocation decision obtained while computing $f^k_t(\boldsymbol{S}^k(t-1), \boldsymbol{X}(t))$. After we obtain the capacity profile $\bar{\boldsymbol{S}}^k$ in the k^{th} iteration, we update the Lagrangian multiplier $\boldsymbol{\lambda}^{k+1}$ using the classical gradient ascent algorithm:

$$\lambda_i^{k+1} := \lambda_i^k + \kappa_k \left(\beta_i \mathbf{E}\left[X_i\right] - \sum_{j \in \Gamma(\{i\})} D^k_{j,i} \right), \tag{16}$$

where $\kappa_k := 1/\sqrt{k}$ represents the step size, and $D^k_{j,i} := (1/T)\sum_{t=1}^{T} D^k_{j,i}(t)$ denotes the expected capacity allocation quantities in the k^{th} iteration.

To sum up, we sketch the main steps of this two-stage online gradient descent approach in Algorithm 2.

Algorithm 2 Optimal Capacity Configuration for Problem (P1)

* *Input:* $\boldsymbol{\lambda}^1 = \mathbf{0}$ *and* $\boldsymbol{S}^0 = \mathbf{0}$.

1. For $k = 1, 2, \ldots,$ do the following:
   - Compute the capacity profile $\bar{\boldsymbol{S}}^k$ and the expected capacity allocation quantity $\boldsymbol{D}^k$ using the online gradient descent algorithm (Algorithm 3 in Appendix C);
   - Update the dual multiplier $\boldsymbol{\lambda}^{k+1}$ based on (16).
2. Terminate the above process when $\max_{i \in \mathcal{I}} \left\{ |\lambda_i^{k+1} - \lambda_i^k| \right\} \leq \epsilon$, where $\epsilon > 0$ is a predetermined tolerance threshold. We choose $\epsilon = 0.01$ in our numerical studies. Then the capacity profile $\bar{\boldsymbol{S}}^k$ is the desired capacity profile for Problem (P1).

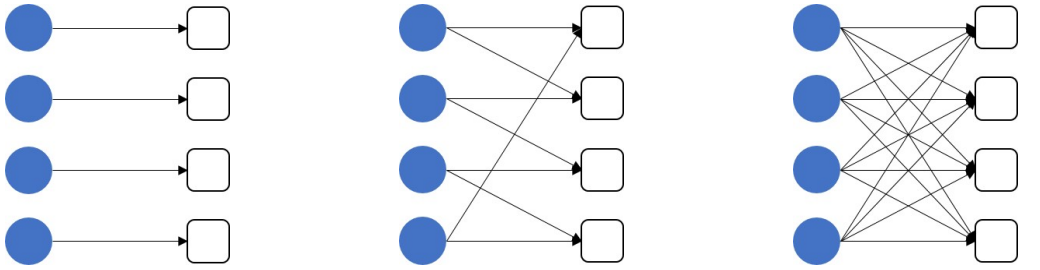
For completeness, we provide the online gradient descent algorithm used in Step 1 as Algorithm 3 in Appendix C. In addition, we discuss the technical conditions related to the regret bounds of Algorithm 3—which are satisfied in our problem—and provide a numerical example to demonstrate the performance of this algorithm in Appendix C.

5. Application: Optimal Capacity in the Long-Chain System

In this section, we investigate the connection between the flexibility structure and the configured optimal capacity. Figure 1 shows various classes of flexible network often used in the study of flexible production network. In the process flexibility literature, a k-chain network is an N by N bipartite graph in which each plant node i is used to serve product nodes $i, i+1, \ldots, i+k-1$ (modulo N).

When $k = 2$, the 2-chain network is also known as long-chain. Jordan and Graves (1995) showed that the long-chain can serve almost the same amount of demand as the full-flexibility network, in a balanced and symmetrical production system. In this section, we provide another evidence for a similar phenomenon in the capacity configuration problem by numerically showing that the optimal capacity needed to meet the fill rate requirements for a long-chain is almost the same as that for a full-flexibility network.

Figure 1 Flexibility configuration: Solid blue circles represent plant nodes and the squares represent product nodes. A link connecting a circle and square implies that the product can be produced by the plant. In a dedicated system, each product can only be produced at one plant and each plant can also only produce one product. In the partial flexibility system, each product can be produced at multiple sites and each plant can also produce different types of products. In the full-flexibility system, each product can be produced at any plant and each plant can produce all the products.



(a) Dedicated system (b) Partial-flexibility system (2-Chain) (c) Full-flexibility system

For ease of exposition, we restrict our explorations to the class of k-chain production network with N products and N plants. We assume the demands faced by products share independent and identical distribution with mean μ and require identical fill rate target β. Consequently, according to the optimality conditions (4), it is straightforward to see from symmetry that the optimal capacity levels for all plants in the k-chain network are identical. In this case, we can compute the optimal capacity configuration using a much simpler approach—performing a bisection search algorithm on the optimal capacity level, followed by implementing the MFD policy to check whether the capacity is sufficient to deliver the required fill rate services to all the products or not.

5.1. Numerical Analysis

We first consider N ($N = 4, 8, \ldots, 20$) products with i.i.d. demands and identical service level requirements (0.99) in the k-chain network ($k = 1, 2, 3, 4$) and full-flexibility network. We assume the demands are normally distributed with parameters $\mathcal{N}(10, 3^2)$.

To examine the performance of k-chain network, we define the capacity gap as:

$$\frac{(\text{Total Capacity in the } k\text{-chain Network}) - (\text{Total Capacity in the Full Flexibility Network})}{(\text{Total Capacity in the Full Flexibility Network})} \times 100\%.$$

Table 1 Comparison of optimal capacity profiles in the k-chain and full-flexibility networks.

	Total Capacity Level					Capacity Gap (%)			
N	$k=1$	$k=2$	$k=3$	$k=4$	Full Flex.	$k=1$	$k=2$	$k=3$	$k=4$
4	57.58	47.02	47.02	47.02	47.02	22.47	0.00	0.00	0.00
8	114.70	88.35	88.27	88.27	88.27	29.94	0.09	0.00	0.00
12	172.40	130.33	129.63	129.63	129.57	33.05	0.59	0.05	0.05
16	228.93	173.54	171.24	171.24	171.16	33.75	1.39	0.05	0.05
20	286.16	215.64	211.35	211.32	211.20	35.49	2.10	0.07	0.06

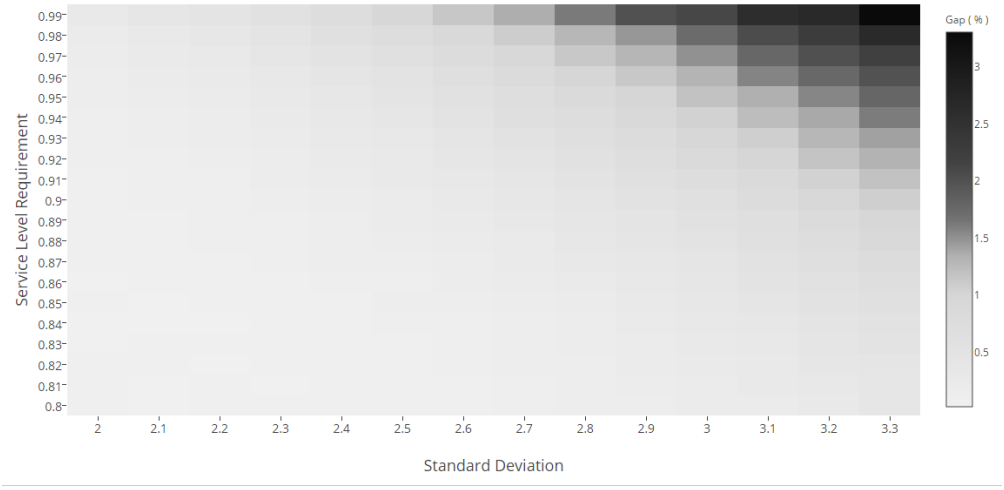
As shown in Table 1, the total capacity in the dedicated system ($k=1$) is much higher than that for the full-flexibility network and the capacity gap between these two systems also increases when the network size becomes larger. Note that in the k-chain network, there are k out of N plants to serve each product and the ratio $\frac{k}{N}$ increases in k but decreases in N. Therefore, in a large size network, the proportion of plants to serve each product diminishes and we expect a relatively large capacity gap. Surprisingly, for the long-chain ($k=2$), the total capacity required is extremely close to that for the full-flexibility network, especially in the small size networks. However, moving from long-chain network towards 3-chain and 4-chain networks, the marginal capacity reduction is negligible.

Next, we explore how the demand distribution and service level requirement affect the capacity gap between the long-chain network and full-flexibility network. We consider i.i.d. normal demands with mean $=10$ for network size $N=20$. We vary the demand standard deviation from 2.0 to 3.3 and the homogeneous fill rate target from 0.80 to 0.99. From Figure 2(a), we observe that the gap increases in both standard deviation and fill rate target, but never more than 3.5% in the range of values tested. We also consider different service level targets for different demand nodes. Similarly, we generate a sequence of i.i.d. normal demands with mean $=10$ and standard deviation $=3.3$ for network size $N=20$. To represent different degrees of differentiation in fill rate requirements, we choose the fill rate targets from 7 different intervals. For each interval, denoted as $[a,b]$, we randomly generate 50 sets of fill rate targets from the uniform distribution $[a,b]$. As shown in Figure 2(b), the capacity gaps are around 0.9% on average for all instances. Interestingly, as the mean fill rate targets, $(a+b)/2$, for different intervals are identical, the mean performance gaps under different fill rate intervals are also similar. With less variability in the fill rate requirements, the performance gaps become more concentrated.

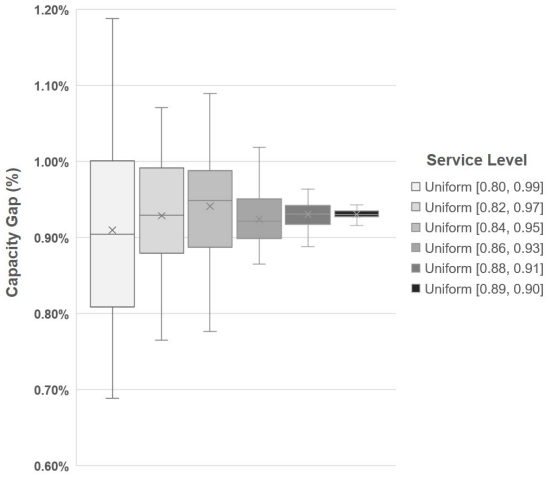
Furthermore, we also investigate the impact of demand correlation on the capacity gap between the long-chain network and full-flexibility network. We consider identical normal demands with mean $=10$ and standard deviation $=3.3$ for network size $N=20$. We force every pair of demand nodes to share the same correlation coefficient,[10] ranging from -0.05 to 0.8. The capacity gaps

[10] In a long-chain network with $N>2$, it is hard to generate strongly positively or negatively correlated demand samples. Therefore, we only consider a moderate range of correlation coefficients in our numerical studies.

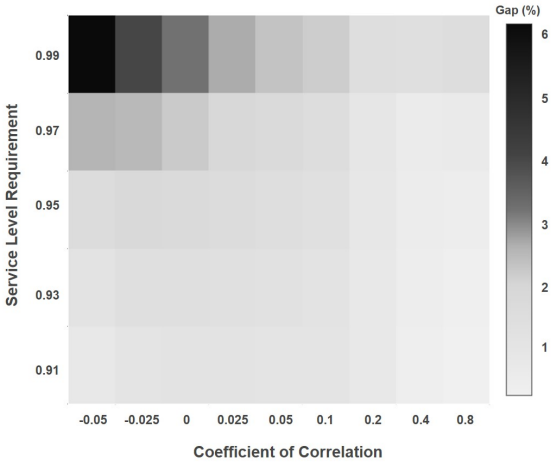
Figure 2 Comparison of optimal capacity levels in the long-chain and full-flexibility networks: (a) We vary the standard deviation from 2.0 to 3.3, and service level from 0.80 to 0.99. The degree of shade in each cell represents the size of capacity gap (%). (b) We randomly sample 50 instances of fill rate targets uniformly from each interval. (c) We vary the demand correlation coefficients from -0.05 to 0.8, and fill rate targets from 0.91 to 0.99.



(a) Effect of demand standard deviation and service level requirement



(b) Effect of service level requirement



(c) Effect of demand correlation

under different correlation coefficients are plotted in Figure 2(c). We observe a decreasing trend when the correlation coefficient increases. This is intuitive since positive correlation would reduce the benefits of flexibility due to reduced pooling effect. In the extreme case when the correlation coefficient is 1, i.e., the demand realizations are perfectly correlated among all the demand nodes, the optimal capacity required in the long-chain network and full-flexibility network would be also equivalent, as there is no room for capacity pooling.

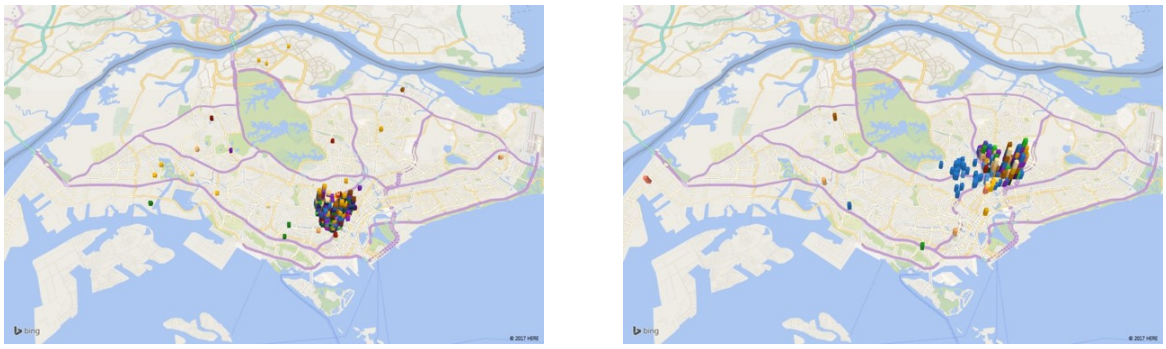
To recap, although the effectiveness of long-chain structure has been validated theoretically in many works (e.g., Jordan and Graves 1995, Chou et al. 2010), to the best of our knowledge, this is

the first work to validate the near-optimal performance of long-chain structure from the perspective of optimal capacity profile given service level guarantees. This observation provides a new direction of research to study the sparse flexible process design problem.

5.2. Case Study I: Last Mile Delivery Problems

We present an application of long-chain structure to address the challenges faced by a Singapore logistic company. At the beginning of each day, delivery personnel pick up parcels from a distribution hub and deliver these parcels to different destinations in the island. The company has adopted a standard zonal system to organize the delivery activities. The land of Singapore is grouped into 28 districts[11] and each worker is primarily responsible for deliveries into a small zone in each district. Figure 3 shows the delivery duties of two different workers over 46 days in our data set. Although they were assigned occasionally delivery tasks in other zones, their main activities are significantly clustered into a specific zone.

Figure 3 Parcel delivery activities of two delivery workers over 46 periods: Each period is identified by a unique color and the colored bars represent the delivery destinations. In each period, the worker mainly focuses on a specific zone.



(a) Worker A (b) Worker B

In this way, the number of workers assigned to work in each district is fixed each day, but the delivery volume into a specific district/zone can vary and fluctuate widely across each day. Thus manpower assigned to each zone can be under-utilized on some days, but severely under-staffed (and led to numerous failed deliveries) on other days.

We obtain a delivery dataset that contains six-week delivery information over the island (between 15th February and 31st March 2016), with around 2452 daily average deliveries. Based on the number of active delivery worker, we estimate that the median daily workload undertaken by each worker is around 44 parcels. However, we observe that 86 out of 135 delivery men actually have to

[11] Singapore districts; retrieved from `https://buyingpropertysingapore.com/map-of-singapore-districts/`

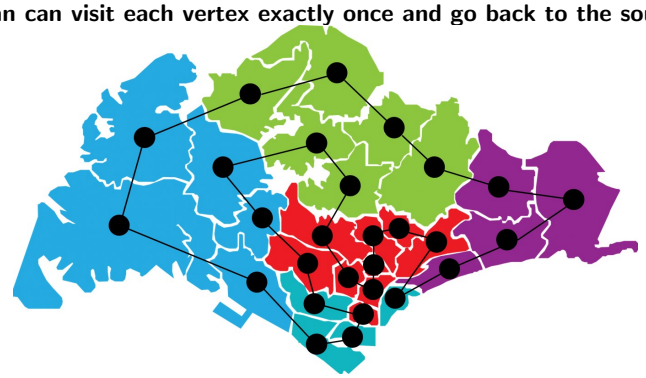
work overtime, i.e., working after 20:00 p.m., to deliver all the parcels. Despite these efforts, 6.30% of the parcels still could not be physically delivered to the customers.

The data indicated that the zonal based workload assignment system is inefficient in matching supply and demand in this last mile delivery problem. This not only increases the burden on the delivery workers, but also decreases the service quality offered to customers. The question then arises: *Can one re-design the workload assignment system, going beyond the dedicated zonal system, to obtain a new system to better match supply with demand?*

Of course the obvious approach is to pool all workers together, and assign them delivery jobs based on the actual demands received. Unfortunately this complete pooling mechanism has been ruled out by the company due to its complexity in execution. It will also lead to a volatile work environment which will add unnecessary stress on the delivery workers. Can a limited flexibility system overcome these limitations and deliver a huge improvement in performance? In what follows, we show that a long-chain type assignment mechanism is much more effective to match supply with demand.

We re-design the workload assignment system without changing the structure of 28 typical zones. For ease of exposition, we treat each zone as a graphical vertex, then we can draw a Hamiltonian Circle[12] over the Singapore map (cf. Figure 4). For the long-chain zonal assignment mechanism, each delivery worker is responsible for two neighboring zones while for the dedicated zonal mechanism, each delivery personnel is only responsible for one zone (cf. Figure 5).

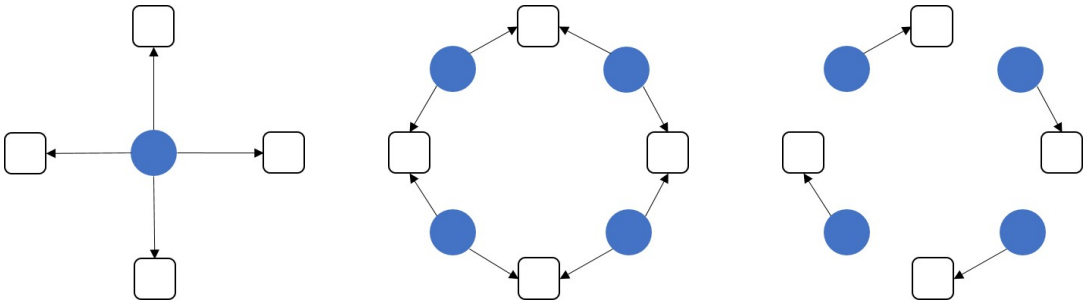
Figure 4 Hamiltonian cycle: Each black vertex represents a zone. Accorded by the geographic information, adjacent vertices are linked by a line and each vertex is exactly linked by two lines. Starting from any vertex, the delivery man can visit each vertex exactly once and go back to the source vertex.



We use the technique developed earlier to compute the workload (i.e. capacity, by number of parcels delivered) needed in the different mechanisms, by sampling on the actual delivery data

[12] Readers may refer to `http://mathworld.wolfram.com/HamiltonianCycle.html` for more information about Hamiltonian circle.

Figure 5 **Zonal assignment mechanisms: The blue circle represents the delivery personnel and the square represents the zonal vertex. A link connecting the circle and square implies that the delivery worker is responsible for the parcel delivery in the zone.**



(a) Pooling mechanism (b) Long-chain mechanism (c) Dedicated mechanism

provided. Note that the failed delivery proportion is 6.30%, and hence in the following simulation, we increase the fill rate target (β) from 94% to 99% and compare the total workload required under different zonal mechanisms.

Table 2 **Workload required under different assignment mechanisms: "Gap" refers to extra workload required compared with the optimal solution under full-flexibility mechanism.**

	Dedicated		Long-Chain		Full Flex.
β (%)	*Workload*	*Gap (%)*	*Workload*	*Gap (%)*	*Workload*
94	3192	25.53	2643	3.96	2543
95	3290	27.44	2704	4.73	2582
96	3404	29.73	2756	5.01	2624
97	3542	32.31	2837	5.97	2677
98	3726	36.03	2919	6.57	2739
99	4018	41.56	3077	8.40	2838

The full-flexibility mechanism pools all workers together and assigns them delivery tasks based on the actual demands received. However, this mechanism is hard to execute due to its intrinsic complexity in coordination. We set the full-flexibility mechanism as a benchmark to evaluate the performances of long-chain mechanism and the dedicated mechanism. We show in Table 2 that the long-chain mechanism leads to significant workload reduction. Furthermore, although the workload increases in the service level target, the marginal increment in the long-chain system is lower than that in the dedicated system. For example, increasing the fill rate from 97% to 98%, the total workload under the dedicated mechanism increases 5.19% while the one under the long-chain mechanism only increases 2.89%. These results demonstrate the advantage of long-chain mechanism in mitigating demand variation and service level fluctuation.

Last but not least, we caution that adding the two-chain flexibility brings extra traveling cost to each worker since s/he needs to deliver parcels to two adjacent zones. This could potentially

decrease the number of packages that can be delivered in a day per worker. For this strategy to be effective, the long-chain network needs to be carefully constructed and workers should be properly trained to balance the costs and benefits of introducing such flexibility into the last mile delivery systems.

6. Case Study II: Value of Subcontractors in Flexible Manufacturing

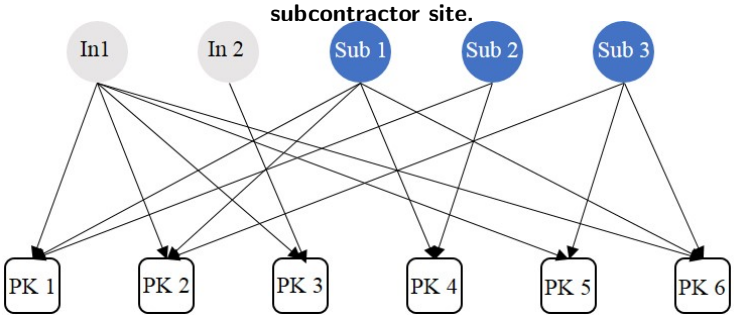
We analyze next a capacity configuration problem faced by a large semiconductor firm with a diverse group of downstream clients from various industry sectors (e.g., automotive industries). In practice, the orders from these clients change rapidly due to fierce market competition, even though a 6-month forecast (revised on a monthly basis) is given in advance to help the firm to plan its production. The firm uses an order oriented make-to-order (MTO) system and adopts a rolling horizon planning methodology to plan the capacity requirements periodically. Other than using in-house production plants, the firm also uses numerous subcontractors to provide additional capacity buffers to meet production needs.

The planning process usually starts with product group level forecast, which is further divided into multiple line items (called package classes). Based on the demand forecast, the core challenge is to acquire sufficient capacity, either through in-house sites or subcontractor plants, to serve the diverse needs of clients, which are measured by the fill rate targets for different package classes. Since the investment in capacity configuration of in-house sites is costly, the production capacity of these sites fluctuates normally in a narrow range (depending on machine conditions, and labor resources etc.). Driven by the fill rate targets, subcontractors plants provide the additional flexibility for the firm to handle last minute surge in production needs. Therefore, the firm needs to address the capacity configuration trade-off between the in-house sites and subcontractors, and determine how much flexible capacity it must procure from the subcontractors each month.

Note that each package class can only be produced at certified plants. None of the plants are certified for all package classes. The production network is therefore 'partially flexible'. Figure 6 shows the production network of the firm, for a subgroup of six package classes, denoted as PK1, PK2, ..., and PK6. A package class can only be produced at a given plant if there is a link connecting the package class node to the plant node. For example, in this production network, PK1 can be produced at In1 (in-house plant), Sub1 (subcontractor) and Sub2, while PK3 can only be produced at In1 and In2. As (monthly) demand for each package class varies over the year, and production capacity for in-house sites cannot vary too much in the short term, the variable production capacity from subcontractors are viewed as an integral part of the flexibility strategy used by the firm.

Note that the capacity profiles for In-house sites should be prepared six months in advance due to long scheduling lead time. Therefore, the main concern is whether the current in-house capacity

Figure 6 **Production network for six package classes: "In" represents in-house site, and "Sub" represents subcontractor site.**



is enough for the demand after six months. If not, the question then arises: *Based on the demand forecast and the stipulated fill rate requirement for each package class, how should the manager configure the capacities for subcontractors? How to allocate the capacity in the product network to serve and meet the service requirement of each package class?*

In the following, we formally describe how our results can be applied to address this capacity configuration problem. Our production dataset contains both forecast and actual monthly demands of each package class from July 2012 to July 2013, which allows us to measure the demand forecast accuracy measured by the ratio of actual demand over forecast. For each package class, we calculate this ratio for every month.

Interestingly, there is a consistent bias in their forecasts and the accuracy ratios fit log-normal distributions very well. Table 3 lists the estimated parameters of log-normal distribution for each package class's forecast accuracy ratio. This is then used to model the demand distribution of each package class in the model, based on the forecast provided.

Table 3 **Parameters of Log-Normal distributions for each package class's forecast accuracy ratio**

Parameters	PK1	PK2	PK3	PK4	PK5	PK6
MU	-0.1408	0.0159	-0.0264	-0.5370	-0.2730	-0.0050
SIGMA	0.5074	0.2660	0.1340	0.6328	0.0823	0.4796

We collected the forecasted demand data from the company for each package class from Aug 2013 to Jun 2014 (cf. Table 4). We also obtain the production capacity plan for each in-house and subcontractor site during the same period. Table 5 summarizes the company's in-house production capacities and the reserved subcontractor capacities in the corresponding periods.

We first determine the attainable fill rates, using the current capacity plan for in-house and subcontractor sites. We set a fill rate target of 98% for each plant, and apply the MFD policy for capacity allocation. Table 6 shows the actual service level attained for each package in the production networks.

Table 4 Forecasted demands from Aug 2013 to Jun 2014 in the company

PK	Aug-13	Sep-13	Oct-13	Nov-13	Dec-13	Jan-14	Feb-14	Mar-14	Apr-14	May-14	Jun-14
PK1	5,265	5,498	7,152	7,168	7,104	7,835	7,718	7,537	7,089	7,029	7,089
PK2	3,806	3,936	2,092	2,178	2,153	5,201	5,176	5,151	5,340	5,329	5,497
PK3	2,223	2,276	2,350	2,349	2,313	2,395	2,390	2,386	2,410	2,400	2,404
PK4	2,363	2,393	1,965	2,252	2,178	1,406	1,332	1,330	1,228	1,219	1,208
PK5	9,072	9,289	7,741	7,881	7,798	10,055	9,832	9,941	8,573	8,377	8,169
PK6	2,389	2,451	2,140	2,169	2,124	2,317	2,286	2,212	2,122	2,012	1,911

Table 5 Production capacities planned from Aug 2013 to Jun 2014 in the company

Sites	Aug-13	Sep-13	Oct-13	Nov-13	Dec-13	Jan-14	Feb-14	Mar-14	Apr-14	May-14	Jun-14
In 1	17951	21766	18082	20253	19084	20618	19116	20612	20218	20230	20174
In 2	1399	1789	1632	1703	1620	1750	1750	1750	1787	1781	1785
Sub1	531	518	200	193	163	300	300	300	650	650	650
Sub2	497	483	973	1,297	1,258	3,150	3,150	3,150	3,100	3,100	3,100
Sub3	2,415	2,347	2,005	2,673	2,598	4,890	4,890	4,890	5,035	5,035	5,035

Table 6 Achieved Service Level for Each Package from Aug 2013 to Jun 2014 in the company

PK	Aug-13	Sep-13	Oct-13	Nov-13	Dec-13	Jan-14	Feb-14	Mar-14	Apr-14	May-14	Jun-14
PK1	94.8%	97.8%	96.0%	97.3%	96.9%	97.2%	96.7%	97.5%	98.2%	98.3%	98.4%
PK2	94.2%	98.2%	93.7%	98.2%	97.0%	97.3%	96.7%	97.8%	98.5%	98.6%	98.7%
PK3	91.8%	98.2%	94.0%	98.3%	96.8%	97.8%	96.3%	98.8%	99.9%	99.9%	99.9%
PK4	90.5%	93.5%	92.2%	96.9%	96.2%	98.5%	96.2%	99.9%	99.9%	99.9%	99.9%
PK5	95.3%	97.7%	95.8%	97.5%	97.0%	97.2%	96.7%	97.5%	98.3%	98.5%	98.5%
PK6	92.8%	98.9%	94.2%	98.1%	97.0%	97.7%	96.3%	98.7%	99.9%	99.9%	99.9%

From Table 6, we observe that the company can achieve more than 90% fill rate performance for all products with the planned capacities. We also find that the realized fill rate for each package improves later into the planning horizon. For target fill rate of 98%, the company has consistently under-performed prior to Feb-14, but may have configured too much capacity after that. What is the right capacity configuration in the network to obtain a consistent fill rate performance of 98%?

To address this issue, we use the methodology developed in this paper to configure the minimum capacity such that the required service level targets (98% for all packages) can be achieved. More concretely, assuming that in-house production capacity plan does not change, we target to obtain the optimal capacities for all subcontractor sites in each month. We use the two-stage online gradient descent approach to configure the optimal profile so that the fill rates targets can be attained. The performance is evaluated by calculating the relative difference between the capacity obtained by our approach, denoted as C_a, and the current capacity installed in the system, denoted as C_b, i.e., $Performance := (C_a - C_b)/C_b \times 100\%$.

As shown in Table 7, the company can check whether the planned capacities for subcontractors sites are sufficient or not. For example, from August 2013 to Feb 2014, the capacities are insufficient to achieve 98% fill rate for each package class and hence the company needs to increase the capacity

level to meet the service level requirement. On the contrary, after Feb 2014, the planned capacities are enough for achieving 98% fill rate for each package class. Our solutions suggest reducing the total capacity level while the service level can still be attained.

Table 7 Optimal capacities at subcontractor sites and differences compared to reserved capacity plan

Month	Aug-13	Sep-13	Oct-13	Nov-13	Dec-13	Jan-14	Feb-14	Mar-14	Apr-14	May-14	Jun-14
Total Capacity	6536	4341	6191	4769	5617	9196	9749	8262	7306	6635	6566
Performance (%)	89.83	29.66	94.80	14.55	39.76	10.27	16.89	-0.93	-16.83	-24.47	-25.26

7. Conclusion

In this paper, we study the capacity allocation problem in flexible production networks and characterize the relationship between the capacity configuration and service levels delivered to the products, depending on the flexibility structure pre-configured in the network. We approach the capacity configuration problem from a new perspective of service level performance, and develop an optimal resource allocation mechanism so that the required service level targets can be attained as long as the capacity level satisfies the set of necessary and sufficient conditions developed in our paper. The robust performance of this allocation policy is evaluated and we establish a non-asymptotic bound to guarantee its performance. In addition, the constructed allocation policy is anticipative in terms of the allocation priority, and thus it can be easily implemented to solve a single-period problem or a sequence of online allocation problems. The anticipative nature of our policy also encourages truth-telling so that there is no incentive to misreport the realized demand information. Furthermore, we also obtain fill rate performance guarantee by our allocation policy even if the capacity configured into the network is insufficient.

Our capacity configuration problem involves two sets of decisions—(1) the capacity configuration decision that determines the capacity levels for each supply node in the network, and (2) the capacity allocation decision that distributes the resource to each demand node. To address the capacity configuration problem, we develop a two-stage online gradient algorithm, and apply the classical results in online convex optimization and stochastic convex optimization to demonstrate the convergence from the capacity profile obtained by our algorithm to the optimal one. We believe that this algorithm can be generalized to solve other two-stage stochastic convex problems.

As an application of our theories, we revisit a multi-item newsvendor problem with product upgrades. We show that our optimality conditions generalize the critical fractile condition for the classical newsvendor model. We also revisit the process flexibility problem and observe that the optimal capacity needed to meet the fill rate target for a long-chain network is already as small as that for a full-flexibility network. Different from previous works on process flexibility (e.g., Jordan

and Graves 1995, Chou et al. 2010), we have demonstrated the effectiveness of long-chain network in mitigating the demand uncertainty from a new perspective of optimal capacity configuration. We use data from a last mile delivery problem, and a flexible production planning problem, to demonstrate the efficacy of this approach and important insights generated for these problems. We focus on a single-period model in this paper. It will be interesting to extend this work to incorporate multi-period issues, and when capacity is "storable" for future use or when demand can be backlogged. We leave these and other issues for future research.

Acknowledgments

The authors would like to thank Prof. Terry Taylor, the associate editor, and three anonymous reviewers for their valuable comments and suggestions on improving this manuscript. This study was supported in part by National Natural Science Foundation of China (Grant Numbers 71501077 and 71520107001), Singapore Ministry of Education Social Science Research Thematic Grant (Grant Number MOE2016-SSRTG-059), and Guangdong Province Universities and Colleges Pearl River Scholar Funded Scheme (Grant Number 2017).

References

Alptekindğlu, A., A. Banerjee, A. Paul, N. Jain. 2013. Inventory pooling to deliver differentiated service. *Manufacturing & Service Operations Management*, **15(1)**, 33–44.

Asadpour, A., X. Wang, J. Zhang. 2017. Online resource allocation with limited flexibility. *Working Paper*.

Cesa-Bianchi, N., A. Conconi, C. Gentile. 2002. On the generalization ability of on-line learning algorithms. In *Advances in neural information processing systems*, 359–366.

Cesa-Bianchi, N., G. Lugosi, 2006. Prediction, learning, and games. *Cambridge University Press*.

Chen, J., D. K. Lin, D. J. Thomas. 2003. On the item fill rate for a finite horizon. *Operations Research Letters*, **31**, 119–199.

Chen, X., Zhang, J., Zhou, Y. Thomas. 2015. Optimal sparse designs for process flexibility via probabilistic expanders. *Operations Research*, **63(5)**, 1159–1176.

Chen, X., Ma, T., Zhang, J., Zhou, Y. 2018. Optimal Design of Process Flexibility for General Production Systems. *Working Paper*.

Chou, M. C., G. A. Chua, C. P. Teo, H. Zheng. 2010. Design for process flexibility: Efficiency of the long chain and sparse structure. *Operations research*, **58(1)**, 43–58.

Chou, M. C., G. A. Chua, C. P. Teo, H. Zheng. 2011. Process flexibility revisited: The graph expander and its applications. *Operations research*, **59(5)**, 1090–1105.

Chou, M. C., G. A. Chua, H. Zheng. 2014. On the performance of sparse process structures in partial postponement production systems. *Operations research*, **62(2)**, 348–365.

Désir, A., Goyal, V., Wei, Y., Zhang, J. 2016. Sparse process flexibility designs: is the long chain really optimal? *Operations Research*, **64(2)**, 416–431.

Edmonds, J. 2003. Submodular Functions, Matroids, and Certain Polyhedra. *Combinatorial Optimization – Eureka, You Shrink!: Papers Dedicated to Jack Edmonds 5th International Workshop Aussois, France, March 5–9*, 11–26.

Goyal, M., S. Netessine. 2011. Volume flexibility, product flexibility, or both: The role of demand correlation and product substitution. *Manufacturing & service operations management*, **13(2)**, 180–193.

Hallgren, M., Olhager, J. 2009. Flexibility configurations: Empirical analysis of volume and product mix flexibility. *Omega*, **37(4)**, 746–756.

Hopp, W. J., M. L. Spearman. 2008. Factory Physics: Foundations of Manufacturing Management. McGraw-Hill.

Jordan, W. C., S. C. Graves. 1995. Principles on the benefits of manufacturing process flexibility *Management Science,*, **41(4)**, 577–594.

Kakade, S. M., A. Tewari. 2009. On the generalization ability of online strongly convex programming algorithms. In *Advances in neural information processing systems*, 801–808.

Lyu, G., M. C. Chou, C. P. Teo, Z. Zheng, Y. Zhong. 2017. Capacity allocation with stock-out probability targets: Theory and applications. *Working Paper.*

Mahdavi, M., R. Jin, T. Yang. 2012. Trading regret for efficiency: online convex optimization with long term constraints. *Journal of Machine Learning Research*, **13(Sep)**, 2503–2528.

Mahdavi, M., T. Yang, R. Jin. 2013. Stochastic convex optimization with multiple objectives. In *Advances in Neural Information Processing Systems*, 1115–1123.

Megiddo, N. 1974. Optimal flows in networks with multiple sources and sinks. *Mathematical Programming*, **7(1)**, 97–107.

Megiddo, N. 1977. A good algorithm for lexicographically optimal flows in multi-terminal networks. *American Mathematical Society*, **83(3)**.

Nocedal, J., S. J. Wright. 2006. Numerical Optimization. *Springer New York.*

Porteus, E. L. 1990. Stochastic inventory theory. D. P. Heyman and M. J. Sobel (Eds.). *Handbooks in OR & MS, Vol.2.* Elsevier, North-Holland, 605-652.

Shalev-Shwartz, S. 2011. Online learning and online convex optimization. *Foundations and Trends in Machine Learning*, **4(2)**, 107–194.

Shalev-Shwartz, S., O. Shamir, N. Srebro, K. Sridharan. 2009. Stochastic Convex Optimization. In *COLT.*

Schrijver, A. 2003. Combinatorial Optimization : Polyhedra and Efficiency (Algorithms and Combinatorics). *Springer*, Chapter 44.

Shapiro, A., Dentcheva, D., Ruszczyski, A. 2009. Lectures on stochastic programming: modeling and theory. *Society for Industrial and Applied Mathematics.*

Shi, C., Y. Wei, Y. Zhong. 2015. Process flexibility for multi-period production systems. *Working Paper.*

Simchi-Levi, D., Y. Wei. 2012. Understanding the performance of the long chain and sparse designs in process flexibility. *Operations Research*, **60(5)**, 1125–1141.

Simchi-Levi, D., Y. Wei. 2015. Worst-case analysis of process flexibility designs. *Operations Research*, **63(1)**, 166–185.

Suarez, F. F., M. A. Cusumano, C. H. Fine. 1996. An empirical study of manufacturing flexibility in printed circuit board assembly. *Operations research*, **44(1)**, 223–240.

Wang, X., J. Zhang. 2015. Process flexibility: A distribution-free bound on the performance of k-chain. *Operations Research*, **63(3)**, 555–571.

Zinkevich, M. 2003. Online convex programming and generalized infinitesimal gradient ascent. In *Proceedings of the 20th International Conference on Machine Learning (ICML).*

Zhong, Y., Z. Zheng, Z., M. C. Chou, C. P. Teo. 2018. Resource pooling and allocation policies to deliver differentiated service. *Management Science*, **64(4)**, 1555–1573.

Appendix. Supplemental Materials for Capacity Allocation in Flexible Production Networks: Theory and Applications

A. Allocation Policy under Arbitrary Capacity Profile

In section 3, we have demonstrated that the debt under the MFD policy converges to zero when the array of capacity levels $\boldsymbol{S}$ is sufficient, i.e., satisfying conditions (4). Surprisingly, we show that the same policy can also be implemented to achieve a certain performance guarantee even if such a sufficiency assumption is violated. In this section, we investigate the following question: Given an array of fixed but arbitrary capacity levels $\boldsymbol{S}$, which does not necessarily satisfy conditions (4), what is $\lim_{T\to\infty}\boldsymbol{\rho}^+(T)$, the positive part of the average debt under the MFD policy, as T becomes large?

To answer the question, we first consider the optimization Problem (P-debt) provided below, which serves as a benchmark for the debt under the MFD policy in a certain sense that we make precise in subsequent discussion.

$$
\begin{aligned}
\text{(P-debt)} \quad \min_{\boldsymbol{y}\in\mathbb{R}_+^{|\mathcal{I}|},\, \boldsymbol{\rho}\in\mathbb{R}^{|\mathcal{I}|}} \quad & \|\boldsymbol{\rho}^+\|_2^2 \\
\text{s.t.} \quad & \rho_i = \beta_i \mathbf{E}[X_i] - y_i,\ \forall i\in\mathcal{I} && (17)\\
& \sum_{i\in U} y_i \le \mathbf{E}\left[\min_{V\subseteq U}\left\{\sum_{i\in V} X_i + \sum_{j\in\Gamma(U\setminus V)} S_j\right\}\right],\ \forall U\subseteq\mathcal{I} && (18)\\
& \sum_{i\in\mathcal{I}} y_i = \mathbf{E}\left[\min_{V\subseteq\mathcal{I}}\left\{\sum_{i\in V} X_i + \sum_{j\in\Gamma(\mathcal{I}\setminus V)} S_j\right\}\right]. && (19)
\end{aligned}
$$

Denote $(\boldsymbol{y}^*, \boldsymbol{\rho}^*)$ as an optimal solution to Problem (P-debt). Variable y_i is the expected amount of capacity allocated to demand node i through a max flow, by virtue of constraints (18, 19). Consequently, variable ρ_i defined in (17) represents the corresponding debt associated with demand node i, for each $i\in\mathcal{I}$.

The objective of (P-debt) is to minimize the squared 2-norm of $\boldsymbol{\rho}^+$, the positive part of debt $\boldsymbol{\rho}$. When $\boldsymbol{S}$ is sufficient for the required service requirement, i.e., $\boldsymbol{S}$ satisfies conditions (4), then we can assign capacity such that $y_i^* \ge \beta_i\mathbf{E}[X_i]$, and $\rho_i^* \le 0$ for all $i\in\mathcal{I}$. Hence, the optimal value of (P-debt) is equal to 0. Otherwise, if $\boldsymbol{S}$ is not sufficient, then the optimal solution $\boldsymbol{y}^*$ minimizes the squared 2-norm of the positive part of debt $\boldsymbol{\rho}^*$, and the optimal value of (P-debt) is the corresponding squared 2-norm. In this case, the optimal value of (P-debt) is strictly larger than 0, that is, positive debts are incurred for some product nodes.

In the current case of insufficient capacity, the allocated capacity $\boldsymbol{y}^*$ and its associated debt $\boldsymbol{\rho}^*$ serve as our benchmark. We prove Theorem 3, which shows that the average debt $\boldsymbol{\rho}^+(T)$'s positive part under the MFD policy converges to the benchmark debt's positive part $(\boldsymbol{\rho}^*)^+$ as T becomes sufficiently large. Theorem 3, which generalizes Theorem 2, is provided below.

THEOREM 3. *Suppose we are given an arbitrary but fixed array of capacities $\boldsymbol{S}$. The average debt $\boldsymbol{\rho}(T+1)$ under the MFD policy satisfies the following non-asymptotic convergence guarantees in expectation:*

$$
\mathbf{E}\left[\|\boldsymbol{\rho}^+(T+1)\|_2^2\right] - \|(\boldsymbol{\rho}^*)^+\|_2^2 \le \frac{\Upsilon\Lambda(1+\log T)}{T}. \tag{20}
$$

The policy also satisfies the following non-asymptotic convergence with high probability: For any fixed $\delta \in (0,1)$*, we have*

$$\mathbf{P}\left(\|\boldsymbol{\rho}^+(T+1)\|_2^2 - \|(\boldsymbol{\rho}^*)^+\|_2^2 \le \frac{1}{\sqrt{T}}\left\{\frac{\Upsilon\Lambda(1+\log T)}{\sqrt{T}} + 4\Upsilon'\Lambda'\sqrt{2\log\left(\frac{1}{\delta}\right)}\right\}\right) \ge 1-\delta, \tag{21}$$

where constants $\Upsilon, \Lambda, \Upsilon', and\ \Lambda'$ *are independent of* T*.*

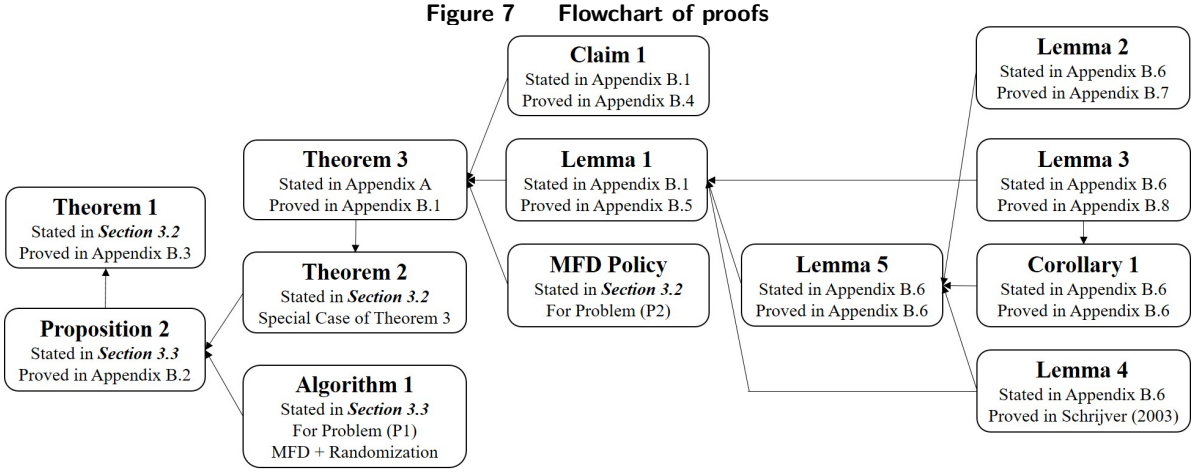
We remark that Theorem 2 is the special case of Theorem 3 when $\boldsymbol{\rho}^* = \mathbf{0}$. Theorem 3 is proved by tracking the decay of $\|\boldsymbol{\rho}^+(t)\|_2^2$ as t increases. In the analysis, we use the property that $\boldsymbol{y}^*$ lies in a *base polymatroid* associated with the max flow in $\mathcal{G}$, as detailed in Appendix B.5. Intuitively, Theorem 3 implies that the Euclidean distance between the attained service level by performing the MFD policy and the required service target is minimal. Therefore, even if the capacity profile is not sufficient to deliver the required service level, the attained service level still converges to the optimal one, when T becomes large.

B. Technical Proofs

Note that Part (i) of Theorem 1 has already been proved in the discussion of Section 3.2 before presenting the theorem. In this appendix, we focus on discussing Part (ii) of the theorem, which is supported by Proposition 2 and further proved by Theorem 2 and the randomized allocation policy in Algorithm 1. Moreover, as Theorem 2 is a special case of Theorem 3 (discussed in Appendix A), we prove Theorem 3 instead. The proof of Theorem 3 involves a range of different results, which are proved in different parts of this appendix. We provide a diagram that demonstrates the connections between different results in Figure 7.

In addition, we prove the stand-alone result of Proposition 1 in Appendix B.9. Note that Proposition 1 is not required in establishing the main theorems in the paper, but it helps to demonstrate the efficiency of solving the weighted max flow problem in the MFD policy.

Figure 7 Flowchart of proofs



B.1. Proof of Theorem 3

THEOREM 3. *Suppose we are given an arbitrary but fixed array of capacities $\boldsymbol{S}$. The average debt $\boldsymbol{\rho}(T+1)$ under the MFD policy satisfies the following non-asymptotic convergence guarantees in expectation:*

$$\mathbf{E}\left[\|\boldsymbol{\rho}^+(T+1)\|_2^2\right] - \|(\boldsymbol{\rho}^*)^+\|_2^2 \le \frac{\Upsilon\Lambda(1+\log T)}{T}. \tag{20}$$

The policy also satisfies the following non-asymptotic convergence with high probability: For any fixed $\delta \in (0,1)$, we have

$$\mathbf{P}\left(\|\boldsymbol{\rho}^+(T+1)\|_2^2 - \|(\boldsymbol{\rho}^*)^+\|_2^2 \le \frac{1}{\sqrt{T}}\left\{\frac{\Upsilon\Lambda(1+\log T)}{\sqrt{T}} + 4\Upsilon'\Lambda'\sqrt{2\log\left(\frac{1}{\delta}\right)}\right\}\right) \ge 1-\delta, \tag{21}$$

where constants $\Upsilon, \Lambda, \Upsilon'$, and Λ' are independent of T.

The proof of Theorem 3 involves Claim 1 and Lemma 1, provided below:

CLAIM 1. *The function $g(\boldsymbol{\rho}) := \|\boldsymbol{\rho}^+\|_2^2$ is 1-smooth with respect to the Euclidean norm. Equivalently, for any $\boldsymbol{\rho}, \boldsymbol{\psi} \in \mathbb{R}^{|\mathcal{I}|}$, the following inequality holds:*

$$\|\boldsymbol{\psi}^+\|_2^2 + 2\left(\boldsymbol{\psi}^+\right)^\top[\boldsymbol{\rho}-\boldsymbol{\psi}] \le \|\boldsymbol{\rho}^+\|_2^2 \le \|\boldsymbol{\psi}^+\|_2^2 + 2\left(\boldsymbol{\psi}^+\right)^\top[\boldsymbol{\rho}-\boldsymbol{\psi}] + \|\boldsymbol{\rho}-\boldsymbol{\psi}\|_2^2. \tag{22}$$

LEMMA 1. *Consider the MFD policy. For each $i \in \mathcal{I}$, let*

$$\tilde{y}_i(t) := \mathbf{E}\left[\sum_{j\in\Gamma(\{i\})} D_{j,i}(t)\Big|\boldsymbol{\rho}(t)\right] \tag{23}$$

be the expected amount of resources allocated to product node i under the policy for the t^{th} sample of demand scenario, conditional on the average debt $\boldsymbol{\rho}(t)$ across the first $(t-1)$ samples. The following inequality holds for any realization of $\boldsymbol{\rho}(t)$:

$$\sum_{i\in\mathcal{I}} \rho_i^+(t)\tilde{y}_i(t) \ge \sum_{i\in\mathcal{I}} \rho_i^+(t) y_i^*, \tag{24}$$

where $\boldsymbol{y}^*$ (together with a suitably chosen $\boldsymbol{\rho}^*$) constitutes an optimal solution to (P-debt).

Claim 1 and Lemma 1 are proved in Appendix B.4 and B.5, respectively. We use Claim 1 to quantify the convergence of $\{\boldsymbol{\rho}^+(t+1)\}_{t=0}^\infty$, and we use Lemma 1 to argue that $\boldsymbol{\rho}^+(t+1)$ indeed converges to $(\boldsymbol{\rho}^*)^+$, the positive part of the optimal debt $\boldsymbol{\rho}^*$ in (P-debt).

PROOF OF THEOREM 3. Recall from (5) that $\boldsymbol{\rho}(t+1)$ is the average debt for the first t demand samples, and $\boldsymbol{R}(s)$ is the debt for the s^{th} sample. For any $0 \le t \le T-1$, we have

$$\begin{aligned}
&\|\boldsymbol{\rho}^+(t+1)\|_2^2 \\
\le&\|\boldsymbol{\rho}^+(t)\|_2^2 + 2\boldsymbol{\rho}^+(t)^\top[\boldsymbol{\rho}(t+1)-\boldsymbol{\rho}(t)] + \|\boldsymbol{\rho}(t+1)-\boldsymbol{\rho}(t)\|_2^2 && (25)\\
=&\|\boldsymbol{\rho}^+(t)\|_2^2 + \frac{2\boldsymbol{\rho}^+(t)^\top[\boldsymbol{R}(t)-\boldsymbol{\rho}(t)]}{t} + \frac{\|\boldsymbol{R}(t)-\boldsymbol{\rho}(t)\|_2^2}{t^2} && (26)\\
\le&\|\boldsymbol{\rho}^+(t)\|_2^2 + \frac{2\boldsymbol{\rho}^+(t)^\top[\boldsymbol{R}(t)-\boldsymbol{\rho}(t)]}{t} + \frac{\Upsilon\Lambda}{t^2} && (27)\\
=&\|\boldsymbol{\rho}^+(t)\|_2^2 + \frac{2\boldsymbol{\rho}^+(t)^\top\{\mathbf{E}[\boldsymbol{R}(t)\mid\boldsymbol{\rho}(t)]-\boldsymbol{\rho}(t)\}}{t} + \frac{\Upsilon\Lambda}{t^2} + \frac{2\boldsymbol{\rho}^+(t)^\top\{\boldsymbol{R}(t)-\mathbf{E}[\boldsymbol{R}(t)\mid\boldsymbol{\rho}(t)]\}}{t}. && (28)
\end{aligned}$$

Step (25) is justified by applying the upper bound in Claim 1, with $\boldsymbol{\rho} \to \boldsymbol{\rho}(t+1)$, $\boldsymbol{\psi} \to \boldsymbol{\rho}(t)$. Step (26) is by the definitions of $\boldsymbol{\rho}(t), \boldsymbol{R}(t)$:

$$\boldsymbol{\rho}(t+1) = \frac{t-1}{t}\boldsymbol{\rho}(t) + \frac{1}{t}\boldsymbol{R}(t) \quad \Leftrightarrow \quad \boldsymbol{\rho}(t+1) - \boldsymbol{\rho}(t) = \frac{\boldsymbol{R}(t) - \boldsymbol{\rho}(t)}{t}.$$

In step (27), we bound the term $\|\boldsymbol{R}(t) - \boldsymbol{\rho}(t)\|/t^2$ with certainty from above:

$$\|\boldsymbol{R}(t) - \boldsymbol{\rho}(t)\|_2^2 \le \sum_{i\in\mathcal{I}} \left[\sum_{j\in\Gamma(\{i\})} D_{j,i}(t)\right]^2 \le \Upsilon \sum_{i\in\mathcal{I}} \sum_{j\in\Gamma(\{i\})} D_{j,i}(t) = \Upsilon \sum_{j\in\mathcal{J}} \sum_{i\in\Gamma(\{j\})} D_{j,i}(t) \le \Upsilon\Lambda,$$

by the definitions of Υ and Λ in (6). Finally, in step (28), we decompose the difference term $\boldsymbol{R}(t) - \boldsymbol{\rho}(t)$ by considering the conditional expectation $\mathbf{E}[\boldsymbol{R}(t) \mid \boldsymbol{\rho}(t)]$.

Lemma 1 implies that

$$\boldsymbol{\rho}^+(t)^\top \mathbf{E}[\boldsymbol{R}(t) \mid \boldsymbol{\rho}(t)] \le \boldsymbol{\rho}^+(t)^\top \boldsymbol{\rho}^*.$$

Thus, we can proceed with bounding (28) in the following:

$$\begin{aligned}
\|\boldsymbol{\rho}^+(t+1)\|_2^2 &\le \|\boldsymbol{\rho}^+(t)\|_2^2 + \frac{2\boldsymbol{\rho}^+(t)^\top[\boldsymbol{\rho}^* - \boldsymbol{\rho}(t)]}{t} + \frac{\Upsilon\Lambda}{t^2} + \frac{2\boldsymbol{\rho}^+(t)^\top\{\boldsymbol{R}(t) - \mathbf{E}[\boldsymbol{R}(t) \mid \boldsymbol{\rho}(t)]\}}{t} \\
&\le \|\boldsymbol{\rho}^+(t)\|_2^2 + \frac{\|(\boldsymbol{\rho}^*)^+\|_2^2 - \|\boldsymbol{\rho}^+(t)\|_2^2}{t} + \frac{\Upsilon\Lambda}{t^2} + \frac{2\boldsymbol{\rho}^+(t)^\top\{\boldsymbol{R}(t) - \mathbf{E}[\boldsymbol{R}(t) \mid \boldsymbol{\rho}(t)]\}}{t}, \qquad (29)
\end{aligned}$$

where step (29) is by applying the lower bound in Claim 1 with $\boldsymbol{\psi} \leftarrow \boldsymbol{\rho}(t)$ and $\boldsymbol{\rho} \leftarrow \boldsymbol{\rho}^*$. Rearranging (29) gives

$$\begin{aligned}
\|\boldsymbol{\rho}^+(t+1)\|_2^2 - \|(\boldsymbol{\rho}^*)^+\|_2^2 \le \left(1 - \frac{1}{t}\right)\left[\|\boldsymbol{\rho}^+(t)\|_2^2 - \|(\boldsymbol{\rho}^*)^+\|_2^2\right] + \frac{\Upsilon\Lambda}{t^2} \\
+ \frac{2\boldsymbol{\rho}^+(t)^\top\{\boldsymbol{R}(t) - \mathbf{E}[\boldsymbol{R}(t) \mid \boldsymbol{\rho}(t)]\}}{t}. \qquad (30)
\end{aligned}$$

Now, we assert that for any $0 \le t \le T$,

$$\begin{aligned}
\|\boldsymbol{\rho}^+(T+1)\|_2^2 - \|(\boldsymbol{\rho}^*)^+\|_2^2 \le \frac{t}{T}\left[\|\boldsymbol{\rho}^+(t+1)\|_2^2 - \|(\boldsymbol{\rho}^*)^+\|_2^2\right] + \frac{\Upsilon\Lambda}{T}\sum_{s=t+1}^{T}\frac{1}{s} \\
+ \sum_{s=t+1}^{T}\frac{2\boldsymbol{\rho}^+(s)^\top\{\boldsymbol{R}(s) - \mathbf{E}[\boldsymbol{R}(s) \mid \boldsymbol{\rho}(s)]\}}{T}, \qquad (31)
\end{aligned}$$

by a backward induction on t. Now, (31) clearly holds when $t = T$. Inductively, assume that (31) holds for $t = \tau$, where $\tau > 0$. Then we have

$$\begin{aligned}
&\|\boldsymbol{\rho}^+(T+1)\|_2^2 - \|(\boldsymbol{\rho}^*)^+\|_2^2 \\
\le&\frac{\tau}{T}\left[\|\boldsymbol{\rho}^+(\tau+1)\|_2^2 - \|(\boldsymbol{\rho}^*)^+\|_2^2\right] + \frac{\Upsilon\Lambda}{T}\sum_{s=\tau+1}^{T}\frac{1}{s} + \sum_{s=\tau+1}^{T}\frac{2\boldsymbol{\rho}^+(s)^\top\{\boldsymbol{R}(s) - \mathbf{E}[\boldsymbol{R}(s) \mid \boldsymbol{\rho}(s)]\}}{T} \\
\le&\frac{\tau}{T}\left\{\left(1 - \frac{1}{\tau}\right)\left[\|\boldsymbol{\rho}^+(\tau)\|_2^2 - \|(\boldsymbol{\rho}^*)^+\|_2^2\right] + \frac{\Upsilon\Lambda}{\tau^2} + \frac{2\boldsymbol{\rho}^+(\tau)^\top\{\boldsymbol{R}(\tau) - \mathbf{E}[\boldsymbol{R}(\tau) \mid \boldsymbol{\rho}(\tau)]\}}{\tau}\right\} \\
&+ \frac{\Upsilon\Lambda}{T}\sum_{s=\tau+1}^{T}\frac{1}{s} + \sum_{s=\tau+1}^{T}\frac{2\boldsymbol{\rho}^+(s)^\top\{\boldsymbol{R}(s) - \mathbf{E}[\boldsymbol{R}(s) \mid \boldsymbol{\rho}(s)]\}}{T} \\
=&\frac{\tau-1}{T}\left[\|\boldsymbol{\rho}^+(\tau)\|_2^2 - \|(\boldsymbol{\rho}^*)^+\|_2^2\right] + \frac{\Upsilon\Lambda}{T}\sum_{s=\tau}^{T}\frac{1}{s} + \sum_{s=\tau}^{T}\frac{2\boldsymbol{\rho}^+(s)^\top\{\boldsymbol{R}(s) - \mathbf{E}[\boldsymbol{R}(s) \mid \boldsymbol{\rho}(s)]\}}{T},
\end{aligned}$$

hence showing that the induction claim is true for $t=\tau-1$, and demonstrating (31) for all t. In particular, letting $t=0$ in (31) yields

$$\|\boldsymbol{\rho}^+(T+1)\|_2^2 - \|\left(\boldsymbol{\rho}^*\right)^+\|_2^2 \le \frac{\Upsilon\Lambda(1+\log T)}{T} + \frac{2}{T}\sum_{s=1}^{T}\left\{\boldsymbol{\rho}^+(s)^\top \boldsymbol{R}(s) - \mathbf{E}\left[\boldsymbol{\rho}^+(s)^\top \boldsymbol{R}(s) \mid \boldsymbol{\rho}(s)\right]\right\}. \tag{32}$$

Taking expectation on both sides of (32) immediately yields the required convergence in expectation in (20). Indeed, the second term on the right hand side of (32) is zero, by collapsing the conditional expectations.

To establish the convergence with high probability, consider the random variable

$$Z(t) := \sum_{s=1}^{t} \boldsymbol{\rho}^+(s)^\top \boldsymbol{R}(s) - \mathbf{E}\left[\boldsymbol{\rho}^+(s)^\top \boldsymbol{R}(s) \mid \boldsymbol{\rho}(s)\right],$$

and the $\sigma-$algebra

$$\mathcal{F}(t) := \sigma\left(\{\boldsymbol{R}(s)\}_{s=1}^{t}\right) = \sigma\left(\{\boldsymbol{\rho}(s)\}_{s=1}^{t+1}, \{\boldsymbol{R}(s)\}_{s=1}^{t}\right).$$

Evidently, the stochastic process $\{Z(t)\}_{t=1}^{T}$ is a martingale with respect to the filtration $\{\mathcal{F}(t)\}_{t=1}^{T}$. Indeed, for each $t\ge 1$, the random variable $Z(t)$ is $\mathcal{F}(t)$-measurable, and we also have $\mathbf{E}[Z(t)|\mathcal{F}(t-1)] = Z(t-1)$. In addition, the absolute value $|Z(t+1)-Z(t)|$ of the martingale difference is at most $2\Upsilon'\cdot\Lambda'$ with certainty for each t (Recall the definition of constants Υ', Λ' in equation (6)). To see this, first note that

$$\left|Z(t+1)-Z(t)\right| = \left|\boldsymbol{\rho}^+(t+1)^\top \boldsymbol{R}(t+1) - \mathbf{E}\left[\boldsymbol{\rho}^+(t+1)^\top \boldsymbol{R}(t+1) \mid \boldsymbol{\rho}(t+1)\right]\right|.$$

Now, to bound the difference, we have

$$\max_{i\in\mathcal{I}} |\rho_i^+(t)| \le \max_{i\in\mathcal{I}, s\in\{1,2,\ldots,t-1\}} \left\{\left|\beta_i \mathbf{E}[X_i] - \sum_{j\in\Gamma(\{i\})} D_{j,i}(s)\right|\right\} \le \max_{i\in\mathcal{I}}\left\{\max\left\{\beta_i\mathbf{E}[X_i], \sum_{j\in\Gamma(\{i\})} S_j\right\}\right\} = \Upsilon',$$

$$\left|\sum_{i\in\mathcal{I}} R_i(t)\right| = \left|\sum_{i\in\mathcal{I}}\beta_i\mathbf{E}[X_i] - \sum_{i\in\mathcal{I}}\sum_{j\in\Gamma(\{i\})} D_{j,i}(s)\right| \le \max\left\{\sum_{i\in\mathcal{I}}\beta_i\mathbf{E}[X_i], \sum_{j\in\mathcal{J}} S_j\right\} = \Lambda'.$$

Altogether, we see that $|\boldsymbol{\rho}^+(t)^\top \boldsymbol{R}(t)| \le \Upsilon'\Lambda'$, and hence we have the asserted bound on the martingale difference. Furthermore, by Hoeffding Inequality on the martingale $\{Z(t)\}_{t=1}^{T}$, we have

$$\begin{aligned}&\mathbf{P}\left(\sum_{s=1}^{T}\left\{\boldsymbol{\rho}^+(s)^\top \boldsymbol{R}(s) - \mathbf{E}\left[\boldsymbol{\rho}^+(s)^\top \boldsymbol{R}(s) \mid \boldsymbol{\rho}(s)\right]\right\} > z\right)\\ =&\mathbf{P}\left(Z(T)-Z(0) > z\right) \le \exp\left[-\frac{z^2}{8T(\Upsilon'\cdot\Lambda')^2}\right]\end{aligned} \tag{33}$$

for every $z>0$.

For any fixed $\delta\in(0,1)$, let's put $z=\sqrt{8T\log(1/\delta)}\Upsilon'\cdot\Lambda'$ in (33), which yields

$$\begin{aligned}&\mathbf{P}\left(\sum_{s=1}^{T}\left\{\boldsymbol{\rho}^+(s)^\top \boldsymbol{R}(s) - \mathbf{E}\left[\boldsymbol{\rho}^+(s)^\top \boldsymbol{R}(s) \mid \boldsymbol{\rho}(s)\right]\right\} > \sqrt{8T\log(1/\delta)}\Upsilon'\cdot\Lambda'\right)\\ \le&\exp\left[-\frac{(\sqrt{8T\log(1/\delta)}\Upsilon'\cdot\Lambda')^2}{8T(\Upsilon'\cdot\Lambda')^2}\right] = \exp\left[-\log\frac{1}{\delta}\right] = \delta.\end{aligned} \tag{34}$$

Then, we incorporate the high probability bound (34) in inequality (32), which holds with certainty. Altogether, we conclude that, with probability at least $1-\delta$, we have

$$\begin{aligned}\|\boldsymbol{\rho}^+(T+1)\|_2^2 - \|\,(\boldsymbol{\rho}^*)^+\,\|_2^2 &\le \frac{\Upsilon\Lambda(1+\log T)}{T} + \frac{2}{T}\sum_{s=1}^{T}\left\{\boldsymbol{\rho}^+(s)^\top \boldsymbol{R}(s) - \mathbf{E}\left[\boldsymbol{\rho}^+(s)^\top \boldsymbol{R}(s)\mid \boldsymbol{\rho}(s)\right]\right\}\\ &\le \frac{\Upsilon\Lambda(1+\log T)}{T} + \frac{2}{T}\sqrt{8T\log\frac{1}{\delta}}\Upsilon'\cdot\Lambda' = \frac{1}{\sqrt{T}}\left\{\frac{\Upsilon\Lambda(1+\log T)}{\sqrt{T}} + 4\Upsilon'\Lambda'\sqrt{2\log\left(\frac{1}{\delta}\right)}\right\},\end{aligned} \tag{35}$$

which yields the convergence with high probability stated in (21). ■

B.2. Proof of Proposition 2

PROPOSITION 2. *If conditions* (4) *are satisfied by the capacity* $\boldsymbol{S}$*, Algorithm 1 provides a feasible capacity allocation solution to the single-period Problem (P1).*

PROOF. We first apply the non-asymptotic convergence guarantees on the average debt from Theorem 3 to derive the optimality of the MFD policy for Problem (P2), and then we combine the anticipative property of the priority list in the MFD policy and the randomization mechanism detailed in Algorithm 1 to prove the optimality of Algorithm 1 for Problem (P1).

Note that for any $\epsilon>0$, we have

$$\sum_{t=0}^{\infty}\mathbf{P}\left(\|\boldsymbol{\rho}^+(t+1)\|_2^2>\epsilon\right) \le C\sum_{t=0}^{\infty}\exp\left[-\frac{t\epsilon^2}{\Upsilon\Lambda}\right] < \infty. \tag{36}$$

The first inequality in (36) is by (21) in Theorem 3, and C is an absolute constant. Now, (36) implies that $\lim_{T\to\infty}\|\boldsymbol{\rho}^+(T+1)\|_2^2=0$ almost surely. In particular, for every $i\in\mathcal{I}$ we have

$$\lim_{T\to\infty}\rho_i^+(T+1)=0 \quad \text{a.s.} \tag{37}$$

$$\Rightarrow \limsup_{T\to\infty}\rho_i(T+1)\le 0 \quad \text{a.s.} \tag{38}$$

$$\Rightarrow \liminf_{T\to\infty}\frac{1}{T}\sum_{t=1}^{T}\sum_{j\in\Gamma(\{i\})}D_{j,i}(t)\ge\beta_i\mathbf{E}[X_i] \quad \text{a.s.} \tag{39}$$

$$\Rightarrow \liminf_{T\to\infty}\frac{\sum_{t=1}^{T}\sum_{j\in\Gamma(\{i\})}D_{j,i}(t)}{\sum_{t=1}^{T}X_i(t)}\ge\beta_i \quad \text{a.s.} \tag{40}$$

The last implication is by the Strong Law of Large Numbers. Therefore, the MFD policy is able to deliver the fill rate requirements for Problem (P2).

Following the randomized allocation policy in Algorithm 1, we have

$$\mathbf{E}\left[\sum_{j\in\Gamma(\{i\})}D_{j,i}^{\mathcal{L}}(\boldsymbol{X},\boldsymbol{S})\right] = \mathbf{E}\left[\frac{\sum_{t=1}^{T}\sum_{j\in\Gamma(\{i\})}D_{j,i}^{\mathcal{L}(t)}(\boldsymbol{X},\boldsymbol{S})}{T}\right],$$

where the superscripts $\mathcal{L}$ are used to highlight the priority lists that are carried out for resources allocation, and $\mathcal{L}(t)$ denotes the allocation priority list for sample t from implementing the MFD policy. Since $\boldsymbol{X}(t)$ follows the same distribution as $\boldsymbol{X}$, and $\boldsymbol{X}(t)$ is independent of the priority list $\mathcal{L}(t)$, we have

$$\mathbf{E}\left[\sum_{j\in\Gamma(\{i\})}D_{j,i}^{\mathcal{L}(t)}(\boldsymbol{X},\boldsymbol{S})\right] = \mathbf{E}\left[\sum_{j\in\Gamma(\{i\})}D_{j,i}^{\mathcal{L}(t)}(\boldsymbol{X}(t),\boldsymbol{S})\right],\ \forall t=1,2,\ldots,T.$$

Therefore,

$$\begin{aligned}\mathbf{E}\left[\sum_{j\in\Gamma(\{i\})} D^{\mathcal{L}}_{j,i}(\boldsymbol{X},\boldsymbol{S})\right] &= \mathbf{E}\left[\frac{\sum_{t=1}^{T}\sum_{j\in\Gamma(\{i\})} D^{\mathcal{L}(t)}_{j,i}(\boldsymbol{X},\boldsymbol{S})}{T}\right]\\ &= \mathbf{E}\left[\frac{\sum_{t=1}^{T}\sum_{j\in\Gamma(\{i\})} D^{\mathcal{L}(t)}_{j,i}(\boldsymbol{X}(t),\boldsymbol{S})}{T}\right]\\ &= \mathbf{E}\left[\frac{\sum_{t=1}^{T}\sum_{j\in\Gamma(\{i\})} D^{\mathcal{L}(t)}_{j,i}(t)}{T}\right]\\ &\geq \beta_i\mathbf{E}[X_i],\ \forall i\in\mathcal{I},\ \text{as } T\to\infty.\end{aligned}$$

The second last equality is a result of changing notation, and the last inequality follows from the above results that the MFD policy is able to deliver the target fill rate for Problem (P2), which shares the equivalent definition of fill rate requirement. To summarize, we conclude that the service level requirement in Problem (P1) can be attained by the randomized allocation policy, with a sufficiently large T. ■

B.3. Proof of Theorem 1

THEOREM 1. *(i) The set of conditions (4) is necessary for all feasible capacity profile $\boldsymbol{S}$ to Problem (P1). (ii) If the capacity level $\boldsymbol{S}$ satisfies conditions (4), then there exists an allocation policy for Problem (P1) such that the service level requirement $\boldsymbol{\beta}$ can be attained for each and every product.*

PROOF. Part (i) is already proved in our discussion in Section 3.2 and Part (ii) is proved in Proposition 2. The main Theorem in this paper is hence proved. ■

B.4. Proof of Claim 1

CLAIM 1. *The function $g(\boldsymbol{\rho}) := \|\boldsymbol{\rho}^+\|_2^2$ is 1-smooth with respect to the Euclidean norm. Equivalently, for any $\boldsymbol{\rho},\boldsymbol{\psi}\in\mathbb{R}^{|\mathcal{I}|}$, the following inequality holds:*

$$\|\boldsymbol{\psi}^+\|_2^2 + 2\left(\boldsymbol{\psi}^+\right)^\top[\boldsymbol{\rho}-\boldsymbol{\psi}] \leq \|\boldsymbol{\rho}^+\|_2^2 \leq \|\boldsymbol{\psi}^+\|_2^2 + 2\left(\boldsymbol{\psi}^+\right)^\top[\boldsymbol{\rho}-\boldsymbol{\psi}] + \|\boldsymbol{\rho}-\boldsymbol{\psi}\|_2^2. \tag{22}$$

PROOF. We first prove the lower bound $\|\boldsymbol{\psi}^+\|_2^2 + 2\left(\boldsymbol{\psi}^+\right)^\top[\boldsymbol{\rho}-\boldsymbol{\psi}] \leq \|\boldsymbol{\rho}^+\|_2^2$ in (22) component-wise. For each $i\in\mathcal{I}$, we claim that

$$2\psi_i[\rho_i-\psi_i]\ \mathbf{1}(\psi_i>0) \leq \left[(\rho_i)^2\ \mathbf{1}(\rho_i>0)\right] - \left[\psi_i^2\ \mathbf{1}(\psi_i>0)\right]. \tag{41}$$

Inequality (41) is justified by the following cases:

- If $\psi_i\leq 0, \rho_i\leq 0$, then both sides of (41) are equal to 0.
- If $\psi_i\leq 0, \rho_i>0$, then (41) is equivalent to $0\leq\rho_i^2$.
- If $\psi_i>0, \rho_i\leq 0$, then (41) is equivalent to $2\psi_i[\rho_i-\psi_i]\leq -\psi_i^2$, which is clearly true.
- If $\psi_i>0, \rho_i>0$, then (41) is equivalent to $2\psi_i[\rho_i-\psi_i]\leq\rho_i^2-\psi_i^2$, which is equivalent to $0\leq[\rho_i-\psi_i]^2$, hence it is evidently true.

Altogether, the lower bound in (22) is proved by summing (41) over $i\in\mathcal{I}$.

We next prove the upper bound $\|\boldsymbol{\rho}^+\|_2^2 \leq \|\boldsymbol{\psi}^+\|_2^2 + 2\left(\boldsymbol{\psi}^+\right)^\top[\boldsymbol{\rho}-\boldsymbol{\psi}] + \|\boldsymbol{\rho}-\boldsymbol{\psi}\|_2^2$ in (22) component-wise, by demonstrating the inequality

$$\rho_i^2\ \mathbf{1}(\rho_i>0) \leq \left\{\psi_i^2 + 2\psi_i[\rho_i-\psi_i]\right\}\ \mathbf{1}(\psi_i>0) + [\rho_i-\psi_i]^2. \tag{42}$$

Inequality (42) is evidently true, by the following cases:

- If $\psi_i \le 0, \rho_i \le 0$, then (42) is equivalent to $0 \le [\rho_i - \psi_i]^2$, which is clearly true.
- If $\psi_i \le 0, \rho_i > 0$, then (42) is equivalent to $\rho_i^2 \le [\rho_i - \psi_i]^2$, which is true since $\rho_i - \psi_i = |\rho_i| + |\psi_i|$.
- If $\psi_i > 0, \rho_i \le 0$, then (42) is equivalent to $0 \le \{\psi_i^2 + 2\psi_i[\rho_i - \psi_i]\} + [\rho_i - \psi_i]^2$. Thus, it is equivalent to $0 \le \rho_i^2$, which is again clearly true.
- If $\psi_i > 0, \rho_i > 0$, then (42) is equivalent to $\rho_i^2 \le \{\psi_i^2 + 2\psi_i[\rho_i - \psi_i]\} + [\rho_i - \psi_i]^2$, which in fact holds with equality.

Altogether, the upper bound in (22) is established by summing (42) over $i \in \mathcal{I}$. Hence, (22) is established. ■

B.5. Proof of Lemma 1

LEMMA 1. *Consider the MFD policy. For each $i \in \mathcal{I}$, let*

$$\tilde{y}_i(t) := \mathbf{E}\left[\sum_{j \in \Gamma(\{i\})} D_{j,i}(t) \Big| \boldsymbol{\rho}(t)\right] \tag{23}$$

be the expected amount of resources allocated to product node i under the policy for the t^{th} sample of demand scenario, conditional on the average debt $\boldsymbol{\rho}(t)$ across the first $(t-1)$ samples. The following inequality holds for any realization of $\boldsymbol{\rho}(t)$:

$$\sum_{i \in \mathcal{I}} \rho_i^+(t)\tilde{y}_i(t) \ge \sum_{i \in \mathcal{I}} \rho_i^+(t) y_i^*, \tag{24}$$

where $\boldsymbol{y}^*$ (together with a suitably chosen $\boldsymbol{\rho}^*$) constitutes an optimal solution to (P-debt).

PROOF. The Lemma is proved by demonstrating that $\tilde{\boldsymbol{y}}(t)$ is an optimal solution to a certain optimization Problem (P-t) defined in the forthcoming analysis, and $\boldsymbol{y}^*$ is feasible for (P-t). For clarity sake, denote $\mathcal{L}$ as the priority list used for the t^{th} sample. By Lemma 5 (discussed in Appendix B.6), we know that for every node $\mathcal{L}_\ell \in \mathcal{I}$, we have

$$\tilde{y}_{\mathcal{L}_\ell}(t) = \mathbf{E}\left[f(U^{\mathcal{L}}(\ell)|\boldsymbol{X}(t), \boldsymbol{S}) - f(U^{\mathcal{L}}(\ell-1)|\boldsymbol{X}(t), \boldsymbol{S}) \mid \boldsymbol{\rho}(t)\right] \tag{43}$$

$$= \mathbf{E}_{\boldsymbol{X}}\left[f(U^{\mathcal{L}}(\ell)|\boldsymbol{X}, \boldsymbol{S}) - f(U^{\mathcal{L}}(\ell-1)|\boldsymbol{X}, \boldsymbol{S}) \mid \mathcal{L}\right]. \tag{44}$$

To justify the calculation, we remark that $\mathcal{L}$ is *deterministic* conditional on $\boldsymbol{\rho}(t)$, the average debt for the first $(t-1)$ samples, by our policy. Note that without conditioning on $\boldsymbol{\rho}(t)$, the priority list $\mathcal{L}$ is a random variable in general. Therefore, the expectation in (43) is solely on the randomness on $\boldsymbol{X}(t)$, the random demand scenario for the t^{th} sample, with $\mathcal{L}$ fixed.

The optimization Problem (P-t) is provided below:

$$\begin{aligned} \text{(P-t)}: \max_{\boldsymbol{y} \in \mathbb{R}_+^{|\mathcal{I}|}} \ & \sum_{i \in \mathcal{I}} \rho_i^+(t) y_i \\ s.t. \ & \sum_{i \in U} y_i \le \mathbf{E}\left[f\left(U \mid \boldsymbol{X}, \boldsymbol{S}\right)\right], \forall U \subseteq \mathcal{I} \\ & \sum_{i \in \mathcal{I}} y_i = \mathbf{E}\left[f\left(\mathcal{I} \mid \boldsymbol{X}, \boldsymbol{S}\right)\right]. \end{aligned}$$

We then prove the required inequality (24) by the following three observations, which are underlined.

1. The feasible region of (P-t) is a base polymatroid.
   By Lemma 3, the function $f(\cdot|\boldsymbol{X},\boldsymbol{S})$ is a non-decreasing submodular function with $f(\emptyset|\boldsymbol{X},\boldsymbol{S})=0$, for every realization of $\boldsymbol{X}$. Therefore, the function $\mathbf{E}[f(\cdot|\boldsymbol{X},\boldsymbol{S})]$ is also submodular, non-decreasing, and has the property $\mathbf{E}[f(\emptyset|\boldsymbol{X},\boldsymbol{S})]=0$. Our observation then follows by Definition 2 (stated in Appendix B.6).
2. The solution $\tilde{\boldsymbol{y}}(t)$ is an extreme point optimal solution to (P-t).
   By Lemma 5, $\boldsymbol{D}(t)$ is a lexi-cographical maximum flow associated with a permutation $\mathcal{L}(t)$ such that $\rho_{\mathcal{L}_1(t)}(t)\geq\rho_{\mathcal{L}_2(t)}(t)\geq\ldots\geq\rho_{\mathcal{L}_{|\mathcal{I}|}(t)}(t)$. This implies that we also have $\rho^+_{\mathcal{L}_1(t)}(t)\geq\rho^+_{\mathcal{L}_2(t)}(t)\geq\ldots\geq\rho^+_{\mathcal{L}_{|\mathcal{I}|}(t)}(t)$. By the expression of $\tilde{\boldsymbol{y}}$ in (44) and Lemma 4 (discussed in Appendix B.6), our observation is justified.
3. The solution $\boldsymbol{y}^*$ is feasible to (P-t).
   Constraints (18, 19) in (P-debt) are identical to the constraints in (P-t).

Altogether, the Lemma is proved. ■

B.6. Allocating Resources to Deterministic Demands, and Some Combinatorial Folklore

To facilitate our analysis in Section 3, we study the capacity allocation problem when the demand $\boldsymbol{X}$ is deterministic. We provide a characterization of the collection of feasible capacity allocations by *base polymatroids* (Edmonds (2003), Schrijver (2003)), which constitutes an important class of polyhedrons in combinatorial optimization. We utilize some of the beautiful properties of base polymatroids for our policy design and analysis in the stochastic allocation problem. While the results presented in this section are folklore in combinatorial optimization, their proofs are provided for completeness sake.

Suppose we are provided with capacity $\boldsymbol{S}$ and a realization of demand $\boldsymbol{X}$ for our production network $\mathcal{G}$. For a feasible flow $\boldsymbol{D}$ in $\mathcal{G}$, we define $\boldsymbol{y}\in\mathbb{R}^{|\mathcal{I}|}$ as the allocated capacity under $\boldsymbol{D}$, that is, we have $y_i=\sum_{j\in\Gamma(\{i\})}D_{j,i}$. Equivalently, we have $(\boldsymbol{y},\boldsymbol{D})\in P_f(\boldsymbol{S},\boldsymbol{X})$, where

$$P_f(\boldsymbol{S},\boldsymbol{X}):=\Bigg\{(\boldsymbol{y},\boldsymbol{D})\in\mathbb{R}_+^{|\mathcal{I}|}\times\mathbb{R}_+^{|\mathcal{G}|}:\ \sum_{j\in\Gamma(\{i\})}D_{j,i}=y_i,\ \forall i\in\mathcal{I} \tag{45}$$

$$\sum_{j\in\Gamma(\{i\})}D_{j,i}\leq X_i,\ \forall i\in\mathcal{I} \tag{46}$$

$$\sum_{i\in\Gamma(\{j\})}D_{j,i}\leq S_j,\ \forall j\in\mathcal{J}\Bigg\}. \tag{47}$$

In order to fully utilize the available capacity in $\mathcal{J}$, we focus our interest in the case when $\boldsymbol{D}$ is a *max-flow*. That is, $\boldsymbol{D}$ is feasible to $\mathcal{G}$ and

$$\sum_{i\in\mathcal{I}}\sum_{j\in\Gamma(\{i\})}D_{j,i}=\max_{\boldsymbol{D}'\in P_f(\boldsymbol{S},\boldsymbol{X})}\left\{\sum_{i\in\mathcal{I}}\sum_{j\in\Gamma(\{i\})}D'_{j,i}\right\}=\min_{V\subseteq\mathcal{I}}\left\{\sum_{j\in\Gamma(V)}S_j+\sum_{i\in\mathcal{I}\setminus V}X_i\right\},$$

where the last equality is by virtue of the max-flow-min-cut Theorem. Consequently, the collection of max-flows and their associated allocated capacities is equal to the following polyhedron:

$$P_{mf}(\boldsymbol{S},\boldsymbol{X}):=\Bigg\{(\boldsymbol{y},\boldsymbol{D})\in\mathbb{R}_+^{|\mathcal{I}|}\times\mathbb{R}_+^{|\mathcal{E}|}:\ (\boldsymbol{y},\boldsymbol{D})\in P_f(\boldsymbol{S},\boldsymbol{X})$$

$$\sum_{i\in\mathcal{I}}y_i=\ f(\mathcal{I}|\boldsymbol{X},\boldsymbol{S})\Bigg\}.$$

For any subset U of products, define $f(U|\boldsymbol{X},\boldsymbol{S})$ as the maximum capacity that can be allocated to these products given demand $\boldsymbol{X}$ and capacity $\boldsymbol{S}$ in the network. Using the max-flow-min-cut theorem, we have

$$f(U|\boldsymbol{X},\boldsymbol{S}) = \min_{V\subseteq U}\left\{\sum_{j\in\Gamma(V)} S_j + \sum_{k\in U\setminus V} X_k\right\}.$$

To facilitate our analysis for Section 3, we consider another polyhedron $P_{mc}(\boldsymbol{X},\boldsymbol{S})$, defined as follows:

$$P_{mc}(\boldsymbol{X},\boldsymbol{S}) := \Bigg\{\boldsymbol{y}\in\mathbb{R}_+^{|\mathcal{I}|}:\ \sum_{i\in U} y_i \le f(U|\boldsymbol{X},\boldsymbol{S}),\ \forall U\subseteq\mathcal{I} \qquad \sum_{i\in\mathcal{I}} y_i = f(\mathcal{I}|\boldsymbol{X},\boldsymbol{S}) \Bigg\}. \tag{48}$$

The following folklore Lemma relates P_{mf}, P_{mc} by a suitable application of the max-flow min-cut Theorem:

LEMMA 2. *For any $\boldsymbol{X},\boldsymbol{S}$, the polyhedrons $P_{mf}(\boldsymbol{X},\boldsymbol{S}), P_{mc}(\boldsymbol{X},\boldsymbol{S})$ are equivalent. That is, there exists $\boldsymbol{D}\in\mathbb{R}^{|\mathcal{G}|}$ such that $(\boldsymbol{y},\boldsymbol{D})\in P_{mf}(\boldsymbol{X},\boldsymbol{S})$ iff $\boldsymbol{y}\in P_{mc}(\boldsymbol{X},\boldsymbol{S})$.*

We provide a proof to Lemma 2 in Appendix B.7 for completeness sake. We refer to Megiddo (1974) for more technical details. After establishing the equivalence, we demonstrate that the alternative formulation P_{mc} is a base polymatroid, which is defined in the following:

DEFINITION 1. A function $g: 2^{|\mathcal{I}|}\to\mathbb{R}$ is submodular iff we have

$$g(U)+g(V)\ge g(U\cup V)+g(U\cap V) \quad \text{for any } U,V\subseteq\mathcal{I}.$$

In addition, g is a non-decreasing submodular function if the inequality above holds, and we also have $g(V)\le g(U)$ for any $V\subseteq U$.

DEFINITION 2. Let g be a non-decreasing submodular function, where $g(\emptyset)=0$. A polyhedron $P\subseteq\mathbb{R}^{|\mathcal{I}|}$ is a *base polymatroid* associated with g iff $P=\{\boldsymbol{y}\in\mathbb{R}_+^{|\mathcal{I}|}:\sum_{i\in U} y_i\le g(U)$ for all $U\subseteq\mathcal{I}$, and $\sum_{i\in\mathcal{I}} y_i = g(\mathcal{I})\}$.

Crucially, we demonstrate the submodularity of $f(U|\boldsymbol{X},\boldsymbol{S})$:

LEMMA 3. *For any $\boldsymbol{X},\boldsymbol{S}$, the function $f(\cdot|\boldsymbol{X},\boldsymbol{S}): 2^{|\mathcal{I}|}\to\mathbb{R}_+$ is a non-decreasing submodular function, with $f(\emptyset|\boldsymbol{X},\boldsymbol{S})=0$.*

The proof for Lemma 3 is provided in Appendix Section B.8. Definition 2 and Lemma 3 immediately lead us to the following Corollary:

COROLLARY 1. *P_{mc} is a base polymatroid associated with the submodular function $f(\cdot|\boldsymbol{X},\boldsymbol{S})$.*

The following property about an extreme point in a base polymatroid is pivotal in our design and analysis of the MFD policy.

LEMMA 4 **(Edmonds (2003), Schrijver (2003))**. *Let $P\subseteq\mathbb{R}_+^{|\mathcal{I}|}$ be the base polymatroid associated with a non-decreasing submodular function $g: 2^{|\mathcal{I}|}\to\mathbb{R}_+$, where $g(\emptyset)=0$. A solution $\bar{\boldsymbol{y}}\in P$ is an extreme point of P if and only if there exists a permutation $\mathcal{L}=(\mathcal{L}_1,\ldots,\mathcal{L}_{|\mathcal{I}|})$ on $\mathcal{I}$ such that:*

$$\bar{y}_{\mathcal{L}_\ell} = g(U^{\mathcal{L}}(\ell)) - g(U^{\mathcal{L}}(\ell-1))$$

for all $\ell\in\mathcal{I}$, where we recall our notation that $U^{\mathcal{L}}(\ell)=\{\mathcal{L}_1,\ldots,\mathcal{L}_\ell\}$. In addition, for any $\boldsymbol{\rho}\in\mathbb{R}^{|\mathcal{I}|}$, let $\mathcal{L}$ be a permutation such that $\rho_{\mathcal{L}_1}\ge\rho_{\mathcal{L}_2}\ge\cdots\ge\rho_{\mathcal{L}_{|\mathcal{I}|}}$. Then, for the linear optimization problem $\max_{\boldsymbol{y}\in P}\boldsymbol{\rho}^\top\boldsymbol{y}$, the extreme point solution $\bar{\boldsymbol{y}}$, defined by $\bar{y}_{\mathcal{L}_\ell} = g(U^{\mathcal{L}}(\ell)) - g(U^{\mathcal{L}}(\ell-1))$ for all $\ell\in\mathcal{I}$, is optimal.

The proof of Lemma 4 is provided in page 772 in Schrijver (2003). The proof is based on demonstrating that the greedy algorithm returns an optimal solution to a linear optimization problem over a base polymatroid. Lemma 4 and Corollary 1 directly lead us to Lemma 5, which characterizes the allocated capacity associated with a priority list.

LEMMA 5. *Let $\mathcal{L}$ be a priority list. For each $\ell \in \mathcal{I}$, a lexi-cographical maximum flow $\boldsymbol{D}$ associated with $\mathcal{L}$ allocates a capacity of*

$$\sum_{j \in \Gamma(\{\mathcal{L}_\ell\})} D_{j,\mathcal{L}_\ell}(\boldsymbol{X}, \boldsymbol{S}) = f(U^{\mathcal{L}}(\ell)|\boldsymbol{X}, \boldsymbol{S}) - f(U^{\mathcal{L}}(\ell-1)|\boldsymbol{X}, \boldsymbol{S}) \tag{49}$$

to demand node $\mathcal{L}_\ell$, where $U^{\mathcal{L}}(l)$ denotes the subset of top l products under priority list $\mathcal{L}$, i.e., $U^{\mathcal{L}}(l) := \{\mathcal{L}_1, \ldots, \mathcal{L}_l\}$ if $\mathcal{L} = (\mathcal{L}_1, \mathcal{L}_2, \ldots, \mathcal{L}_{|\mathcal{I}|})$.

We remark that the allocation quantity can also be determined from Lemma 5, but Problem (Q) proposed in the MFD policy is much more computationally efficient.

B.7. Proof of Lemma 2

LEMMA 2. *For any $\boldsymbol{X}, \boldsymbol{S}$, the polyhedrons $P_{mf}(\boldsymbol{X}, \boldsymbol{S}), P_{mc}(\boldsymbol{X}, \boldsymbol{S})$ are equivalent. That is, there exists $\boldsymbol{D} \in \mathbb{R}^{|\mathcal{G}|}$ such that $(\boldsymbol{y}, \boldsymbol{D}) \in P_{mf}(\boldsymbol{X}, \boldsymbol{S})$ iff $\boldsymbol{y} \in P_{mc}(\boldsymbol{X}, \boldsymbol{S})$.*

PROOF. $\underline{\exists \boldsymbol{D} \text{ s.t. } (\boldsymbol{y}, \boldsymbol{D}) \in P_{mf}(\boldsymbol{X}, \boldsymbol{S}) \Rightarrow \boldsymbol{y} \in P_{mc}(\boldsymbol{X}, \boldsymbol{S}).}$

This implication is justified by (2).

$\underline{\boldsymbol{y} \in P_{mc}(\boldsymbol{X}, \boldsymbol{S}) \Rightarrow \exists \boldsymbol{D} \text{ s.t. } (\boldsymbol{y}, \boldsymbol{D}) \in P_{mf}(\boldsymbol{X}, \boldsymbol{S}).}$

We justify the implication by applying the max-flow-min-cut Theorem on the arc-capacitated network $\hat{\mathcal{G}}$, an augmentation of $\mathcal{G}$, defined as follows. The network $\hat{\mathcal{G}}$ has nodes $\{s\} \uplus \mathcal{J} \uplus \mathcal{I} \uplus \{t\}$. Node s is the source, and node t is the sink. The arcs in $\hat{\mathcal{G}}$ are defined as follows:

1. For s and every $j \in \mathcal{J}$, there is an arc $\overrightarrow{sj}$ in $\hat{\mathcal{G}}$ with capacity S_j.
2. For every $j \in \mathcal{J}, i \in \mathcal{I}$ such that $j \in \Gamma(\{i\})$ in $\mathcal{G}$, there is an arc $\overrightarrow{ji}$ in $\hat{\mathcal{G}}$ with capacity ∞.
3. For every $i \in \mathcal{I}$ and t, there is an arc $\overrightarrow{it}$ in in $\hat{\mathcal{G}}$ with capacity y_i.

To demonstrate implication, it suffices to show that there exists a feasible flow in $\hat{\mathcal{G}}$ with flow value equal to $\sum_{i \in \mathcal{I}} y_i = f(\mathcal{I}|\boldsymbol{X}, \boldsymbol{S})$. Indeed, for a flow in $\hat{\mathcal{G}}$ that achieves such a flow value, the flow in the arc $\overrightarrow{it}$ must have value y_i for every $i \in \mathcal{I}$. By setting $D_{j,i}(\boldsymbol{X}, \boldsymbol{S})$ to be the flow in the arc $\overrightarrow{ji}$, we assert that the solution $(\boldsymbol{y}, \boldsymbol{D})$ satisfies all the constraints in $P_f(\boldsymbol{X}, \boldsymbol{S})$. Our assertion follows from the following checklist on the feasibility of $(\boldsymbol{y}, \boldsymbol{D})$ to $P_f(\boldsymbol{X}, \boldsymbol{S})$. By specializing the constraints (48) with U being singletons, we know that the inequality $y_i \le X_i$ holds. In particular, by the conservation of flow at node i in $\hat{\mathcal{G}}$, we have $\sum_{j \in \Gamma(\{i\})} D_{j,i}(\boldsymbol{X}, \boldsymbol{S}) = y_i \le X_i$, meaning that the constraints (45, 46) hold for the solution $(\boldsymbol{y}, \boldsymbol{D})$. By the conservation of flow at node j in $\hat{\mathcal{G}}$, constraints (47) hold for the solution $(\boldsymbol{y}, \boldsymbol{D})$.

For our analysis, we define two notations. First, for an arc $\overrightarrow{ab}$ in $\hat{G}$, we denote $\text{cap}(\overrightarrow{ab})$ as the capacity of $\overrightarrow{ab}$. Second, for $C \subseteq \{s\} \uplus \mathcal{J} \uplus \mathcal{I} \uplus \{t\}$, we denote $\delta^+(C) = \{\overrightarrow{ab} : a \in C, b \notin C\}$.

To argue for the existence of the required flow, it suffices to show that for any cut C that separates the source s from the sink t, i.e. $C \subseteq \{s\} \uplus \mathcal{J} \uplus \mathcal{I}$ and $C \ni s$, we have

$$\sum_{\overrightarrow{ab} \in \delta^+(C)} \text{cap}(\overrightarrow{ab}) \geq \sum_{i \in \mathcal{I}} y_i. \tag{50}$$

Due to the complexity of $\hat{\mathcal{G}}$, the analysis for a generic cut C could be complicated. Now, we claim the following. For a generic cut C, let $C_{\mathcal{I}} := C \cap \mathcal{I}$, $C_{\mathcal{J}} := C \cap \mathcal{J}$, and also let $C' := \{s\} \uplus \{\mathcal{J} \setminus \Gamma(\mathcal{I} \setminus C_{\mathcal{I}})\} \uplus C_{\mathcal{I}}$. We claim that

$$\sum_{\overrightarrow{ab} \in \delta^+(C)} \text{cap}(\overrightarrow{ab}) \geq \sum_{\overrightarrow{ab} \in \delta^+(C')} \text{cap}(\overrightarrow{ab}). \tag{51}$$

To see that (51), we argue for the following two cases. If $C_{\mathcal{J}} \not\subseteq \mathcal{J} \setminus \Gamma(\mathcal{I} \setminus C_{\mathcal{I}})$, then there exists an arc from $C_{\mathcal{J}}$ to $\mathcal{I} \setminus C_{\mathcal{I}}$, which has a capacity of ∞, immediately implying (51). Else, suppose that $C_{\mathcal{J}} \subseteq \mathcal{J} \setminus \Gamma(\mathcal{I} \setminus C_{\mathcal{I}})$, but $C_{\mathcal{J}} \neq \mathcal{J} \setminus \Gamma(\mathcal{I} \setminus C_{\mathcal{I}})$. Now, for any $j \in \{\mathcal{J} \setminus \Gamma(\mathcal{I} \setminus C_{\mathcal{I}})\} \setminus C_{\mathcal{J}}$, we know that $\Gamma(\{j\}) \subset C_{\mathcal{I}}$. Therefore, we have

$$\sum_{\overrightarrow{ab} \in \delta^+(C')} \text{cap}(\overrightarrow{ab}) = \sum_{\overrightarrow{ab} \in \delta^+(C)} \text{cap}(\overrightarrow{ab}) - \sum_{j \in \left\{\mathcal{J} \setminus \Gamma(\mathcal{I} \setminus C_{\mathcal{I}})\right\} \setminus C_{\mathcal{J}}} \text{cap}(\overrightarrow{sj}) \leq \sum_{\overrightarrow{ab} \in \delta^+(C)} \text{cap}(\overrightarrow{ab}).$$

Consequently, inequality (51) is established.

Therefore, to prove the existence of the required flow, it suffices to show that (50) holds for all cut C for which $C = \{s\} \uplus \mathcal{J} \setminus \Gamma(\mathcal{I} \setminus U) \uplus U$ for some $U \subseteq \mathcal{I}$. Explicitly, now it suffices to show that

$$\sum_{\overrightarrow{ab} \in \delta^+(\{s\} \uplus \mathcal{J} \setminus \Gamma(\mathcal{I} \setminus U) \uplus U)} \text{cap}(\overrightarrow{ab}) = \sum_{j \in \Gamma(\mathcal{I} \setminus U)} S_j + \sum_{i \in U} y_i \geq \sum_{i \in \mathcal{I}} y_i, \, \forall U \subseteq \mathcal{I}.$$

Equivalently, it suffices to have

$$\sum_{j \in \Gamma(\mathcal{I} \setminus U)} S_j \geq \sum_{i \in \mathcal{I} \setminus U} y_i, \, \forall U \subseteq \mathcal{I}, \tag{52}$$

which is the final step in the proof. For every U, (52) is evidently true by considering the constraint (48) in $P_{mc}(\boldsymbol{X}, \boldsymbol{S})$ for the set $\mathcal{I} \setminus U$:

$$\sum_{i \in \mathcal{I} \setminus U} y_i \leq f(\mathcal{I} \setminus U | \boldsymbol{X}, \boldsymbol{S}) = \min_{V \subseteq \mathcal{I} \setminus U} \left\{ \sum_{j \in \Gamma(V)} S_j + \sum_{i \in (\mathcal{I} \setminus U) \setminus V} X_i \right\} \leq \sum_{j \in \Gamma(\mathcal{I} \setminus U)} S_j.$$

Altogether, we have shown (50) for any cut C, hence demonstrated the existence of the required flow. ■

REMARK 1. Interestingly, the proof mentioned above implicitly establishes that $P_{mc}(\boldsymbol{X}, \boldsymbol{S})$ is equivalent to the following polytope:

$$P'_{mc}(\boldsymbol{X}, \boldsymbol{S}) := \left\{ \boldsymbol{y} \in \mathbb{R}_+^{|\mathcal{I}|} : \begin{array}{ll} \sum_{i \in U} y_i \leq \sum_{j \in \Gamma(U)} S_j, & \forall U \subseteq \mathcal{I} \\ \sum_{i \in \mathcal{I}} y_i = f(\mathcal{I} | \boldsymbol{X}, \boldsymbol{S}) & \\ y_i \leq X_i, & \forall i \in \mathcal{I} \end{array} \right\}.$$

B.8. Proof of Lemma 3

LEMMA 3. *For any $\boldsymbol{X}, \boldsymbol{S}$, the function $f(\cdot|\boldsymbol{X}, \boldsymbol{S}) : 2^{|\mathcal{I}|} \to \mathbb{R}_+$ is a non-decreasing submodular function, with $f(\emptyset|\boldsymbol{X}, \boldsymbol{S}) = 0$.*

PROOF. It is clear that $f(\emptyset|\boldsymbol{X}, \boldsymbol{S}) = 0$, and it is also evident that $f(\cdot|\boldsymbol{X}, \boldsymbol{S})$ is non-decreasing, since $W \subseteq U$, letting $V \subseteq U$ be a subset such that $f(U|\boldsymbol{X}, \boldsymbol{S}) = \min_{V' \subseteq U} \left\{ \sum_{j \in \Gamma(V')} S_j + \sum_{i \in U \setminus V'} X_i \right\} = \sum_{j \in \Gamma(V)} S_j + \sum_{i \in U \setminus V} X_i$, we have

$$f(U|\boldsymbol{X}, \boldsymbol{S}) = \sum_{j \in \Gamma(V)} S_j + \sum_{i \in U \setminus V} X_i \geq \sum_{j \in \Gamma(W \cap V)} S_j + \sum_{i \in W \setminus (W \cap V)} X_i \geq f(W|\boldsymbol{X}, \boldsymbol{S}).$$

Now, we demonstrate that $f(\cdot|\boldsymbol{X}, \boldsymbol{S})$ is submodular. For any $U_1, U_2 \subseteq \mathcal{I}$, let $V_1 \subseteq U_1, V_2 \subseteq U_2$ satisfy

$$f(U_1|\boldsymbol{X}, \boldsymbol{S}) = \min_{V \subseteq U_1} \left\{ \sum_{j \in \Gamma(V)} S_j + \sum_{i \in U_1 \setminus V} X_i \right\} = \sum_{j \in \Gamma(V_1)} S_j + \sum_{i \in U_1 \setminus V_1} X_i,$$

$$f(U_2|\boldsymbol{X}, \boldsymbol{S}) = \min_{V \subseteq U_2} \left\{ \sum_{j \in \Gamma(V)} S_j + \sum_{i \in U_2 \setminus V} X_i \right\} = \sum_{j \in \Gamma(V_2)} S_j + \sum_{i \in U_2 \setminus V_2} X_i.$$

We then have

$$\begin{aligned} f(U_1|\boldsymbol{X}, \boldsymbol{S}) + f(U_2|\boldsymbol{X}, \boldsymbol{S}) &= \sum_{j \in \Gamma(V_1)} S_j + \sum_{j \in \Gamma(V_2)} S_j + \sum_{i \in U_1 \setminus V_1} X_i + \sum_{i \in U_2 \setminus V_2} X_i \\ &= \sum_{j \in \Gamma(V_1 \cup V_2)} S_j + \sum_{j \in \Gamma(V_1 \cap V_2)} S_j + \sum_{i \in (U_1 \cup U_2) \setminus (V_1 \cup V_2)} X_i + \sum_{i \in (U_1 \cap U_2) \setminus (V_1 \cap V_2)} X_i \\ &\geq f(U_1 \cup U_2|\boldsymbol{X}, \boldsymbol{S}) + f(U_1 \cap U_2|\boldsymbol{X}, \boldsymbol{S}). \end{aligned}$$

The lemma is hence proved. ■

B.9. Proof of Proposition 1

PROPOSITION 1. *In a bipartite network, the optimal weighted maximum flow derived from Problem (Q) is not only a max-flow solution to the whole network, but also a lexi-cographical maximum flow associated with the priority list $\mathcal{L}(t+1)$.*

PROOF. Without loss of generality, we assume that the product nodes are ordered from 1 to $|\mathcal{I}|$ according to the priority list $\mathcal{L}(t+1)$. For simplicity, the optimal weighted maximum flow solution to Problem (Q) is denoted as $\boldsymbol{D}$, instead of $\boldsymbol{D}^{\mathcal{L}(t+1)}(\boldsymbol{X}(t+1), \boldsymbol{S})$ used in the main text. Then the total flow delivered to node i is $\sum_{j \in \Gamma(\{i\})} D_{j,i}$. Denote the total weighted flow to node i as $\sum_{j \in \Gamma(\{i\})} w_i D_{j,i}$. To break ties if necessary, we can perturb the weight slightly to induce a strict ordering. Therefore, we assume that the weights follows $w_1 > w_2 > \ldots > w_{|\mathcal{I}|} > 0$.

First, suppose the optimal solution $\boldsymbol{D}$ is not a maximum flow in the network, then there exists an augmenting path in the network. By pushing the minimal amount of flow from a supply node to a demand node—which does not violate the capacity constraints—along the augmenting path, the objective of Problem (Q) increases, which contradicts the optimality of $\boldsymbol{D}$. Therefore, we claim that $\boldsymbol{D}$ is a maximum flow to the network.

Next, we use induction to prove that $\boldsymbol{D}$ is also lexi-cographically maximal, associated with the priority list $\mathcal{L}$. Starting from the first demand node 1 with the highest priority, if there exists another solution $\hat{\boldsymbol{D}}$ such that $\sum_{j\in\Gamma(\{1\})}\hat{D}_{j,1} > \sum_{j\in\Gamma(\{1\})} D_{j,1}$, then there exists at least one $j\in\Gamma(\{1\})$ such that $\hat{D}_{j,1} > D_{j,1}$. Increasing $D_{j,1}$ by an amount $\epsilon>0$, and decreasing $D_{j,k}$ for some $k>1$ if necessary, we can strictly increase the objective value since the weight on demand node 1 is larger than the weight on k. This contradicts the assumption that $\boldsymbol{D}$ is optimal.

Assume that $\boldsymbol{D}$ is lexi-cographically maximal for demand nodes in $U_{k-1}=\{1,2,\ldots,k-1\}$. We show that $\boldsymbol{D}$ is also lexi-cographically maximal for demand nodes in $U_k = U_{k-1}\cup\{k\}$ by contradiction. Suppose there exists another lexi-cographical maximal solution $\tilde{\boldsymbol{D}}$ such that $\sum_{i\in U_k}\sum_{j\in\Gamma(U_k)}\tilde{D}_{j,i} > \sum_{i\in U_k}\sum_{j\in\Gamma(U_k)} D_{j,i}$, and $\sum_{i\in U_{k-1}}\sum_{j\in\Gamma(U_{k-1})}\tilde{D}_{j,i} = \sum_{i\in U_{k-1}}\sum_{j\in\Gamma(U_{k-1})} D_{j,i}$. Then $\sum_{j\in\Gamma(k)}\tilde{D}_{j,k} > \sum_{j\in\Gamma(k)} D_{j,k}$. Consider the flow $\Delta\boldsymbol{D} := (\tilde{\boldsymbol{D}}_{i,j} - \boldsymbol{D}_{i,j})$, interpreting the flow to be in the reverse direction if the value is negative. Note that all the product nodes in $\{1,\ldots,k-1\}$ have balanced in-flow and out-flow, whereas the product node k has a net in-flow. We can therefore find an augmenting flow in $\Delta\boldsymbol{D}$, starting either from a supply node or a product node $k'\notin U_k$, and add it to $\boldsymbol{D}$, so that the in-flow into k increases by ϵ, decreasing the flow into k' by ϵ if necessary. Since $w_{k'} < w_k$, this again leads to a contradiction that $\boldsymbol{D}$ is optimum in our model. Therefore, we claim that $\boldsymbol{D}$ is also a lexi-cographical maximum flow associated with the priority list $\mathcal{L}$. ■

B.10. Proof of Proposition 3

Before we proceed to the proof of Proposition 3, we first formalize the problem setting and incentive issues for product managers. Recall that we study the single-period resource allocation problem in a flexible production network, and our anticipative allocation priority list is determined without using the realized demand information. We assume that all product managers are rational and they aim to maximize their expected utilities. For each product manager $i\in\mathcal{I}$, its utility function is defined as follows:

$$u_i\left(y_i(\hat{X}_i,\hat{X}_{-i})\right) := r_i \min\left(y_i(\hat{X}_i,\hat{X}_{-i}),X_i\right) - c_i y_i\left(\hat{X}_i,\hat{X}_{-i}\right) - h_i\left(y_i(\hat{X}_i,\hat{X}_{-i}) - X_i\right)^+ - s_i(X_i - y_i\left(\hat{X}_i,\hat{X}_{-i}\right))^+,$$

where $y_i(\hat{X}_i,\hat{X}_{-i})$ denotes the amount of resources received by product manager i when she reports a demand value $\hat{X}_i$ and the other managers (in the subset $\mathcal{I}/\{i\}$) report $\hat{X}_{-i}$; Note that $\hat{X}_i$ may not be equal to the actual demand X_i as the product manager may misreport her demand, similarly for $\hat{X}_{-i}$; r_i represents the unit profit by fulfilling the demand; c_i, h_i, and s_i represent the unit purchasing cost, holding cost, and shortage cost, respectively. Without loss of generality, we assume that r_i, c_i, h_i, s_i are all strictly positive and $r_i > c_i$ for all $i\in\mathcal{I}$.

To demonstrate that the randomized allocation mechanism is strategy-proof, we need to show that $u_i\left(y_i(X_i,\hat{X}_{-i})\right) \geq u_i\left(y_i(\hat{X}_i,\hat{X}_{-i})\right)$ for each and every product manager i, i.e., there is no incentive for any product manager to misrepresent her actual demands. We show next that the anticipative nature of the priority list is the key to support this argument.

PROPOSITION 3. *The randomized allocation mechanism is strategy-proof in the sense that truth telling is a dominant strategy for each product manager.*

Proof. After the allocation priority list is determined, product managers report their demands and then (1) some managers are fully or partially served if they are at higher positions in the priority list; (2) some are not served if their priorities are lower and the capacity is not sufficient. We show next that reporting the actual demand is a dominant strategy for each product manager in both scenarios, i.e., (1) and (2).

(1) In the first scenario, suppose product manager i misreports her demand $\hat{X}_i$ and the amount of resources allocated to her is $y_i(\hat{X}_i, \hat{X}_{-i}) > 0$. Note that the received amount cannot exceed the reported volume and hence there are four cases to consider under this scenario:

- If she over-reports $\hat{X}_i > X_i$, and she receives $X_i < y_i(\hat{X}_i, \hat{X}_{-i}) \le \hat{X}_i$, then her utility is $u_i\left(y_i(\hat{X}_i, \hat{X}_{-i})\right) = r_i X_i - c_i y_i(\hat{X}_i, \hat{X}_{-i}) - h_i\left(y_i(\hat{X}_i, \hat{X}_{-i}) - X_i\right)^+$. However, if she reports the actual demand X_i, she would have received $y_i(X_i, \hat{X}_{-i}) = X_i$ and her utility would be $u_i\left(y_i(X_i, \hat{X}_{-i})\right) = r_i X_i - c_i X_i > u_i\left(y_i(\hat{X}_i, \hat{X}_{-i})\right)$.
- If she over-reports $\hat{X}_i > X_i$, and she receives $y_i(\hat{X}_i, \hat{X}_{-i}) \le X_i < \hat{X}_i$, then her utility is $u_i\left(y_i(\hat{X}_i, \hat{X}_{-i})\right) = r_i y_i(\hat{X}_i, \hat{X}_{-i}) - c_i y_i(\hat{X}_i, \hat{X}_{-i}) - s_i\left(X_i - y_i(\hat{X}_i, \hat{X}_{-i})\right)^+$. In this case, if she reports the actual demand X_i, she also receives $y_i(X_i, \hat{X}_{-i}) = y_i(\hat{X}_i, \hat{X}_{-i}) \le X_i$, and her utility would be the same as reporting $\hat{X}_i$.
- If she under-reports $\hat{X}_i < X_i$, and she receives $y_i(\hat{X}_i, \hat{X}_{-i}) \le \hat{X}_i < X_i$, then her utility is $u_i\left(y_i(\hat{X}_i, \hat{X}_{-i})\right) = r_i y_i(\hat{X}_i, \hat{X}_{-i}) - c_i y_i(\hat{X}_i, \hat{X}_{-i}) - s_i\left(X_i - y_i(\hat{X}_i, \hat{X}_{-i})\right)^+$. In this case, if she reports the actual demand X_i, she also receives $y_i(X_i, \hat{X}_{-i}) = y_i(\hat{X}_i, \hat{X}_{-i}) \le \hat{X}_i$, and her utility would be the same as reporting $\hat{X}_i$.
- If she under-reports $\hat{X}_i < X_i$, and there is sufficient capacity to assign her more than $\hat{X}_i$, but the amount that she can receive is capped by her reported demand, she can only receive $y_i(\hat{X}_i, \hat{X}_{-i}) = \hat{X}_i$. In this case, her utility is $u_i\left(y_i(\hat{X}_i, \hat{X}_{-i})\right) = r_i \hat{X}_i - c_i \hat{X}_i - s_i\left(X_i - \hat{X}_i\right)^+$. However, if she reports the actual demand X_i, she would have received $\hat{X}_i < y_i(X_i, \hat{X}_{-i}) \le X_i$, and her utility would be $u_i\left(y_i(X_i, \hat{X}_{-i})\right) = r_i y_i(X_i, \hat{X}_{-i}) - c_i y_i(X_i, \hat{X}_{-i}) - s_i\left(X_i - y_i(X_i, \hat{X}_{-i})\right)^+$. Since $\hat{X}_i < y_i(X_i, \hat{X}_{-i})$ in this case, it follows that $u_i\left(y_i(X_i, \hat{X}_{-i})\right) > u_i\left(y_i(\hat{X}_i, \hat{X}_{-i})\right)$.

Considering the possibility of each case, reporting the true demand information is the dominant strategy for product manager i.

(2) In the second scenario when the produce manager is not served, so the reported demand will not affect the resources allocated to her since the priority has already been determined before she reports her demand.

To conclude, in both scenarios, truth telling (reporting the actual demand) is a dominant strategy for each product manager and hence the randomized allocation mechanism is strategy-proof. ■

C. OCO-Based Capacity Updating Algorithm

Note that in Section 4, the key regret bound obtained from Equation (14) relies on the online gradient descent algorithm, an important subroutine in Algorithm 2. For completeness, we describe this algorithm as Algorithm 3 in this appendix.

Algorithm 3 Online Gradient Descent Algorithm for Iteration k in Step 1 of Algorithm 2

* *Input: Lagrangian dual multiplier $\boldsymbol{\lambda}^k$; T demand scenarios $\boldsymbol{X}(t), t=1,2,\ldots,T$; initial capacity level $\boldsymbol{S}^k(0)=\bar{\boldsymbol{S}}^{k-1}$ (expected capacity profile from previous iteration).*

* *Output: Expected capacity profile $\bar{\boldsymbol{S}}^k$; expected capacity allocation quantity $\boldsymbol{D}^k$.*

1. For $t=1,2,\ldots,T$, do the following:

   - Solve the following primal problem given the demand scenario $\boldsymbol{X}(t)$ and the Lagrangian dual multiplier $\boldsymbol{\lambda}^k$:

$$\begin{aligned}
\min_{\boldsymbol{D}(t)} \ & \sum_{i\in\mathcal{I}} \lambda_i^k \left\{ - \sum_{j\in\Gamma(\{i\})} D_{j,i}(t) \right\} \\
\text{s.t.} \ & \sum_{j\in\Gamma(\{i\})} D_{j,i}(t) \le X_i(t), \forall i\in\mathcal{I} \\
& \sum_{i\in\Gamma(\{j\})} D_{j,i}(t) \le S_j^k(t-1), \forall j\in\mathcal{J} \\
& D_{j,i}(t) \ge 0, \forall (i,j)\in\mathcal{G}
\end{aligned}$$

   - Solve the dual problem of the above problem:

$$\begin{aligned}
\max_{\boldsymbol{p}(t),\boldsymbol{q}(t)} \ & \sum_{i\in\mathcal{I}} \left\{ p_i(t) X_i(t) \right\} + \sum_{j\in\mathcal{J}} \left\{ q_j(t) S_j^k(t-1) \right\} \\
\text{s.t.} \ & p_i(t) + q_j(t) \le (-\lambda_i^k), \forall (i,j)\in\mathcal{G} \\
& p_i(t) \le 0, \forall i\in\mathcal{I} \\
& q_j(t) \le 0, \forall j\in\mathcal{J}
\end{aligned}$$

   - Update the capacity using the online gradient descent algorithm (Zinkevich 2003):

$$S_j^k(t) := \max\left\{0, S_j^k(t-1) - \eta_t \nabla_{S_j^k(t-1)} f\left(\boldsymbol{S}^k(t-1), \boldsymbol{X}(t)\right)\right\},$$

   where $\eta_t := 1/\sqrt{t}$ and $\nabla_{S_j^k(t-1)} f\left(\boldsymbol{S}^k(t-1), \boldsymbol{X}(t)\right) := c_j + q_j(t)$ represent the step size and step direction, respectively. Note that $q_j(t)$ is the optimal solution to the above dual problem, and thus its value depends on $\boldsymbol{S}^k(t-1)$.

2. Compute the average capacity profile over T scenarios, $\bar{\boldsymbol{S}}^k = (1/T)\sum_{t=1}^T \boldsymbol{S}^k(t-1)$.

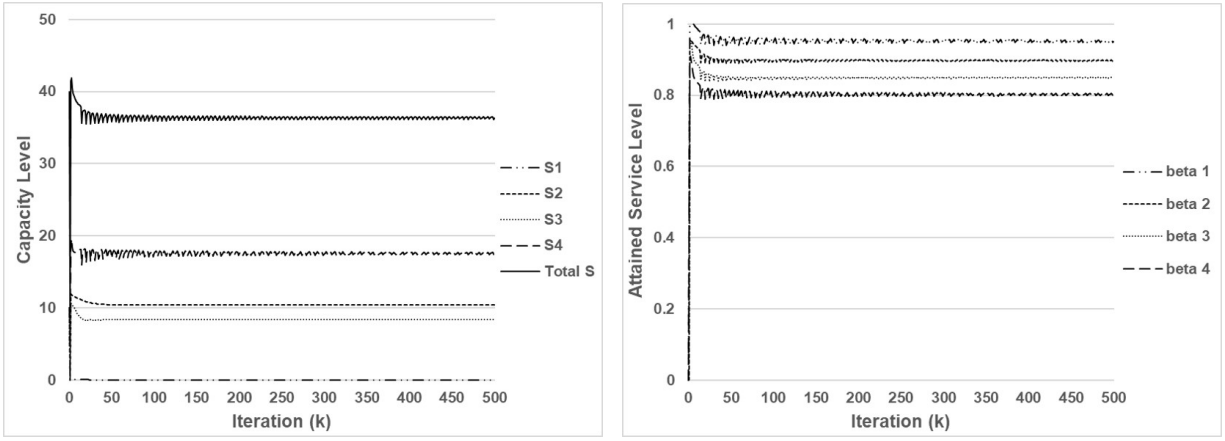
3. Compute the expected capacity allocation quantity from supplier j to customer i over T scenarios, $D_{j,i}^k := (1/T)\sum_{t=1}^T D_{j,i}^*(t)$, for all $(j,i)\in\mathcal{G}$, where $D_{j,i}^*(t)$ denotes the optimal solution to the primal problem at scenario t.

In Step 2, the sub-gradient $\nabla_{S_j^k(t-1)} f\left(\boldsymbol{S}^k(t-1), \boldsymbol{X}(t)\right)$ is composed of two components: (1) c_j comes from the coefficient of $c_j S_j^k(t-1)$, and (2) $q_j(t)$ refers to the shadow price of the constraints in which $S_j^k(t-1)$'s are involved. In addition, $S_j^k(t)$ should be projected to the feasible domain $[0, +\infty)$ whenever it is negative.

We note that the regret bounds in Inequalities (14) and (15) require some conditions on Problem (P4): (1) the feasible region of $\boldsymbol{S}$ must be bounded; (2) function $f_t(\boldsymbol{S}, \boldsymbol{X}(t))$ must be convex and Lipschitz-continuous with respect to $\boldsymbol{S}$. To enforce the first condition, we can place a redundant upper bound on the feasible capacity levels without loss of generality. For the second condition, recall that the first term in $f_t(\boldsymbol{S}, \boldsymbol{X}(t))$ is linear in $\boldsymbol{S}$ and the second term is a minimization problem with $\boldsymbol{S}$ in the right-hand side of the constraints. It is well-known that linear minimization programs are convex and Lipschitz continuous with respect to changes in the right-hand side data of the problem. Therefore, the second condition is also satisfied.

In the end, we provide a numerical example to validate the performance of Algorithm 2. Consider a (4×4) long-chain production network, in which four plants with capacity configuration cost $\boldsymbol{c} = [4, 3, 2, 1]$ are faced with i.i.d. normal demands with mean $\boldsymbol{\mu} = [10, 10, 10, 10]$ and standard deviation $\boldsymbol{\sigma} = [3, 3, 3, 3]$. The service level requirements are $\boldsymbol{\beta} = [0.95, 0.90, 0.85, 0.80]$. We sample $T = 10^4$ scenarios to calculate the average capacity level $\bar{\boldsymbol{S}}^k$ at each iteration k. Figure 8 plots the capacity levels and attained service levels as k increases. The capacity levels converge to the optimal ones around 100 iterations, similarly for the service levels. We implemented this algorithm using MATLAB and CPLEX as the linear programming solver on a 2.70 GHz i7-6820HQ CPU Windows PC with 16GB RAM. It takes 11.5 seconds in CPU time to solve for each iteration.

Figure 8 Numerical performance of Algorithm 2



(a) Capacity level

(b) Attained service level