Evaluating Fairness of Sequential Resource Allocation Policies: A Computational Study

Christopher Hojny
Department of Mathematics and Computer Science,
Eindhoven University of Technology, The Netherlands

Frits C. R. Spieksma
Department of Mathematics and Computer Science,
Eindhoven University of Technology, The Netherlands

Sten Wessel[1]
Department of Mathematics and Computer Science,
Eindhoven University of Technology, The Netherlands

Abstract

In the sequential resource allocation problem there is a single divisible resource that is divided over a number of clients. Allocations are made in a predetermined order and only upon arrival at a client their demand for the resource is revealed; only the probability distribution of the demand of every client is known to the supplier. We consider this problem from a fairness perspective, where the aim is to balance allocations between individual clients. Several allocation policies have been proposed in the literature. In this work, we introduce a new, non-adaptive policy based on linear programming that can also incorporate group fairness. In addition, we provide an extensive computational study to compare allocation policies on several fairness measures. Using an optimized implementation of existing methods, we are able to evaluate significantly larger problem instances than those previously considered in the literature.

2012 ACM Subject Classification Applied computing → Operations research

Keywords and phrases fairness, resource allocation, computational analysis

Digital Object Identifier 10.4230/OASIcs.ATMOS.2025.7

Supplementary Material *Software (Source Code)*: `https://doi.org/10.5281/zenodo.15825047`

Funding This research is supported by NWO Gravitation Project NETWORKS, The Netherlands, Grant Number 024.002.003.

1 Introduction

We consider the following *sequential resource allocation problem*. There is a (single) divisible resource with an initial supply s, which has to be divided over n clients. Clients are visited sequentially, in a predetermined order, identified as $[n] := \{1, \dots, n\}$. Each client has a certain *demand*, which is the amount of the resource they want to receive from the supplier. However, the demands are unknown to the supplier beforehand; it is only revealed when arriving at the client. Instead, the probability distribution for every client is known to the supplier in advance. Let D_i for every client $i \in [n]$ denote the random variable modelling the demand of client i. We assume that the demand distributions for every client are *independent*, *discrete*, and take a finite number of values. The initial supply s, the set of demand distributions $\{D_i : i \in [n]\}$ for every client, together with a specific objective (Section 2.1) define the Sequential Resource Allocation (SRA) problem. A solution

[1] Corresponding author

to the SRA problem comes in the form of a *policy* that determines the amount x_i of the resource that is allocated to client $i \in [n]$ upon visiting that client, which may depend on the allocation history of clients earlier in the sequence.

SRA serves as a basic online allocation problem with stochastic demand. In practice, the problem arises with particular properties depending on the context. Several policies for SRA have been proposed, see also Section 3. Among others, we mention Bassok and Ernst [1] who are one of the first to consider the SRA problem, with the objective of maximizing revenue. There are also settings where maximizing revenue or profit is not the main concern. For instance, Lien et al. [2] consider a setting where the objective is to achieve *equity* among clients, motivated by supplying regional and local food banks in Chicago in the United States. Sinclair et al. [4] propose a model to make allocations close to the optimal offline Nash social welfare solution, using a multicriteria objective approach. Sluijk et al. [5] consider a variant of the SRA problem from a vehicle routing perspective, where they present a model that finds a feasible solution that meets a fairness measure threshold. Manshadi et al. [3] study the SRA problem motivated by allocating scarce medical supplies to hospitals during the COVID-19 pandemic. They propose a policy, and show that, for the fairness measure they consider, this policy matches theoretical upper bounds on what a policy can achieve in the worst case.

It is a fact, however, that from a computational perspective, the size of instances solved has been quite limited. For example, Lien et al. [2] evaluate instances with only up to six clients, where Manshadi et al. [3] consider instances with only four clients. A salient feature of all policies proposed for SRA is that the position of a client in the sequence may have a profound impact on the amount of resource allocated to the client. Indeed, when considering a setting where all clients are identical, i.e., have the same distribution $D_i = D$, results from Lien et al. [2] imply that for several policies the amount of resource allocated to clients depends heavily on the position in which the client is served. This means that all existing policies for SRA violate *group fairness*: the property that identical clients should be treated equally. We propose a method that, by design, ensures that identical clients are allotted the same amount of resource.

The main contributions of our paper can be summarized as follows.

- Using an optimized implementation of existing methods, we provide an extensive computational study, enabling us to analyze several allocation policies for the SRA problem for instances with over 20 clients.
- We provide insight in how these policies behave and how the position of a client in the sequence may affect how much of their demand of the resource gets allocated.
- We provide a new, non-adaptive allocation policy, based on linear programming (LP), that is explainable, transparent, and easy to operationalize in the online allocation process. This policy incorporates group fairness constraints.
- We also provide an adaptive variant of this new policy that performs well on individual fairness measures.

The remainder of this article is structured as follows. In Section 2, we formally introduce the problem and state the objectives we consider in this work. In Section 3 we revisit existing allocation policies in more detail, while we also introduce our new LP-based policy. We describe our extensive computational evaluation in Section 4 where we also discuss the results. We conclude in Section 5.

2 Notation and Terminology

In this section, we further introduce some notation and define the objectives for the SRA problem. For a given client $i \in [n]$ let μ_i and σ_i^2 denote the mean and variance of the demand distribution D_i, respectively. Let $s_i = s - \sum_{j=1}^{i-1} x_j$ denote the remaining available supply upon visiting client i. An allocation is valid when $x_i \in [0, \min\{s_i, d_i\}]$, i.e., we never allocate more than the available supply or the demand of the client for the resource. When visiting client i, the policy may, in order to determine the allocation x_i, use:

- the value of the initial supply s,
- the observed realized demand $d_i \in \mathbb{R}_+$ of the visited client,
- the realized demands $d_1, \ldots, d_{i-1}$ of the previously visited clients,
- the realized allocations $x_1, \ldots, x_{i-1}$ of the previously visited clients, and
- the distributional knowledge of the demand of future clients, i.e., $D_{i+1}, \ldots, D_n$.

Given the allocation x_i and the demand d_i of client i, the *fill rate* of this client is defined as $\varphi_i = x_i/d_i$. Naturally, the fill rate of any client is in the interval $[0, 1]$. We use the notation $\varphi_i^{\min}$ to denote the minimum fill rate of the clients 1 up to i, i.e., $\varphi_i^{\min} = \min\{\varphi_1, \ldots, \varphi_i\}$. Furthermore, we also use $\varphi_1 \wedge \cdots \wedge \varphi_i$ to denote $\min\{\varphi_1, \ldots, \varphi_i\}$. We define $\varphi_0^{\min} = 1$.

We assume that the initial supply, realized demands and allocations are from a discrete set of values, as this is necessary for the computation of some existing policies in Section 3. Without loss of generality, we then assume that these values are integer.

We use the *supply scarcity* R as a measure for the ratio of available supply with the expected demand, defined as

$$R = \frac{\sum_{i=1}^{n} \mu_i}{s}. \tag{1}$$

Our motivation is based on a setting where the initial supply is unlikely to meet the total demand of all clients, i.e., when $R < 1$, which will typically occur in non-profit resource allocation.

2.1 Objectives

We mainly focus on two objectives related to *fairness* of allocations, which are called *ex-ante* and *ex-post* fairness by Manshadi et al. [3]. For the ex-post objective, the aim is to maximize the expected minimum fill rate of all clients, i.e., maximize

$$\mathbb{E}_{(d_1,\ldots,d_n)\sim(D_1,\ldots,D_n)}[\min\{\varphi_i : i \in [n]\}]. \tag{2}$$

In contrast to maximizing the expected minimum fill rate, *ex-ante* fairness maximizes the minimum expected fill rate of all clients, i.e.,

$$\min\{\mathbb{E}_{(d_1,\ldots,d_n)\sim(D_1,\ldots,D_n)}[\varphi_i] : i \in [n]\}. \tag{3}$$

When only a single allocation run is performed, only one realization of the demands is observed. Then, the ex-post objective (2) may be the most appropriate objective, as this ensures that for this single realization the minimum fill rate is maximum, in expectation. However, when allocations are repeated, then (2) may yield too conservative allocations while higher fill rates can be achieved. In this case, ex-ante fairness (3) may be more desirable to achieve, both from the perspective of the policy designer, as well as from the perspective of the client, as their expected fill rate over the repeated allocations is maximized. The existing

methods in the literature focus mainly on ex-post fairness. In the computational analysis that we perform in Section 4, we will evaluate the existing policies as well as our new policy on both the ex-post and ex-ante fairness objectives.

3 Policies

In this section, we give a detailed overview of existing allocation policies from the literature. Furthermore, we introduce our new policies in Sections 3.5 and 3.6.

3.1 Optimal Expected Minimum Fill Rate Policy

Given full knowledge of the demand distributions of the clients, a policy that maximizes the expected minimum fill rate, i.e., ex-post fairness, can be described by the following recursive relation from Lien et al. [2]. Let $Z^i(s_i, \varphi_{i-1}^{\min}, d_i)$ denote the maximum expected minimum fill rate for visiting all clients $1, \ldots, n$, given that the visited clients $1, \ldots, i-1$ have a minimum fill rate $\varphi_{i-1}^{\min}$ and supply s_i is remaining when arriving at client i with demand d_i. The optimal allocation policy can then be determined by solving the following optimality equation, for any $i \in [n-1]$:

$$Z^i(s_i, \varphi_{i-1}^{\min}, d_i) = \max_{x_i \leq s_i \wedge d_i} \mathbb{E}_{d_{i+1} \sim D_{i+1}} \left[\varphi_{i-1}^{\min} \wedge \frac{x_i}{d_i} \wedge Z^{i+1}\left(s_i - x_i, \varphi_{i-1}^{\min} \wedge \frac{x_i}{d_i}, d_{i+1} \right) \right]. \quad (4)$$

For the last node n in the sequence, we have

$$Z^n(s_n, \varphi_{n-1}^{\min}, d_n) = \varphi_{n-1}^{\min} \wedge \frac{s_n \wedge d_n}{d_n}. \quad (5)$$

Then, the optimum expected minimum fill rate over the full sequence is given by

$$F_{\text{opt}}(s) = \mathbb{E}_{d_1 \sim D_1}\left[Z^1(s, 1, d_1) \right]. \quad (6)$$

This can be solved by dynamic programming, starting at the last client in the sequence working in reverse order to the first client, filling a table Z with entries $Z^i(s_i, \varphi_{i-1}^{\min}, d_i)$ for every $i \in [n]$, $s_i \in [s]$, d_i in the support of D_i, and every possible fill rate $\varphi_{i-1}^{\min}$.

To apply this policy, the full table needs to be retained to look up the allocation for every client during the allocation process, using the current history of allocations to previous clients. In addition, it is necessary that the demand distribution as well as the allocations are discrete, and the magnitude of the demand and supply levels directly influence the time and space needed to compute the optimal policy. The amount of space needed to store the table is of order $\mathcal{O}(n \cdot s \cdot \ell^3 \cdot D_{\max})$, while the computational time is $\mathcal{O}(n \cdot s \cdot \ell^3 \cdot D_{\max}^2)$, where ℓ is (maximum) number of demand levels of a client and $D_{\max}$ is the maximum demand of any client. In many cases, these complexities (in particular the space complexit) are so high that the optimal policy can only be computed for small instances. Because of this, several alternative policies are proposed that sacrifice optimality for improved time and space complexity.

3.2 Optimal Forward Expected Minimum Fill Rate Policy

An alternative approach to designing a policy for the SRA problem is as follows. When arriving at client i, we design a policy that maximizes the minimum fill rate of the clients $i, \ldots, n$, i.e., we do not take the fill rates of the already visited clients $1, \ldots, i-1$ into account. This is also called the *forward* objective. The problem where the forward objective is maximized can

be solved optimally using a similar approach as for finding the optimal expected minimum fill rate in Section 3.1. When arriving at a client, the policy optimizing the forward objective maximizes the expected fill rate over the current and future clients, i.e., the fill rates of the already visited clients is not taken into consideration. The optimality equation is, for $i = 1, \dots, n-1$

$$\vec{Z}_f^i(s_i, d_i) = \max_{x_i \le s_i \wedge d_i} \mathbb{E}_{d_{i+1} \sim D_{i+1}} \left[\frac{x_i}{d_i} \wedge \vec{Z}_f^{i+1}(s_i - x_i, d_{i+1}) \right], \tag{7}$$

with for the last node n in the sequence

$$\vec{Z}_f^n(s_n, d_n) = \frac{s_n \wedge d_n}{d_n}. \tag{8}$$

This can also be solved by dynamic programming, where the table entries of Z_f are filled in reverse client order. Since the minimum fill rate for the previous clients is no longer a parameter, this table can be stored in $\mathcal{O}(n \cdot s \cdot \ell \cdot D_{\max})$ and filled in $\mathcal{O}(n \cdot s \cdot \ell \cdot D_{\max}^2)$ time. Compared to the policy in Section 3.1, this policy requires significantly less time and space.

3.3 Two-Node Decomposition Heuristic

The two-node decomposition (TND) heuristic as proposed in the literature by Lien et al. [2] is designed to maximize the expected minimum fill rate. At every client $i \in [n]$, the allocation is heuristically determined in two steps. First, the available supply is split in an *allotment* $\hat{s}_i$ that is available to solve the allocation problem for the current and next client in the sequence, while the remaining supply is reserved for clients further in the sequence. The allotment is proportional to the mean demand of the first two clients in the sequence relative to the total mean demand of all unvisited clients, i.e.

$$\hat{s}_i = s_i \frac{\mu_i + \mu_{i+1}}{\sum_{j=i}^{N} \mu_j}. \tag{9}$$

Second, the allotment is then used to determine an allocation for client i, based on its observed demand d_i and the distributional knowledge of the demand of client $i+1$. The allocation is equal to

$$x_i^{\mathrm{TND}} = \min\{\hat{H}_i(\hat{s}_i, d_i), \varphi_{i-1}^{\min} d_i\} \tag{10}$$

where $\hat{H}_i(\hat{s}_i, d_i)$ is an approximation of the optimal allocation in case there would only be two clients i and $i+1$ left. It is defined as

$$\hat{H}_i(\hat{s}_i, d_i) = \hat{s}_i \frac{d_i}{d_i + \tilde{m}_{i+1} + \delta_{i+1}\sqrt{\sigma_{i+1}}}, \tag{11}$$

with

$$\delta_{i+1} = \frac{\tilde{m}_i - \tilde{m}_{i+1}}{(\tilde{m}_i + \tilde{m}_{i+1})/2}, \tag{12}$$

where $\tilde{m}_i$ and σ_i denote the median and standard deviation of the demand of client i, respectively. For the last client n in the sequence, the allocation is simply $x_n = \min\{s_n, d_n\}$. For more details, we refer to the original paper of Lien et al. [2].

3.4 Projected Proportional Allocation Heuristic

Manshadi et al. [3] introduce the projected proportional allocation (PPA) heuristic. This policy is designed to maximize ex-post fairness, and is based on the following idea. Suppose that in an offline variant of the problem all demand realizations of all clients are known a-priori. Then, the optimal (offline) allocation for every client i is

$$x_i^* = \min\left\{d_i, s_i \frac{d_i}{\sum_{j\in[n]} d_j}\right\}. \tag{13}$$

In the online setting, the future demand is unknown. The PPA allocation rule substitutes the mean future demand as a heuristic policy, allocating

$$x_i^{\mathrm{PPA}} = \min\left\{d_i, s_i \frac{d_i}{d_i + \sum_{j=i+1}^{n} \mu_j}\right\}. \tag{14}$$

3.5 LP-Based Non-Adaptive Allocation Heuristic

In this work, we introduce a new, alternative, allocation policy using an LP-based approach. The objective is to maximize ex-post fairness by maximizing the expected minimum fill rate. The policy pre-assigns a single allocation value for each client, in order to keep the policy compact. This can be modeled with the following program:

$$\max_{x_1,\ldots,x_n}\left\{\sum_{d_1}\cdots\sum_{d_n} p_{d_1,\ldots,d_n}\left[\frac{x_1}{d_1}\wedge\cdots\wedge\frac{x_n}{d_n}\right] : x_1+\cdots+x_n \le s\right\}. \tag{15}$$

Here, $p_{d_1,\ldots,d_n}$ is the probability that the *scenario* occurs that the n clients realize the demands $d_1,\ldots,d_n$. Note that the objective (as a sum of minima) is a concave function. It can be modeled as a linear program, by introducing variables $f_{d_1,\ldots,d_n}$ for every "demand scenario" that corresponds to each term in the objective. This yields the LP

$$\text{maximize} \quad \sum_{d_1,\ldots,d_n} p_{d_1,\ldots,d_n} f_{d_1,\ldots,d_n} \tag{16a}$$

$$\text{subject to} \quad \sum_{i=1}^{n} x_i \le s, \tag{16b}$$

$$f_{d_1,\ldots,d_n} \le \frac{x_i}{d_i} \qquad \forall i\in[n]\ \forall d_1,\ldots,d_n, \tag{16c}$$

$$x_i \ge 0 \qquad \forall i\in[n]. \tag{16d}$$

Note that the number of variables depends largely on the number of scenarios, which can be large when the number of clients and/or the number of demand realizations of the clients is large. This is also the case for the optimal dynamic programming model. However, this model can handle continuous allocations, i.e., allocations that are not limited to a discrete set of values, and furthermore it is not sensitive to the magnitude of supply and demand values. More importantly, this policy can incorporate *group fairness*. Indeed, when clients with identical demand distributions are regarded as groups, this policy will allocate the same amount of resource to each client in a group. Thus, there is an optimal solution to this model where all clients from the same group will have an identical allocations.

An informal argument for this property is as follows. Suppose that there is an optimal solution x where two clients $i,j\in[n]$ with the same demand distribution have different allocations in an optimal solution. Without loss of generality, assume that $x_i > x_j$. Then,

we can construct another solution x' where $x'_i = x'_j = x_j$, and $x'_k = x_k$ for all $k \in [n]$, $k \neq i$ and $k \neq j$ which will have the same objective value, as the minimum fill rate will be identical for allocation x and x' in any demand realization. This allows for aggregation of these clients in the model, ensuring faster computation times.

Suppose there are m groups (or types) of clients $\theta_1, \ldots, \theta_m$ with $n_1, \ldots, n_m$ number of clients, respectively. Then the aggregated model is as follows.

$$\text{maximize} \quad \sum_{d_1,\ldots,d_m} p_{d_1,\ldots,d_m} f_{d_1,\ldots,d_m} \tag{17a}$$

$$\text{subject to} \quad \sum_{\theta=1}^{m} n_\theta x_\theta \leq s, \tag{17b}$$

$$f_{d_1,\ldots,d_m} \leq \frac{x_\theta}{d_\theta} \qquad \forall \theta \in [m]\ \forall d_1, \ldots, d_m, \tag{17c}$$

$$x_\theta \geq 0 \qquad \forall \theta \in [m]. \tag{17d}$$

This makes the model significantly smaller when the number of groups is small, achieving group fairness efficiently.

In addition, the allocation procedure is now simple: there is a single predetermined allocation for every client, which is not only easily implemented when allocations take place, but it is also explainable and easily communicated to clients.

3.6 LP-Based Adaptive Allocation Heuristic

The policy introduced in Section 3.5 can also be used as an adaptive policy by re-computing the LP for the future clients after observing the demand of the current client. Thus, after arriving at client i, the observed demand is d_i and the LP model is now adapted to only include the current client with a deterministic demand d_i and all future clients with their demand distribution, with the remaining supply s_i. The new solution yields an allocation x_i for the current client. Upon visiting the next client, the model is re-computed analogously.

The adaptive nature of this policy makes it more comparable to the other existing policies, which are all also adaptive in nature. Note that this policy also violates group fairness. This policy is expected to score higher on ex-ante individual fairness, as it can account for uncertainty in the demand realizations throughout the client sequence. Furthermore, it requires solving a linear program throughout the allocation process.

4 Computational Experiments

In order to evaluate the performance of the different policies described in Section 3 on the various objectives, we have executed a large number of computational experiments. In previous work, computational experiments were limited to considering only few clients. Lien et al. [2] consider up to six clients, where Manshadi et al. [3] have instances with four clients. While their experiments show that their heuristic policies perform well in comparison to the optimal dynamic program, it is not clear whether this is intrinsic to their policies or a consequence of the small number of clients in the instances. We therefore aim to provide a more extensive computational analysis with significantly more clients.

In order to analyze the effects on client ordering, we evaluate the policies on instances with up to 21 clients. For the optimal dynamic programming policy, this requires significant optimization in the implementation in code in order to make it computationally feasible to construct the dynamic programming table. The dynamic programing table must be filled in

reverse client order, but all entries in a "slice" of the table for a fixed client i can be computed in parallel. Together with optimizing the memory layout of the dynamic programming table, this allows for massive speedup in computation time when multiple cores or threads can be utilized. A limiting factor remains the memory requirements, as the full dynamic programming table needs to be stored to be used as a lookup table during the allocation process.

The experiments are run on a computing cluster with Intel Xeon Platinum 8260 processors at 2.40 GHz. Each processor has 24 threads and 256 GB available memory. The optimal dynamic programming policy is computed in a parallelized implementation using 24 threads, whereas all other policies are computed using a single thread. Policies requiring a linear programming solver use Gurobi version 12.0.1. Our implementation is available at `https://doi.org/10.5281/zenodo.15825047`.

4.1 Setup

We follow the general structure of generated instances used by Lien et al. [2]. We generate instances with the following parameters.

- The number of *groups* of clients of identical demand distribution is an element of $\{2, 3, 4\}$.
- The total number of clients n, which is either *small* or *large*. Small instances have a total number of clients of 14, 15, and 16 for 2, 3, and 4 groups of identical clients, respectively. Large instances have 20 clients for 2 and 4 groups, and 21 clients for 3 groups.
- The ordering of the groups of clients is specified under the Sequences column in Table 1. The symbol $\text{CV}^{\nearrow}$ ($\text{CV}^{\searrow}$, $\text{CV}^{\sim}$) refers to a setting where the coefficient of variation (CV) is uniformly increasing (decreasing, alternating) with the position of the clients in the sequence. A similar explanation applies to $\mu^{\nearrow}$, $\mu^{\searrow}$, $\mu^{\sim}$. The coefficient of variation is defined as the ratio of the standard deviation σ to the mean μ, i.e., $\text{CV} = \sigma/\mu$. For the last two rows, the mean and coefficient of variation are matched as follows: clients with the k-th largest μ have the k-th largest CV for any k, and vice versa. Each row in Table 1 therefore corresponds to 3 instances.
- The interleaving of the groups, which is either *separated* or *repeating*. For example, in an instance with 9 clients in 3 groups a, b, and c, the separated interleaving yields the sequence *aaabbbccc*, while the repeating interleaving yields *abcabcabc*.
- The initial supply scarcity $R \in \{0.5, 0.75, 1, 1.25, 1.5\}$, which is the fraction of the total expected demand $\sum_{i=1}^{n} \mu_i$ that is available as initial supply of the resource.

The total number of instances that follows from this setup equals $3 \cdot 2 \cdot 30 \cdot 2 \cdot 5 = 1800$. For each of these instances, 250 simulations are performed on which every allocation method is evaluated. The demand distributions of all clients follow a discretized gamma distribution with mean μ_i and coefficient of variation CV, of which the domains are specified in Table 1. The number of demand *levels*, i.e., the size of the support of the demand distribution, is fixed to be 21.

For the adaptive version of the LP-based method we only evaluate instances with $n = 14$ or $n = 15$ clients, due to computational limitations.

4.2 Results

As our aim is to evaluate the policies with respect to their performance on the two objectives, we choose to not report computation times, but restrict ourselves to the following general comment. As expected, the optimal dynamic programming policy is the most time-intensive

Table 1 Group ordering scenarios.

μ	CV	Sequences
50	0.5–1.5	CV$^{\nearrow}$, CV$^{\searrow}$, CV$^{\sim}$
150	0.5–1.5	CV$^{\nearrow}$, CV$^{\searrow}$, CV$^{\sim}$
50	0.75–1.25	CV$^{\nearrow}$, CV$^{\searrow}$, CV$^{\sim}$
150	0.75–1.25	CV$^{\nearrow}$, CV$^{\searrow}$, CV$^{\sim}$
50–150	0.5	$\mu^{\nearrow}$, $\mu^{\searrow}$, $\mu^{\sim}$
50–150	1.5	$\mu^{\nearrow}$, $\mu^{\searrow}$, $\mu^{\sim}$
75–125	0.5	$\mu^{\nearrow}$, $\mu^{\searrow}$, $\mu^{\sim}$
75–125	1.5	$\mu^{\nearrow}$, $\mu^{\searrow}$, $\mu^{\sim}$
50–150	0.5–1.5	CV$^{\nearrow}$, CV$^{\searrow}$, CV$^{\sim}$, large CV with large μ
50–150	0.5–1.5	CV$^{\nearrow}$, CV$^{\searrow}$, CV$^{\sim}$, large CV with small μ

policy – an instance with 21 clients takes roughly 2 hours to solve on 24 threads, which is a significant computation time. Furthermore, this policy requires more than 100 GB of memory space to store the dynamic programming table.

We now discuss the performance of the policies on the two objectives. Tables 2 and 3 summarize average scores on the ex-post and ex-ante objectives for the evaluated policies: the optimal dynamic program (Opt), the optimal forward dynamic program (Fw), the two-node decomposition policy (TND), the projected proportional allocation policy (PPA), our non-adaptive LP-based policy (Non-adaptive) and our adaptive LP-based policy (Adaptive). All instances are grouped by supply scarcity $R < 1$, $R = 1$, and $R > 1$, as well as by the number of clients n. The ex-post fairness performance of the TND heuristic and our non-adaptive policy decreases when the number of clients increases, whereas the other policies retain similar performance for different number of clients. Wile our non-adaptive policy clearly does not perform well on the ex-post fairness objective compared to the other policies, it does perform well on the ex-ante fairness objective, especially with supply scarcity $R < 1$ where it outperforms all other methods. The adaptive variant of our policy is competitive with the other allocation policies on both ex-post and ex-ante fairness.

Figure 1 shows for all instances $n = 21$ clients and supply scarcity $R = 0.5$ and $R = 1$ the distribution of the fill rate of each client in the sequence. For each policy, a box plot indicates the maximum and minimum fill rate, as well as the quantiles and the average fill rate, indicated by a green line. It can be observed that the methods vary in variance of fill rates allocated, where the optimal dynamic program has the lowest variance when supply is scarce. Our non-adaptive policy achieves a fill rate of 1 often for any client in the sequence, however with a high variance between allocation runs. It is also clearly visible that, for the optimal dynamic program, the last client in the sequence will in many cases achieve a high fill rate due to conservative allocation decisions for clients earlier in the sequence. Similar plots for instances with a different number of clients, which are omitted due to space limitations, show an identical trend.

To further evaluate the ex-ante individual fairness objective, we analyze the average fill rate of every client in the sequence and take the minimum. To compare the overall performance of the methods, we summarize the results by grouping the problem instances on number of clients and supply scarcity. Figure 2 displays the average fill rate of each client in the sequence for all instances of $n = 15$ clients for supply scarcity $R > 1$, $R = 1$, and $R < 1$, respectively. Although omitted due to space limitations, the results for $n \in \{14, 20, 21\}$ show

Table 2 Ex-post fairness.

Scarcity	n	Opt	Fw	TND	PPA	Non-adaptive	Adaptive
$R > 1$	14	0.9581	0.9434	0.8621	0.9551	0.5386	0.9504
	15	0.9611	0.9464	0.8666	0.9572	0.5199	0.9534
	16	0.9616	0.9480	0.8594	0.9591	0.5066	
	20	0.9683	0.9560	0.8339	0.9647	0.4827	
	21	0.9704	0.9585	0.8390	0.9685	0.4729	
$R = 1$	14	0.8103	0.7724	0.6963	0.7894	0.3936	0.7679
	15	0.7999	0.7638	0.6873	0.7816	0.3783	0.7498
	16	0.8045	0.7690	0.6788	0.7825	0.3696	
	20	0.8157	0.7796	0.6505	0.7889	0.3514	
	21	0.8099	0.7764	0.6439	0.7811	0.3433	
$R < 1$	14	0.5141	0.4630	0.4373	0.4747	0.2443	0.4154
	15	0.5064	0.4539	0.4272	0.4669	0.2362	0.4031
	16	0.5071	0.4537	0.4226	0.4669	0.2312	
	20	0.5117	0.4544	0.4022	0.4660	0.2180	
	21	0.5077	0.4477	0.3992	0.4609	0.2144	

Table 3 Ex-ante fairness.

Scarcity	n	Opt	Fw	TND	PPA	Non-adaptive	Adaptive
$R > 1$	14	0.9691	0.9812	0.8631	0.9797	0.9214	0.9769
	15	0.9735	0.9818	0.8680	0.9798	0.9195	0.9780
	16	0.9723	0.9827	0.8603	0.9812	0.9151	
	20	0.9764	0.9871	0.8343	0.9847	0.9212	
	21	0.9792	0.9874	0.8393	0.9858	0.9195	
$R = 1$	14	0.8490	0.9143	0.7002	0.9116	0.8652	0.8862
	15	0.8444	0.9082	0.6921	0.9067	0.8591	0.8803
	16	0.8441	0.9137	0.6810	0.9100	0.8564	
	20	0.8518	0.9221	0.6514	0.9149	0.8629	
	21	0.8488	0.9204	0.6449	0.9132	0.8612	
$R < 1$	14	0.5694	0.6776	0.4412	0.6370	0.7354	0.6340
	15	0.5607	0.6655	0.4316	0.6352	0.7308	0.6494
	16	0.5589	0.6657	0.4251	0.6351	0.7277	
	20	0.5622	0.6669	0.4031	0.6346	0.7289	
	21	0.5576	0.6615	0.4002	0.6342	0.7263	

an almost identical trend. In situations where supply is not scarce, i.e., when $R \geq 1$, the non-adaptive policy performs significantly worse on ex-ante individual fairness. However, when $R < 1$, which is often the case for applications for this problem, our method significantly outperforms existing methods on ex-ante individual fairness. Moreover, it can be seen that for all adaptive policies that the average fill rate of a client heavily depends on the place of the client in the sequence, whereas our non-adaptive policy achieves a much more equal fill rate for every client in the sequence. From a fairness perspective, the sequencing of customers should not impact their fill rate and our policy may be beneficial in some applications.

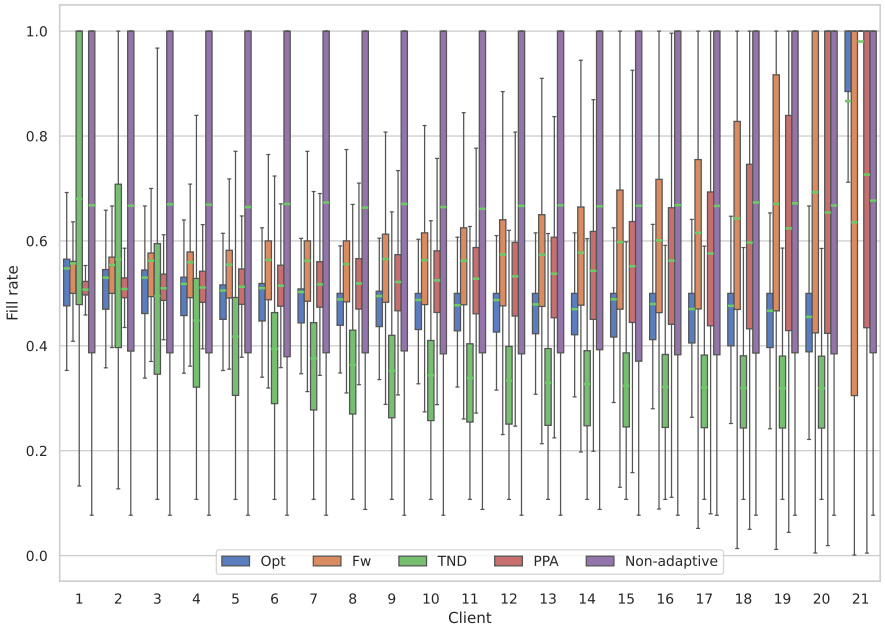
Let us now discuss to what extent the order of the clients impacts the allocation. For the ex-post objective, when the supply is scarce ($R < 1$), all policies perform best when the clients are ordered by decreasing coefficient of variation. This confirms the results by Lien et al. [2], and our computational analysis shows that this also extends to instances with significantly more clients. The TND heuristic, however, performs best on instances where the coefficient of variation is constant and small and where the mean demand of the clients decreases in the order of the sequence. When $R \geq 1$, all policies perform best when the coefficient of variation is constant and small, as in this case a fill rate close to 1 can be achieved for many clients. For the ex-ante objective when $R < 1$, the forward optimal dynamic program and our policies perform best when the coefficient of variation is high for all clients. The PPA heuristic and the optimal dynamic program perform best when the mean demand is decreasing and the coefficient of variation is high. The TND heuristic performs best on the ex-ante objective when the mean demand is decreasing and the coefficient of variation is low for all clients.

5 Conclusion

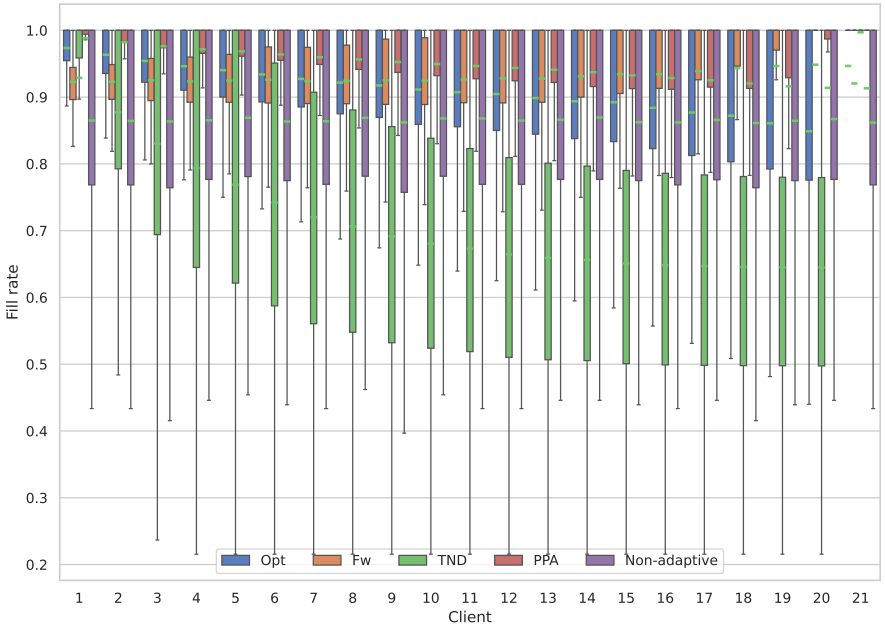
In this work, we have conducted an extensive computational study on several allocation policies for the sequential resource allocation problem. While several objectives can be used to design a policy, we focus on maximizing fairness in several ways. The main focus is on maximizing expected fill rates of the clients, which can be done in an ex-ante or ex-post fashion. We add results on computational experiments to the existing body of literature on sequential resource allocation, by evaluating significantly larger problem instances with up to 21 clients. From these results, we confirm that existing heuristic methods perform well on the ex-post and ex-ante fairness objective, also for large instances.

We additionally introduce a new allocation method based on a linear program, applied in a non-adaptive and adaptive fashion. The non-adaptive policy can be a desirable property in certain applications where dynamic allocation decisions are either not possible or not desired. Furthermore, the non-adaptive policy is explainable and easily operationalized as it retains a single allocation value for every client. The policy also allows the policy designer to also include group fairness as an additional constraint on the allocations of the resource; our method ensures that clients with the same demand distribution are assigned equal allocations. From computational evaluation, it can be seen that this new allocation policy performs worse on the ex-post fairness objective, whereas it outperforms existing methods on ex-ante fairness when the supply of the resource is scarce. Furthermore, the average fill rate is much more equal for all clients in the sequence compared to any of the methods, where effects of position in the sequence are clearly noticeable.

For further work, it would be possible to explore whether our new method can be adapted to further explore the trade-off between ex-post and ex-ante fairness. For example, pre-computing multiple possible allocation values for every client that can be used based on the realized demand during the allocation process possibly increases ex-post fairness, which makes the policy in between the non-adaptive and adaptive variant introduced in this work.

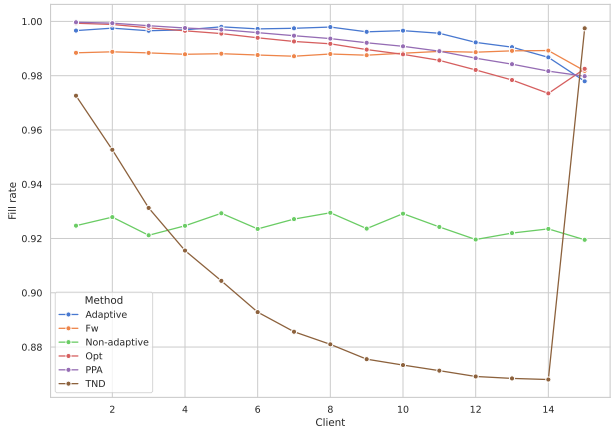


(a) Fill rate distribution of clients for all instances with $R = 0.5$ and $n = 21$.

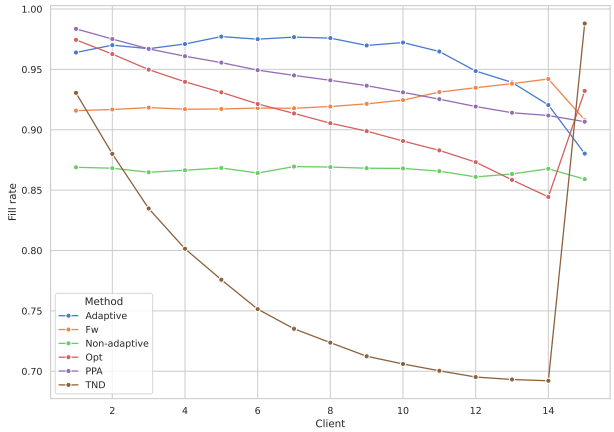


(b) Fill rate distribution of clients for all instances with $R = 1.0$ and $n = 21$.

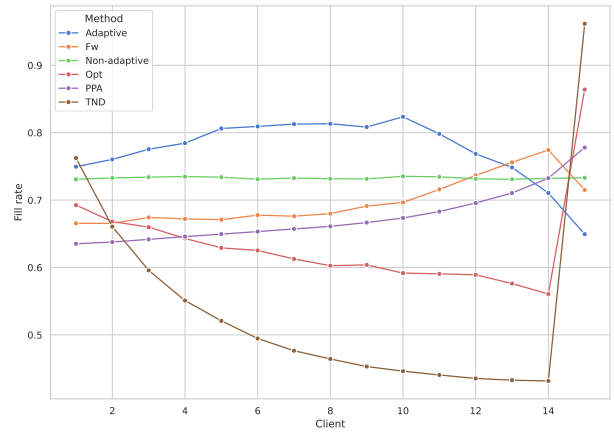
Figure 1 Distribution of fill rate of individual clients for the large size instances.



(a) Average fill rate when $R > 1$.



(b) Average fill rate when $R = 1$.



(c) Average fill rate when $R < 1$.

Figure 2 Average fill rate per client in the sequence of all tested instances with $n = 15$, split on supply scarcity.

References

1 Yehuda Bassok and Ricardo Ernst. Dynamic allocations for multi-product distribution. *Transportation Science*, 29(3):256–266, 1995. doi:10.1287/trsc.29.3.256.

2 Robert W. Lien, Seyed M. R. Iravani, and Karen R. Smilowitz. Sequential resource allocation for nonprofit operations. *Operations Research*, 62(2):301–317, 2014. doi:10.1287/opre.2013.1244.

3 Vahideh Manshadi, Rad Niazadeh, and Scott Rodilitz. Fair dynamic rationing. *Management Science*, 69(11):6818–6836, 2023. doi:10.1287/mnsc.2023.4700.

4 Sean R. Sinclair, Gauri Jain, Siddhartha Banerjee, and Christina Lee Yu. Sequential fair allocation of limited resources under stochastic demands. *Preprint*, 2022. arXiv:2011.14382.

5 N Sluijk, W Rei, J Kinable, M Gendreau, and T Van Woensel. Fair stochastic vehicle routing with partial deliveries. *Preprint*, 2023. URL: https://optimization-online.org/?p=22778.