Parameterized Algorithms for Optimal Refugee Resettlement

Jiehua Chen[a,1,*], **Ildikó Schlotter**[b,c,1] **and Sofia Simola**[a,1]

[a]TU Wien, Austria
[b]HUN-REN Centre for Economic and Regional Studies, Hungary
[c]Budapest University of Technology and Economics, Hungary
ORCID (Jiehua Chen): https://orcid.org/0000-0002-8163-1327, ORCID (Ildikó Schlotter): https://orcid.org/0000-0002-0114-8280, ORCID (Sofia Simola): https://orcid.org/0000-0001-7941-0018

Abstract. We study variants of the Optimal Refugee Resettlement problem where a set F of refugee families need to be allocated to a set P of possible places of resettlement in a feasible and optimal way. Feasibility issues emerge from the assumption that each family requires certain services (such as accommodation, school seats, or medical assistance), while there is an upper and, possibly, a lower quota on the number of service units provided at a given place. Besides studying the problem of finding a feasible assignment, we also investigate two natural optimization variants. In the first one, we allow families to express preferences over P, and we aim for a Pareto-optimal assignment. In a more general setting, families can attribute utilities to each place in P, and the task is to find a feasible assignment with maximum total utilities. We study the computational complexity of all three variants in a multivariate fashion using the framework of parameterized complexity. We provide fixed-parameter tractable algorithms for a handful of natural parameterizations, and complement these tractable cases with tight intractability results.

1 Introduction

At the 2023 Global Refugee Forum, the UN High Commissioner for Refugees reported that *114 million* people are currently displaced due to persecution, human rights violations, violence, and wars, and made a direct appeal to everyone to join forces to help refugees find protection.[2] This immense number highlights the critical need for effective resettlement strategies that cater to diverse populations.

Refugee resettlement involves not just relocating individuals but also families, each with distinct needs and service requirements ranging from accommodation to education and medical assistance. Delacrétaz et al. [15] and Ahani et al. [2] propose a *multi-dimensional* and *multiple knapsack* model to address these challenges. Their model takes into account the specific needs of *refugee families* who require a range of services, as well as the capacity constraints of potential hosting places that have specific upper and lower quotas on the services they can offer. The goal is to determine a *feasible* assignment from the families to the places which satisfies the specific needs of the families while ensuring that no place is over- or under-subscribed according to its capacity constraints. Additionally, the model may include a *utility score* for each family–place pair which estimates the "profit" that a family may contribute to a place; such profit could be for example the employment outcome. Optimizing the assignment means finding a feasible assignment that yields a maximum total utility.

If we care about the welfare and choices of the refugee families, we may allow them to express *preferences* over places which they find acceptable [15]. A standard optimality criterion in such a case is Pareto-optimality, which means that we aim for a feasible assignment for which no other feasible assignment can make one family better off without making another worse off.

Unfortunately, it is computationally intractable (i.e., NP-hard) to determine whether a feasible assignment exists [8]. Similarly, it is NP-hard to find a feasible assignment with maximum total utility or one that is Pareto-optimal, even if there are no lower quotas [7, 19, 2]. To tackle these complexities, we examine the parameterized complexity of the three computational problems for refugee resettlement that we study, FEASIBLE-RR, MAXUTIL-RR and PARETO-RR, and provide parameterized algorithms for them. We focus on canonical parameters such as the number of places (m), the number of refugee families (n), the number of services (t), and the desired utility (u^*). We also consider additional parameters that are motivated from real-life scenarios, including the maximum number $r_{\max}$ of units required by a family per service and the maximum utility $u_{\max}$ a family can contribute. The service units can reasonably be assumed to be small integers in practical situations when a family's requirements describe their need for housing (e.g., number of beds or bedrooms) or education (e.g., the number of school seats or kindergarten places). Our study provides new insights into the parameterized complexities of these problems, presenting fixed-parameter (FPT) algorithms for several natural parameterizations, and contrasting these with strong intractability results. See Table 1 for an overview. We summarize our main contributions as follows.

Single service. We develop an FPT algorithm w.r.t. $r_{\max}$ for FEASIBLE-RR; the algorithm also applies to MAXUTIL-RR and PARETO-RR when all families have the same utilities for all places (*equal utilities*) or are indifferent between all of them (*equal preferences*), respectively; see Theorem 1. The main idea is to group all families together that have the same requirements, and group all places together

[*] Corresponding Author. Email: jiehua.chen@tuwien.ac.at.
[1] Equal contribution.
[2] https://www.unhcr.org/global-refugee-forum-2023

with the same lower and upper quotas. Then, we observe that either the upper quotas are small (i.e., bounded by a function in $r_{\max}$) so we can brute-force search all possible partitions of the families into different places, or there is a so-called *homogeneous ρ-block* (see Section 3 for the formal definition) that can be exchanged across the places, which enables us to replace the upper quota of each place with a value bounded by a function of $r_{\max}$. In this way, we bound the number of groups of families and places, and can use integer linear programming (ILP) to obtain an FPT algorithm for $r_{\max}$.

We also propose an FPT algorithm for the combined parameter $m + r_{\max}$ for the general case when families may have different utilities or preferences (Theorem 3). The generalized algorithm additionally uses the idea that *homogeneous ρ-blocks* can be exchanged across places, and combines dynamic programming with color-coding [4] to find an optimal solution in FPT time.

Multiple services. In Theorem 6, we extend the FPT algorithm of Theorem 1 for the setting of equal preferences or utilities to multiple service types by combining the parameters $r_{\max}$ and t, the number of services; we use the technique of N-fold integer programming [21]. We present a more general FPT algorithm for PARETO-RR with parameter $r_{\max}+t+m$ which also solves MAXUTIL-RR if the number of different utility values is bounded (Theorem 7); this result relies on Lenstra's result on solving ILPs with bounded dimension [27]. Contrasting our algorithmic results, we prove that PARETO- and MAXUTIL-RR are both NP-hard already for three places, even if there are no lower quotas, all upper quotas are 1, and families have equal preferences or utilities, respectively; see Theorem 5.

Related work. The model we study is the same as that of Ahani et al. [2]. They formulate MAXUTIL-RR via Integer Linear Programming (ILP) and study its performance. The same model without lower quotas has attracted previous study: It was introduced in a working paper by Delacrétaz et al. [14] (see also [15]). The paper provides an algorithm for finding a Pareto-efficient matching when the preferences are strict, and also studies other stability concepts. Aziz et al. [7] show that *finding* a Pareto-optimal assignment is NP-hard even when the families are indifferent between places, and study a few other stability concepts. Nguyen et al. [30] use fractional matchings to find group-stable assignments which violate the quotas only a little. None of the works above focuses on the parameterized complexity of the problems.

As already mentioned by Ahani et al. [2], the MAXUTIL-RR problem is a generalization of the MULTIPLE/MULTIDIMENSIONAL KNAPSACK problem [2]. The parameterized complexity of the latter has been studied by Gurski et al. [19], and several of our hardness-results are obtained either directly from them or from modifications of their reductions. MULTIPLE/MULTIDIMENSIONAL KNAPSACK however has neither lower quotas nor different profits for items depending on which knapsack they are placed in. They also assume the sizes and profits are encoded in binary, whereas we assume they are encoded in unary. Hence, their parameterized algorithms are not directly applicable to our problems. FEASIBLE-RR generalizes BIN PACKING [22] and hence SIMPLE MULTIDIMENSIONAL PARTITIONED SUBSET SUM [17]; note that the latter two problems are equivalent. Since BIN PACKING is W[1]-hard w.r.t. the number of bins and the bins correspond to the places in our setting, W[1]-hardness for FEASIBLE-RR follows; see Proposition 1.

The problem can be seen as an extension of different classical matching problems. We can model MATCHING WITH DIVERSITY CONSTRAINTS [12, 20, 13, 8, 1, 26] by using services as types. In the case where we have a single service, the problem can be seen as a variant of MATCHING WITH SIZES [11, 29], where the service requirements correspond to the sizes.

Refugee resettlement has also been studied in the literature under other types of models: Online setting [6, 3, 9], one-to-one housing [5], preferences based on weighted vectors [33], hedonic games [25], and placing refugees on a graph [23, 28, 32].

Paper structure. In Section 2, we formally define RR. We investigate the case when there is only one service and when there are multiple services in Section 3 and Section 4, respectively. In Section 4, we first look at the FEASIBLE-RR problem, followed by PARETO-RR the problem, and finally the MAXUTIL-RR problem. We conclude with a discussion on potential areas for future research in Section 5.

2 Preliminaries

For an integer z, we use $[z]$ to denote the set $\{1, 2, \ldots, z\}$. Given two vectors $\boldsymbol{x}$ and $\boldsymbol{y}$ of same length, we write $\boldsymbol{x} \le \boldsymbol{y}$ if for each coordinate i it holds that $\boldsymbol{x}[i] \le \boldsymbol{y}[j]$.

An *instance* of RR is a tuple $(F, P, S, (\boldsymbol{r}_i)_{f_i \in F}, (\underline{\boldsymbol{c}}_j, \bar{\boldsymbol{c}}_j)_{p_j \in P})$ with the following information.

- F denotes a set of n refugee *families* with $F = \{f_1, \ldots, f_n\}$,
- P denotes a set of m *places* with $P = \{p_1, \ldots, p_m\}$, and
- S denotes a set of t *services* $S = \{s_1, \ldots, s_t\}$, such that
- each family $f_i \in F$ has a *requirement* vector $\boldsymbol{r}_i \in \mathbb{N}^t$ where, for every $s_k \in S$, the value $\boldsymbol{r}_i[k]$ determines how many units of service s_k the family f_i requires, and
- each place $p_j \in P$ has two vectors $\underline{\boldsymbol{c}}_j, \bar{\boldsymbol{c}}_j \in \mathbb{N}^t$, denoted as *lower quota* and *upper quota* which indicate for every service $s_k \in S$, the minimum and maximum number of units place p_j can provide. Non-zero lower quotas for places may for example follow from an obligation for a place to house at least a certain number of refugees. If the lower quota of every place is a zero-vector, then we say that the instance has *no lower quotas*.

Assignments. Given an instance of REFUGEE RESETTLEMENT, an *assignment* is a function $\sigma\colon F \to P \cup \{\perp\}$; we say that $f_i \in F$ is *assigned* to a place $p_j \in P$ if $\sigma(f_i) = p_j$, and f_i is *unassigned* if $\sigma(f_i) = \perp$. We define the *load* vector of a place $p_j \in P$ under σ as $\mathrm{load}(p_j, \sigma) \coloneqq \sum_{f_i \in \sigma^{-1}(\ell_j)} \boldsymbol{r}_i[k]$; for each service $s_k \in S$, $\mathrm{load}(p_j, \sigma)[k]$ denotes the number of units that are required by the refugees that are assigned to p_j. An assignment is *complete* if it does not leave any families unassigned. An assignment is *feasible* if for every place $p_j \in P$ the load vector is within the lower and upper quota, i.e., $\underline{\boldsymbol{c}}_j \le \mathrm{load}(p_j, \sigma) \le \bar{\boldsymbol{c}}_j$. Place p_j can *accommodate* a set of families $F' \subseteq F$ if $\sum_{f_{i'} \in F'} \boldsymbol{r}_{i'}[k] \le \bar{\boldsymbol{c}}_j[k]$ for each $s_k \in S$.

Utilities. Each family may contribute a certain utility to each place. To model this, each family $f_i \in F$ expresses an integral *utility* vector $\boldsymbol{u}_i \in \mathbb{Z}^m$, where for every $p_j \in P$, the value $\boldsymbol{u}_i[j]$ indicates the utility of family f_i if assigned to p_j. Note that we also allow negative utilities, but it will be evident that all hardness results hold even if the utilities are non-negative. Given an assignment σ, we define the *(total) utility* of the assignment as the sum of all utilities obtained by the families, i.e., $\mathrm{util}(\sigma) = \sum_{p_j \in P} \sum_{f_i \in \sigma^{-1}(p_j)} \boldsymbol{u}_i[j]$. We consider two special kinds of utility vectors. We say that the families have *equal utilities* if all utility values $\boldsymbol{u}_i[j]$ are equal and positive over all families $f_i \in F$ and places $p_j \in P$, and families have *binary* utilities if each utility value is either zero or one.

Preferences and Pareto-optimal assignments. Each family $f_i \in F$ may only find a subset of places *acceptable* and may have a *preference list* $\succeq_i$ over the acceptable places, i.e., a weak order over a subset of P. For a family f_i and two places p and p' in its preference list, $p \succeq_i p'$ means that f_i *weakly prefers* p to p'. If $p \succeq_i p'$ and $p' \succeq_i p$, then we write $p \sim_i p'$ and say that f_i is *indifferent* between p and p'. We write $p \succ_i p'$ to denote that f_i *(strictly) prefers* p to p', meaning that $p \succeq_i p'$ but $p \not\preceq_i p'$. If the preference list of f_i contains p, then f_i finds p *acceptable*. We assume that each family f_i prefers being assigned to some place in his preference list over being unassigned; accordingly, we write $p \succ_i \perp$. An assignment is *acceptable* if every family is either unassigned or assigned to a place it finds acceptable.

We also define *equal* and *dichotomous* preferences: If every family finds every place acceptable and is additionally indifferent between them, the preferences are equal. If every family is indifferent between every place it finds acceptable, the preferences are dichotomous.

A feasible and acceptable assignment σ is *Pareto-optimal* if it admits no *Pareto-improvement*, that is, a feasible and acceptable assignment σ' such that $\sigma'(f_i) \succeq_i \sigma(f_i)$ for every $f_i \in F$ and there exists at least one family $f_{i'} \in F$ such that $\sigma'(f_{i'}) \succ_{i'} \sigma(f_{i'})$.

Central problems. We are now ready to define our problems.

FEASIBLE-RR
Input: An instance I of REFUGEE RESETTLEMENT.
Question: Is there a feasible assignment σ for I?

MAXUTIL-RR
Input: An instance I of refugee resettlement, a utility vector $\boldsymbol{u}_i \in \mathbb{Z}^m$ for each family $f_i \in F$, and an integer bound u^*.
Question: Is there a feasible assignment σ for I such that $\text{util}(\sigma) \geq u^*$?

PARETO-RR
Input: An instance I of refugee resettlement and a preference order $\succeq_i$ for every $f_i \in F$.
Task: *Find* a feasible and acceptable Pareto-optimal assignment σ for I or report that none exists.

We remark that there is a straightforward way to reduce PARETO-RR to the optimization variant of MAXUTIL-RR in the following sense. Suppose that an algorithm $\mathcal{A}$ finds a maximum-utility feasible assignment for each instance of MAXUTIL-RR that admits a feasible assignment. Such an algorithm can be used to solve an instance I the PARETO-RR problem as follows.

Observation 1 (★). *Given an instance I of* PARETO-RR, *construct an instance I' of* MAXUTIL-RR *as follows. For each family $f_i \in F$:*
- *for every place p_j that f_i finds acceptable, set $\boldsymbol{u}_i[j] = |\{p_{j'} \in P \mid p_j \succeq_i p_{j'}\}|$;*
- *for each place p_j that f_i finds unacceptable, set $\boldsymbol{u}_i[j] = -m \cdot n$.*

Let σ be a maximum-utility feasible assignment for I'. If $\text{util}(\sigma) > 0$, then σ is a feasible, acceptable, and Pareto-optimal assignment for I; otherwise there is no feasible and acceptable assignment for I.

Parameterization. We study the following parameters:
- number of places ($m = |P|$),
- number of refugee families ($n = |F|$),
- number of services ($t = |S|$),
- maximum number of units required for all services and by all families ($r_{\max} = \max\{\boldsymbol{r}_i[k] : f_i \in F, s_k \in S\}$).

We also study the following parameters for MAXUTIL-RR: the total utility bound u^* and the maximum utility brought by a family $u_{\max} = \max\{\boldsymbol{u}_i[j] : f_i \in F, p_j \in P\}$.

Additionally, we consider the maximum length of the ties in preference lists. However, this parameter is upper-bounded by m, and most problems are already hard w.r.t. m. We also consider the highest upper quota any place has for a service $c_{\max} = \max_{p_j \in P, s_k \in S} \bar{\boldsymbol{c}}_j[k]$, but discover that this parameter behaves very similarly to the smaller and better-motivated parameter $r_{\max}$. Note that we may assume that

Parameter	FEASIBLE	MAXUTIL		PARETO	
		LQ=0 / LQ≠0		LQ=0 / LQ≠0	
m	**W1h** [P1]	**W1h**$^\circ$/**W1h**$^\circ$	[P1]	**W1h**$^=$/**W1h**$^=$	[P1]
	XP [P7]	**XP** / **XP**	[P7]	**XP** / **XP**	[P7]
$r_{\max}$	**FPT** [T1]	**NPh** / **NPh**	[T2]	**NPh** / **NPh**	[T2]
eq. util./pref.	– –	**FPT** / **FPT**	[T1]	**FPT** / **FPT**	[T1]
$m + r_{\max}$	**FPT** [T3]	**FPT** / **FPT**	[T3]	**FPT** / **FPT**	[T3]
$u_{\max}$	– –	NPh$^\circ$/NPh$^\circ$	[19]	–	–
u^*	– –	**FPT**/**NPh**$^\circ$	[T4]/[P1]	–	–
$m + r_{\max}$	**NPh** [P2]	NPh$^\circ$/NPh$^\circ$	[19]	**NPh**$^=$/**NPh**$^=$	[T5]/[P2]
t	**NPh** [P1]	NPh$^\circ$/NPh$^\circ$	[19]	NPh$^=$/NPh$^=$	[7]
n	**FPT** [P6]	**FPT** / **FPT**	[P6]	**FPT** / **FPT**	[P6]
$m + t$	**W1h** [P1]	**W1h**$^\circ$/**W1h**$^\circ$	[P1]	**W1h**$^=$/**W1h**$^=$	[P1]
	XP [P7]	**XP** / **XP**	[P7]	**XP** / **XP**	[P7]
$t + r_{\max}$	**FPT** [T6]	**NPh** / **NPh**	[T2]	**NPh** / **NPh**	[T2]
eq. util./pref.	– –	**FPT**/**FPT**	[T6]	**FPT** / **FPT**	[T6]
$m + t + r_{\max}$	**FPT** [T6]	**XP** / **XP**,**?**	[P7]	**FPT** / **FPT**	[T7]
binary util.	– –	**FPT**	[T7]	–	–
u^*	– –	W1h$^\circ$/**NPh**$^\circ$	[19]/[P2]	–	–
	– –	**XP**/**NPh**$^\circ$	[P8]/[P2]	–	–

Table 1. All three problems are NP-hard in general; see [8], [19, T32],[7, P7.1]. Above: Results for the single-service case ($t = 1$). We skip the parameterization by n since for this case since it is FPT for the more general case. Below: Results for the general case. Here, we skip the parameterization by $u_{\max}$ since it is already NP-hard for the single-service case. Bold faced results are obtained in this paper. LQ=0 (resp. LQ≠0) refers to the case when lower quotas are zero (resp. may be positive). NPh means that the problem remains NP-hard even if the corresponding parameter is constant. All hardness results hold for dichotomous preferences or binary utilities. Additionally, $^\circ$ (resp. $^=$) means hardness results hold even for equal utilities (resp. preferences). The results for the remaining parameter combinations are deferred to Appendix D, Table 2.

for each family there is at least one place that can accommodate it, otherwise we can remove the family from our instance; this implies that we can assume $c_{\max} \geq r_{\max}$.

We also obtain FPT results w.r.t. the sum of the capacities of the places, that is, $c_\Sigma = \sum_{p_j \in P, s_k \in S} \bar{\boldsymbol{c}}_j[k]$, and the sum of the requirements of the families $r_\Sigma = \sum_{f_i \in F, s_k \in S} \boldsymbol{r}_i[k]$. If there are no lower quotas, we also have an FPT result w.r.t. the sum of utilities $u_\Sigma = \sum_{f_i \in F, p_j \in S} \boldsymbol{u}_i[j]$. If the instance has non-zero lower quotas, then the problem is hard even when all utilities are zero, and this parameter is not helpful. We also study the complexity w.r.t. the number of agents who have ties in their preference lists $n_\sim$. Discussion on parameters c_Σ, r_Σ, u_Σ, and $n_\sim$ are deferred to Appendix D.

3 Single service

Let us assume that there is only a single service in our input instance. Thus, we will simply refer to $\boldsymbol{r}_i[1]$ as the *requirement* of a family $f_i \in F$, and we will write $\boldsymbol{r}_i = \boldsymbol{r}_i[1]$ accordingly. Observe that we may assume w.l.o.g. that each family has a positive requirement. Similarly, we will refer to $\bar{\boldsymbol{c}}_j[1]$ and $\underline{\boldsymbol{c}}_j[1]$ as the *upper* and the *lower quota* of a place $p_j \in P$, writing also $\bar{\boldsymbol{c}}_j = \bar{\boldsymbol{c}}_j[1]$ and $\underline{\boldsymbol{c}}_j = \underline{\boldsymbol{c}}_j[1]$.

The reader may observe that when our sole concern is feasibility, then the problem can be seen as a multidimensional variant of the classic BIN PACKING or KNAPSACK problems. On one hand, it is not hard to show that the parameterized hardness of BIN PACKING w.r.t. the number of bins as parameter translates to parameterized hardness of FEASIBLE-RR w.r.t. the number of places; see Proposition 1. On the other hand, the textbook dynamic programming technique for KNAPSACK was used by Gurski et al. [19, Proposition 34] to solve the so-called MAX MULTIPLE KNAPSACK problem which in our model coincides with the MAXUTIL-RR problem without lower quotas. This approach can be adapted in a straightforward way to solve the MAXUTIL-RR problem even for the case when there are multiple services and lower quotas; in Proposition 7 we present an algorithm running in $O((c_{\max})^{mt} nm)$ time.

Proposition 1 (⋆)**.** *The following problems are W[1]-hard w.r.t. m for $t = 1$:*
- FEASIBLE-RR*;*
- PARETO-RR *with no lower quotas and equal preferences;*
- PARETO-RR *when all families have strict preferences;*
- MAXUTIL-RR *with no lower quotas and equal utilities;*
- MAXUTIL-RR *with $u^* = 0$.*

In spite of the strong connection between FEASIBLE-RR and BIN PACKING (or between MAXUTIL-RR and KNAPSACK), the context of REFUGEE RESETTLEMENT motivates parameterizations that have not been studied for these two classical problems. One such parameter is $r_{\max}$, the maximum units of a service that any refugee family may require. Theorem 1 presents an efficient algorithm for FEASIBLE-RR for the case when $r_{\max}$ is small; the proposed algorithm can be used to solve PARETO- and MAXUTIL-RR as well, assuming equal preferences or utilities, when the task is to assign as many refugee families as possible.

Let us introduce an important notion used in our algorithms. Let ρ denote the least common multiple of all integers in the set $\{1, \ldots, r_{\max}\}$; then $\rho \le (r_{\max})!$ is clear. We say that a set $F' \subseteq F$ of families is a *homogeneous ρ-block*, if all families in F' have the same requirement, and their total requirement is exactly ρ.

Observation 2 (⋆)**.** *Suppose that the number of services is $t = 1$. If $F' \subseteq F$ is a set of families such that $\sum_{f_i \in F'} \boldsymbol{r}_i > r_{\max}(\rho - 1)$, then F' contains a homogeneous ρ-block.*

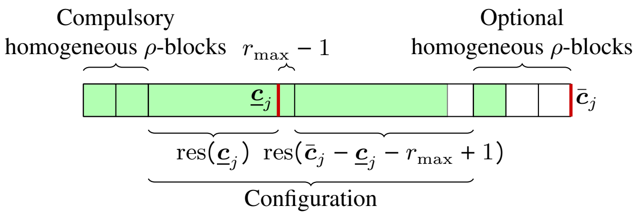
In the case where $r_{\max}$ is a constant and the families are indifferent between the places, we can use Observation 2 to bound the number of different possible places. The idea is to observe that while the location capacities are unbounded, we can bound the "relevant part" of them by a function of $r_{\max}$. If the total requirement of families assigned to a place is more than $r_{\max}(\rho - 1)$, there must be a homogeneous ρ-block among them. We can treat these homogeneous ρ-blocks separately from the places they originate from, and thus bound the upper quotas by a function of $r_{\max}$. The family requirements are also trivially bounded by $r_{\max}$.

Since we have now bounded both the maximum requirements of the families and the capacities of the locations, we can enumerate all the different ways families may be matched to places. We can create an ILP that has a variable for each such way and additionally variables for the homogeneous ρ-blocks. As the number of variables and constraints is bounded above by a function of $r_{\max}$, we can solve this ILP in FPT time w.r.t. $r_{\max}$ [27].

Theorem 1. PARETO- *and* MAXUTIL-RR *are FPT w.r.t. $r_{\max}$ when $t = 1$ and families have equal preferences or utilities, respectively.*

Proof. We start by showing that we can find an assignment that matches the maximum number of families in FPT time w.r.t. $r_{\max}$. Under equal preferences (resp. utilities) this assignment must maximize utility (resp. be Pareto-optimal).

We start by defining two functions which will be useful for typing places by their service quotas: the function $\arg \mathrm{res} \colon \mathbb{N} \to \mathbb{N}$ and the



Compulsory homogeneous ρ-blocks
$r_{\max} - 1$
Optional homogeneous ρ-blocks
$\underline{\boldsymbol{c}}_j$
$\bar{\boldsymbol{c}}_j$
$\mathrm{res}(\underline{\boldsymbol{c}}_j)$
$\mathrm{res}(\bar{\boldsymbol{c}}_j - \underline{\boldsymbol{c}}_j - r_{\max} + 1)$
Configuration

Figure 1. Illustration for the proof of Theorem 1. The bar shows the upper and lower quotas of a place, and the green area represents the requirements of the families matched to it.

function $\mathrm{res} \colon \mathbb{N} \to \{0, \ldots, r_{\max}(\rho - 1)\}$:

$$\mathrm{res}(x) = \max_{\alpha \in \mathbb{N}} \{x - \alpha \cdot \rho : x - \alpha \cdot \rho \le r_{\max}(\rho - 1)\}$$
$$\arg \mathrm{res}(x) = \arg \max_{\alpha \in \mathbb{N}} \{x - \alpha \cdot \rho : x - \alpha \cdot \rho \le r_{\max}(\rho - 1)\}.$$

Observe that $\quad \forall x \in \mathbb{N}, x = \rho \cdot \arg \mathrm{res}(x) + \mathrm{res}(x). \qquad (1)$

Using the res function we can associate each place with a type τ^P. Let $\tau^P(p_j) = (\mathrm{res}(\underline{\boldsymbol{c}}_j), x)$, where

$$x = \begin{cases} \mathrm{res}(\bar{\boldsymbol{c}}_j - \underline{\boldsymbol{c}}_j - r_{\max} + 1) + r_{\max} - 1, & \text{if } \bar{\boldsymbol{c}}_j - \underline{\boldsymbol{c}}_j \ge r_{\max} - 1, \\ \bar{\boldsymbol{c}}_j - \underline{\boldsymbol{c}}_j, & \text{otherwise.} \end{cases}$$

The first element of the type tells us the lower quota of the place after we have discounted all the homogeneous ρ-blocks that are used to satisfy the lower quota. The second element tells us the size of the "optional" quota $\bar{\boldsymbol{c}}_j - \underline{\boldsymbol{c}}_j$ when we have again discounted all the homogeneous ρ-blocks that this part may contain. We however only compute the residue on the part of $\bar{\boldsymbol{c}}_j - \underline{\boldsymbol{c}}_j$ that is larger than $r_{\max} - 1$. This is because the total requirements of the families that are used to satisfy $\underline{\boldsymbol{c}}_j$ may be slightly greater than $\underline{\boldsymbol{c}}_j$, and thus there may be a family whose requirement is partially counted for $\bar{\boldsymbol{c}}_j - \underline{\boldsymbol{c}}_j$, however it may have been used for a homogeneous ρ-block. See Figure 1 for intuition of how the lower and upper quotas of a place are divided into the configuration and homogeneous ρ-blocks.

Let $\mathcal{T}^P = \{0, \ldots, r_{\max}(\rho - 1)\} \times \{0, \ldots, r_{\max}(\rho - 1) + r_{\max} - 1\}$ be the set of possible place types. It is clear that their number is bounded above by $r_{\max}^2 \rho^2$, which is a function of $r_{\max}$. We additionally know that every place must have $\arg \mathrm{res}(\underline{\boldsymbol{c}}_j)$ many homogeneous ρ-blocks assigned to it. We call these *compulsory* homogeneous ρ-blocks. Similarly, we may assign at most $\arg \mathrm{res}(\bar{\boldsymbol{c}}_j - \underline{\boldsymbol{c}}_j - r_{\max} + 1)$ additional homogeneous ρ-blocks to p_j, which we call *optional* homogeneous ρ-blocks. Because the families are indifferent between the places, we do not need to keep track of the place the compulsory and optional homogeneous ρ-blocks belong to, and we only enforce that the families assigned to compulsory homogeneous ρ-blocks create exactly the number of compulsory homogeneous ρ-blocks needed, and the families assigned to optional homogeneous ρ-blocks create at most the number of optional homogeneous ρ-blocks.

Now with a bound on the upper and lower quotas of the places discounting the homogeneous ρ-blocks, we can enumerate all possible ways to satisfy these quotas. Let $\hat{\rho} := 2r_{\max}(\rho - 1) + r_{\max} - 1$. This is the maximum sum of requirements that may be assigned to any place and that are not part of a homogeneous ρ-block. We create configurations $c^F \in \{0, \ldots, \hat{\rho}\}^{r_{\max}}$ that tell us the number of families of each requirement type. Let us denote the set of possible configurations $C^F := \{0, \ldots, \hat{\rho}\}^{r_{\max}}$. It is clear that the number of possible configurations is bounded above by $(\hat{\rho} + 1)^{r_{\max}}$, which is a function of $r_{\max}$.

We say that a place type τ^P is *suitable* for a configuration c^F if $\tau^P[1] \le \sum_{r \in [r_{\max}]} c^F[r] \cdot r \le \tau^P[1] + \tau^P[2]$. This means that if the families are assigned to a place according to the configuration, its upper and lower quotas are satisfied.

We create an ILP with the following variables and constants:

- non-negative integer variables $\underline{b}_r$ and $\bar{b}_r$ for each $r \in [r_{\max}]$, representing the number of compulsory or optional, respectively, homogeneous ρ-blocks filled with families of requirement r.
- non-negative integer variable $x(\tau^P, c^F)$ for every $\tau^P \in \mathcal{T}^P$, $c^F \in C^F$ such that τ^P is suitable for c^F, counting the number of places of type τ^P that are assigned families according to configuration c^F.
- m_{τ^P} is the number of places of type τ^P for each $\tau^P \in \mathcal{T}^P$;
- $\underline{B} = \sum_{p_j \in P} \arg\operatorname{res}(\underline{\boldsymbol{c}}_j)$ (resp. $\overline{B} = \sum_{p_j \in P} \arg\operatorname{res}(\max(\bar{\boldsymbol{c}}_j - \underline{\boldsymbol{c}}_j - r_{\max} + 1), 0))$ is the number of compulsory (resp. optional) homogeneous ρ-blocks;
- n_r is the number of families $f_i \in F$ such that $\boldsymbol{r}_i = r$, for each $r \in [r_{\max}]$.

We create the following ILP:
(ILP$_1$)

$$\max \sum_{r \in [r_{\max}]} \left(\frac{\rho}{r}(\underline{b}_r + \bar{b}_r) + \sum_{\substack{\tau^P \in \mathcal{T}^P \\ c^F \in C^F}} c^F[r] x(\tau^P, c^F) \right) \quad \text{s.t.}$$

$$\forall \tau^P \in \mathcal{T}^P: \quad \sum_{\substack{c^F \in C^F \\ c^F \text{ is suitable for } \tau^P}} x(\tau^P, c^F) = m_{\tau^P} \tag{2}$$

$$\sum_{r \in [r_{\max}]} \underline{b}_r = \underline{B} \quad \text{and} \quad \sum_{r \in [r_{\max}]} \bar{b}_r \leq \overline{B} \tag{3}$$

$$\forall r \in [r_{\max}]: \quad \frac{\rho}{r}(\underline{b}_r + \bar{b}_r) + \sum_{\substack{\tau^P \in \mathcal{T}^P \\ c^F \in C^F}} c^F[r] x(\tau^P, c^F) \leq n_r \tag{4}$$

Constraint (2) enforces that every place has families matched to it according to some suitable configuration. Constraint (3) enforces that every compulsory homogeneous ρ-block is filled with refugee families and that no non-existing optional homogeneous ρ-blocks are filled with refugee families. Constraint (4) enforces that for each service-requirement, only available number of refugees are used. The objective function formulates the total number of families assigned.

It is clear that the number of variables is bounded above by $2r_{\max} + (\hat{\rho} + 1)^{r_{\max}} r_{\max}^2 \rho^2$, and the number of constraints by $3r_{\max}^2 \rho^2 + r_{\max}$, which are functions of $r_{\max}$. Thus the problem can be solved in FPT time w.r.t. $r_{\max}$ [27]. The correctness of this approach follows from Claim 1.

Claim 1 (⋆). *ILP$_1$ admits a solution with value u^* if and only if there is an assignment of families with utility u^*.* □

When preferences or utilities are not equal, parameterization by $r_{\max}$ alone does not yield fixed-parameter tractability: as established by Theorem 2, the case $r_{\max} = c_{\max} = 2$ is NP-hard even in a very restricted case, when there are no lower quotas, families have dichotomous preferences (or binary utilities), and each family finds at most two places acceptable (or of positive utility).

Let us remark that a slightly weaker result (the statement without the condition that each family finds exactly two places acceptable, or has positive utility for exactly two paces) follows via a fairly straightforward reduction from the MATCHING WITH COUPLES problem [11, 18].

Theorem 2 (⋆). PARETO-RR *and* MAXUTIL-RR *for $t = 1$ are NP-hard even when $r_{\max} = c_{\max} = 2$ and there are no lower quotas. The result holds for* PARETO-RR *even if all families have dichotomous preferences and find exactly two places acceptable, and for* MAXUTIL-RR *even if utilities are binary and each family has positive utilities for exactly two places.*

To tackle the computational intractability of Theorem 2, we focus on the parameter $m + r_{\max}$ and propose an FPT algorithm with this parameterization in Theorem 3.

To prove Theorem 3, we are going to present an algorithm for MAXUTIL-RR that constructs a feasible assignment with maximum utility, or concludes that no feasible assignment exists. By Observation 1, such an algorithm can be used to solve the PARETO-RR problem as well. Let I denote our input instance of MAXUTIL-RR.

We use a two-phase dynamic programming approach based on the following key idea: once we have obtained an optimal assignment σ for a partial instance $\mathcal{J}$, then a small modification to this partial instance results in an instance $\mathcal{J}'$ that admits an optimal assignment σ' that is "close" to σ. By guessing how σ' differs from σ, we can compute σ' efficiently. Let us give a high-level view of our algorithm.

In the first phase, we disregard lower quotas, and starting from an instance with only a single family, we add families one by one. For each $i \in [n]$, let $F_i = \{f_1, \dots, f_i\}$ denote the set of the first i families, and I_i the instance obtained by restricting I to F_i and setting all lower quotas as zero. Starting from a maximum-utility feasible assignment σ_1 for I_1, we construct a maximum-utility feasible assignment σ_i for $i = 2, \dots, n$ by slightly modifying the assignment σ_{i-1}.

In the second phase, starting from the instance $\hat{I}_0 = I_n$ without lower quotas, we define a sequence $\hat{I}_1, \dots, \hat{I}_{\underline{c}_\Sigma}$ of instances where each instance is obtained from the previous one by raising the lower quota of a single place by one in an arbitrary way so long as the lower quotas for I are not exceeded; notice that this implies $I = \hat{I}_{\underline{c}_\Sigma}$ where $\underline{c}_\Sigma = \sum_{p_j \in P} \underline{\boldsymbol{c}}_j$. Then starting from $\hat{\sigma}_0 := \sigma_n$ we compute a maximum-utility feasible assignment $\hat{\sigma}_q$ for $q = 1, \dots, \underline{c}_\Sigma$ from the assignment $\hat{\sigma}_{q-1}$ by applying small modifications.

Theorem 3. MAXUTIL- *and* PARETO-RR *for $t = 1$ are FPT w.r.t. $m + r_{\max}$.*

Proof. We may assume w.l.o.g. that there exists a feasible assignment for I_i with maximum utility that is complete. Indeed, to ensure this for each $i \in [n]$, we can simply create a dummy place whose upper quota is $\sum_{f_h \in F} \boldsymbol{r}_h$ and towards which all families have zero utility; this also shows that we can assume $m \geq 2$. For brevity's sake, we say that an assignment is *optimal* if it is feasible and complete, and has maximum utility among all feasible assignments.

Let $\rho_m^\star = m^m \cdot \rho \cdot r_{\max}$. Notice that since ρ is a function of $r_{\max}$, we know that $\rho_m^\star$ is a function of m and $r_{\max}$ only.

The first phase of our algorithm relies on Claim 2, which proves that given an optimal assignment σ_i for I_i for some $i \in [n-1]$, we can obtain an optimal assignment for I_{i+1} whose distance from σ_i is bounded by a function of m and $r_{\max}$. We measure the distance of two assignments σ and σ' as the total requirement of all families that are assigned to different places by σ and σ', that is,

$$\Delta(\sigma, \sigma') = \sum \left\{ \boldsymbol{r}_h : f_h \in F_\cap, \sigma(f_h) \neq \sigma'(f_h) \right\},$$

where $F_\cap$ is the intersection of the domains of σ and σ'.

Claim 2 (⋆). *Suppose that $i \in [n-1]$, and let $\sigma_i : F_i \to P$ denote an optimal allocation for I_i. Then there exists an optimal allocation $\sigma_{i+1} : F_{i+1} \to P$ for I_{i+1} such that $\Delta(\sigma_i, \sigma_{i+1}) \leq m \cdot \rho_m^\star$.*

The second phase of our algorithm relies Claim 3 which is an analog of Claim 2 with a quite similar proof. The proofs of Claims 2 and 3 can be found in Appendices B.5 and B.6. Recall that I_i and $\hat{I}_q$ are defined above.

Claim 3 ($\star$). *Suppose that $q \in [\underline{c}_\Sigma]$, and let $\hat{\sigma}_q : F \to P$ denote an optimal allocation for $\hat{I}_q$. Then there exists an optimal allocation $\hat{\sigma}_{q+1} : F \to P$ for $\hat{I}_{q+1}$ such that $\Delta(\hat{\sigma}_q, \hat{\sigma}_{q+1}) \leq m \cdot \rho^\star_m$.*

We are now ready to present our algorithm for MAXUTIL-RR based on Claims 2 and 3. We use a combination of dynamic programming and color-coding.

Initially, we compute a maximum-utility feasible allocation σ_1 for I_1 by assigning family f_1 to a place that can accommodate it, and among all such places, yields the highest utility for f_1. Then, in the first phase of the algorithm, for each $i \in [n-1]$ we compute an optimal assignment for I_{i+1} by slightly modifying σ_i. In the second phase, starting from the assignment $\hat{\sigma}_0 := \sigma_n$ for $\hat{I}_0 := I_n$, we compute an optimal assignment for $\hat{I}_q$ by slightly modifying $\hat{\sigma}_{q-1}$ for each $q \in [\underline{c}_\Sigma]$. In each step of the first and second phases, we apply a procedure based on color-coding; the remainder of the proof contains the description of this procedure and its proof of correctness.

Let I_{curr} be the instance of phase 1 or 2 for which we have already computed an optimal assignment σ_{curr}, and suppose that I_{next} is the next instance for which we aim to compute an optimal assignment σ_{next}. Thus, I_{next} is either obtained from I_{curr} by adding some family $f_i \in F$, or by raising the lower quota for one of the places in P by one. Let F_{curr} and F_{next} denote the set of families in I_{curr} and in I_{next}, respectively. Due to Claims 2 and 3, we can choose σ_{next} so that $\Delta(\sigma_{\text{next}}, \sigma_{\text{curr}}) \leq m \cdot \rho^\star_m$.

Guessing step. Let $X(p_j, p_{j'}, r)$ denote the set of all families with requirement r that are assigned to p_j by σ_{curr} but are moved to $p_{j'}$ by σ_{next}. We guess the number $x(p_j, p_{j'}, r) = |X(p_j, p_{j'}, r)|$ for each $p_j, p_{j'} \in P$ and $r \in [r_{\max}]$. By our choice of σ_{next}, we have

$$\sum_{j \in [m]} \sum_{j' \in [m] \setminus \{j\}} \sum_{r \in [r_{\max}]} x(p_{j'}, p_j, r) = \Delta(\sigma_{\text{next}}, \sigma_{\text{curr}}) \leq m \cdot \rho^\star_m.$$

Since we need to guess $m \cdot (m-1) \cdot r_{\max}$ values that add up to at most $m \cdot \rho^\star_m$, there are no more than $(\rho^\star_m)^{m \cdot (m-1) \cdot r_{\max}}$ possibilities to choose all values $x(p_{j'}, p_j, r)$. Thus, the number of possibilities for all our guesses is bounded by a function of m and $r_{\max}$ only.

Color-coding step. We proceed by randomly coloring all families in I_{next} with m colors in a uniform and independent way. We say that a coloring is *suitable* for σ_{next}, if for each $p_j \in P$, all families in $\sigma^{-1}_{\text{next}}(p_j) \setminus \sigma^{-1}_{\text{curr}}(p_j)$ have color j. Thus, in a suitable coloring, each family whose assignment changes between σ_{next} and σ_{curr} must be assigned by σ_{next} to the place corresponding to its color. Considering that I_{next} may contain one more family than I_{curr}, we get

$$\sum_{p_j \in P} \left| \sigma^{-1}_{\text{next}}(p_j) \setminus \sigma^{-1}_{\text{curr}}(p_j) \right| \leq 1 + \Delta(\sigma_{\text{next}}, \sigma_{\text{curr}}) \leq m \cdot \rho^\star_m + 1.$$

Therefore, the probability that the algorithm produces a suitable coloring is at least $m^{-m\rho^\star_m + 1}$.

Modification step. Assume that our coloring χ is suitable. In the first phase, this implies that the unique family $f_i \in F_{\text{next}} \setminus F_{\text{curr}}$ must be assigned by σ_{next} to $p_{\chi(f_i)}$. Thus, we fix the assignment on f_i as $p_{\chi(f_i)}$. We proceed with the remaining families of F_{next} as follows.

For each $p_j, p_{j'} \in P$ and $r \in [r_{\max}]$, we compute the set $D(p_j, p_{j'}, r) := \{f_h \in F_{\text{curr}} : \sigma_{\text{curr}}(f_h) = p_j, \chi(f_h) = j', \boldsymbol{r}_h = r\}$; the suitability of χ means that $X(p_j, p_{j'}, r) \subseteq D(p_j, p_{j'}, r)$. With each family $f_h \in D(p_j, p_{j'}, r)$, we associate the value $\boldsymbol{u}^{j'}_h - \boldsymbol{u}^j_h$ which describes the increase in utility caused by moving a_h from p_j to $p_{j'}$. We order the families in $D(p_j, p_{j'}, r)$ in a non-increasing order of these values, and we pick the first $x(p_j, p_{j'}, r)$ families according to this ordering; denote the obtained set $\widetilde{D}(p_j, p_{j'}, r)$. We can now define σ'_{i+1} as follows for each $f_h \in F_{\text{curr}}$:

$$\sigma'_{\text{next}}(f_h) = \begin{cases} p_{\chi(f_h)} & \text{if } f_h \in F_{\text{next}} \setminus F_{\text{curr}}; \\ p_{j'} & \text{if } \exists j, r : f_h \in \widetilde{D}(p_j, p_{j'}, r); \\ \sigma_{\text{curr}}(f_h) & \text{otherwise.} \end{cases}$$

Observe that the total requirement of all families assigned to some place $p_j \in P$ is the same in σ'_{next} as in σ_{next}, due to the definition σ'_{next} and the correctness of our guesses. Therefore, σ'_{next} is feasible. Furthermore,

$$\sum_{\substack{f_h \in F_{\text{curr}}, \\ p_j = \sigma'_{\text{next}}(f_h)}} \boldsymbol{u}_h[j] = \text{util}(p_j, \sigma_{\text{curr}}) + \sum_{\substack{\exists j, j', r: \\ f_h \in \widetilde{D}(p_j, p_{j'}, r)}} (\boldsymbol{u}_h[j'] - \boldsymbol{u}_h[j])$$

$$\geq \text{util}(p_j, \sigma_{\text{curr}}) + \sum_{\substack{\exists j, j', r: \\ f_h \in X(p_j, p_{j'}, r)}} (\boldsymbol{u}_h[j'] - \boldsymbol{u}_h[j]) = \sum_{\substack{f_h \in F_{\text{curr}}, \\ p_j = \sigma_{\text{next}}(f_h)}} \boldsymbol{u}_h[j]$$

where the inequality follows from our choice of the sets $\widetilde{D}(p_j, p_{j'}, r)$ and the facts $|\widetilde{D}(p_j, p_{j'}, r)| = |X(p_j, p_{j'}, r)|$ and $X(p_j, p_{j'}, r) \subseteq D(p_j, p_{j'}, r)$, which in turn follow from our assumptions that our guesses are correct and that the coloring χ is suitable. Since σ'_{next} coincides with σ_{next} on $F_{\text{next}} \setminus F_{\text{curr}}$, the above inequality implies that σ'_{next} is a maximum-utility feasible assignment for I_{next}, proving the correctness of our algorithm.

The presented algorithm can be derandomized using standard techniques, based on $(n, m \cdot \rho^\star_m + 1)$-perfect families of perfect hash functions [4]. Since both the number of possible guesses and the number of families that we have to color correctly are bounded by a function of $m + r_{\max}$, the modification procedure applied in the first or second phases of the algorithm runs in FPT time when parameterized by $m + r_{\max}$. As we have to carry out this procedure $n + \underline{c}_\Sigma$ times and we can assume w.l.o.g. that $\underline{c}_\Sigma \leq n \cdot r_{\max}$, the total running time is FPT w.r.t. $m + r_{\max}$. $\square$

We close this section by showing that if the desired total utility u^* is small and there are no lower quotas, then MAXUTIL-RR for $t = 1$ can be solved efficiently. Recall that with lower quotas, even the case $u^* = 0$ is NP-hard by Proposition 1. The algorithm of Theorem 4, presented in Appendix B.7, starts with a greedily computed assignment, and then deletes irrelevant families to obtain an equivalent instance with at most $(u^*)^3$ families that can be solved efficiently.

Theorem 4 ($\star$). MAXUTIL- *and* PARETO-RR *for $t = 1$ are FPT w.r.t. u^*, the desired utility, if there are no lower quotas.*

4 Multiple services

Let us now consider the model when there are several services, i.e., $t > 1$. We start with a strong intractability result for FEASIBLE-RR. Then we focus in Pareto-optimality, and propose several algorithms that solve PARETO-RR but not MAXUTIL-RR, contrasted by tight hardness results. We close by investigating MAXUTIL-RR.

Feasibility. When the number of services can be unbounded, then a simple reduction from INDEPENDENT SET by Gurski et al. [19, Theorem 23] shows that MAXUTIL-RR is NP-hard even if $m = 1$, there are no lower quotas and the utilities are equal. With a slight modification of their reduction, we obtain Proposition 2 which shows the NP-hardness of FEASIBLE-RR in a very restricted setting.

Proposition 2 (⋆). *The following problems are NP-hard even if $c_{\max} = r_{\max} = 1$ and $m = 1$:*
- FEASIBLE-RR*;*
- PARETO-RR *with equal preferences;*
- MAXUTIL-RR *with equal utilities.*

Pareto-optimality. The reduction from INDEPENDENT SET used by Gurski et al. [19] and also in Proposition 2 can be adapted to show the NP-hardness of PARETO-RR in the case when there are no lower quotas, $m = 2$, and we allow $c_{\max}$ to be unbounded; see Proposition 3. Notice that in instances without lower quotas, a feasible, acceptable and Pareto-optimal assignment always exists. Hence, our hardness results for PARETO-RR rely on the following fact.

Observation 3 (⋆). *Given an instance I of* PARETO-RR *with dichotomous preferences, we can decide the existence of a feasible, acceptable and complete assignment for I by solving* PARETO-RR *on I.*

Proposition 3 (⋆). PARETO- *and* MAXUTIL-RR *are NP-hard even if $m = 2, r_{\max} = 1$, there are no lower quotas, and families have equal preferences or utilities.*

In the reduction proving Proposition 3, the value $c_{\max}$ is unbounded. Next, we show a reduction from 3-COLORING proving that even the case when $c_{\max} = 1$ is NP-hard if there are at least 3 places.

Theorem 5 (⋆). PARETO- *and* MAXUTIL-RR *are NP-hard even when $m = 3, c_{\max} = 1$, there are no lower quotas, and families have equal preferences or utilities, respectively.*

Contrasting the intractability result of Proposition 3 for $m = 2$, we show that a simple, greedy algorithm solves PARETO-RR for $m = 1$ in polynomial time assuming that there are no lower quotas.

Proposition 4 (⋆). PARETO-RR *for $m = 1$ is polynomial-time solvable if there are no lower quotas.*

Our next results shows that PARETO-RR can be solved efficiently if there are only a few families whose preferences contain ties, assuming that there are no lower quotas. Recall that in the presence of lower quotas, PARETO-RR is NP-hard even if $t = 1$ and $n_{\sim} = 0$, i.e., all preferences are strict, as shown in Proposition 1. The algorithm of Proposition 5 first applies serial dictatorship among families whose preferences do not contain ties, and then tries all possible assignments for the remaining families.

Proposition 5 (⋆). PARETO-RR *is FPT w.r.t. the number of families with ties $n_{\sim}$, if there are no lower quotas.*

Maximizing utility. Let us start with a simple fixed-parameter tractable algorithm for MAXUTIL-RR w.r.t n, the number of families. Proposition 6 presents an FPT algorithm for parameter n based on the following approach: We first guess the partitioning $\mathcal{F}$ of families arising from a maximum-weight feasible assignment, and then we map the partitions of $\mathcal{F}$ to the places by computing a maximum-weight matching in an auxiliary bipartite graph.

Proposition 6 (⋆). FEASIBLE-, PARETO-, *and* MAXUTIL-RR *are FPT w.r.t. n.*

Let us now present a generalization of Theorem 1 for MAXUTIL-RR restricted to equal preferences. The algorithm for Theorem 6 is based upon an N-fold IP formulation for this problem. By Observation 1, the obtained algorithm also implies tractability for PARETO-RR for equal utilities.

Theorem 6 (⋆). FEASIBLE-RR, PARETO-RR *for equal preferences, and* MAXUTIL-RR *for equal utilities, are FPT w.r.t. $t + r_{\max}$.*

Our next algorithm is applicable in a more general case than Theorem 6 (which works only when preferences or utilities are equal) at the cost of setting $m + t + r_{\max}$ as the parameter (recall that the parameter considered in Theorem 6 is $t + r_{\max}$). Theorem 6 is based on an ILP formulation that solves PARETO-RR for arbitrary preferences as well as MAXUTIL-RR for a broad range of utilities.

Theorem 7 (⋆). *The following problems are FPT w.r.t. parameter $m + t + r_{\max}$:*
- PARETO-RR,
- MAXUTIL-RR *on instances where the number of different utility values is at most $g(m+t+r_{\max})$ for some computable function g.*

Proof sketch. The main idea is that there is no need to distinguish between families that have the same utilities and requirements. Since the number of possible requirement vectors and the number of possible utility vectors are both bounded by a function of the parameter, the number of family types will also be bounded. This allows us to define a variable for each place and family type describing the number of families of a given type assigned to a given place. The resulting ILP contains a bounded number of variables and constraints, and is therefore solvable by standard techniques in FPT time [27]. □

Taking an even stronger parameterization than Theorem 7, namely $m + t + c_{\max}$, yields fixed-parameter tractability: in Proposition 7 we present an algorithm running in $O((c_{\max})^{mt} nm)$ time. This algorithm is a straightforward adaptation of the textbook dynamic programming method for KNAPSACK. The same approach was also used by Gurski et al. [19, Proposition 34] to solve a simpler variant of MAXUTIL-RR without lower quotas and with a single service.

Proposition 7 (⋆). FEASIBLE-, PARETO-RR, *and* MAXUTIL-RR *are in XP w.r.t. $m + t$ and are FPT w.r.t. $m + t + c_{\max}$.*

We close this section by mentioning that a simple XP algorithm exists for the case when the parameter is the desired total utility u^*, and there are no lower quotas (cf. Proposition 1 stating the intractability of the case $u^* = 0$ when lower quotas are allowed).

Proposition 8 (⋆). FEASIBLE-, PARETO- *and* MAXUTIL-RR *are in XP w.r.t. the desired utility u^* if there are no lower quotas.*

5 Conclusion

We provided a comprehensive parameterized complexity analysis for three variants of REFUGEE RESETTLEMENT, which focus on ensuring feasibility, maximizing utility, and achieving Pareto optimality. There remain some interesting parameter combinations for which the complexity of these problems is open, e.g., is MAXUTIL-RR FPT with parameter $m + t + r_{\max}$ for arbitrary utilities? Another exciting line of future research is to explore the possibilities of tailoring the proposed algorithms to efficiently solve practical instances, and determining which parameterizations are the most relevant in different real-world applications.We believe that our ILP algorithms could perform substantially faster then their theoretical bounds, as the current day ILP solvers are efficient. However, it could also be that a straightforward ILP formulation with a variable for each family and each place outperforms our specialized formulations.

Acknowledgements

Jiehua Chen and Sofia Simola are supported by the Vienna Science and Technology Fund (WWTF) [10.47379/VRG18012]. Ildikó Schlotter is supported by the Hungarian Academy of Sciences under its Momentum Programme (LP2021-2) and its János Bolyai Research Scholarship.

References

[1] A. Abdulkadiroğlu. College admissions with affirmative action. *International Journal of Game Theory*, 33:535–549, 2005.

[2] N. Ahani, T. Andersson, A. Martinello, A. Teytelboym, and A. C. Trapp. Placement optimization in refugee resettlement. *Operations Research*, 69(5):1468–1486, 2021.

[3] N. Ahani, P. Gölz, A. D. Procaccia, A. Teytelboym, and A. C. Trapp. Dynamic placement in refugee resettlement. *Operations Research*, 72 (3):1087–1104, 2023.

[4] N. Alon, R. Yuster, and U. Zwick. Color-coding. *Journal of the ACM*, 42(4):844–856, 1995.

[5] T. Andersson and L. Ehlers. Assigning refugees to landlords in Sweden: Efficient, stable, and maximum matchings. *The Scandinavian Journal of Economics*, 122(3):937–965, 2020.

[6] T. Andersson, L. Ehlers, and A. Martinello. Dynamic refugee matching. Technical Report 2018-10, Université de Montréal, Département de sciences économiques, 2018.

[7] H. Aziz, J. Chen, S. Gaspers, and Z. Sun. Stability and Pareto optimality in refugee allocation matchings. In *Proceedings of the 17th International Conference on Autonomous Agents and MultiAgent Systems (AAMAS 2018)*, pages 964–972, 2018.

[8] H. Aziz, S. Gaspers, Z. Sun, and T. Walsh. From matching with diversity constraints to matching with regional quotas. In *Proceedings of the 18th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2019)*, pages 377–385, 2019.

[9] K. Bansak and E. Paulson. Outcome-driven dynamic refugee assignment with allocation balancing. *Operations Research*, 2024. doi: 10.1287/opre.2022.0445.

[10] P. Berman, M. Karpinski, and A. D. Scott. Approximation hardness of short symmetric instances of MAX-3SAT. Technical Report TR03-049, Electronic Colloquium on Computational Complexity, 2003.

[11] P. Biró and E. McDermid. Matching with sizes (or scheduling with processing set restrictions). *Discrete Applied Mathematics*, 164:61–67, 2014.

[12] P. Biró, T. Fleiner, R. W. Irving, and D. Manlove. The College Admissions problem with lower and common quotas. *Theoretical Computer Science*, 411(34-36):3136–3153, 2010.

[13] J. Chen, R. Ganian, and T. Hamm. Stable matchings with diversity constraints: Affirmative action is beyond NP. In *Proceedings of the 29th International Joint Conference on Artificial Intelligence (IJCAI 2020)*, pages 146–152, 2020.

[14] D. Delacrétaz, S. D. Kominers, and A. Teytelboym. Refugee resettlement, 2016. Working paper. Available at http://www.t8el.com/jmp.pdf.

[15] D. Delacrétaz, S. D. Kominers, and A. Teytelboym. Matching mechanisms for refugee resettlement. *American Economic Review*, 113(10): 2689–2717, 2023.

[16] F. Eisenbrand, C. Hunkenschröder, and K.-M. Klein. Faster Algorithms for Integer Programs with Block Structure. In *45th International Colloquium on Automata, Languages, and Programming (ICALP 2018)*, volume 107, pages 49:1–49:13, 2018.

[17] R. Ganian, S. Ordyniak, and C. S. Rahul. Group activity selection with few agent types. *Algorithmica*, 85(5):1111–1155, 2023.

[18] C. A. Glass and H. Kellerer. Parallel machine scheduling with job assignment restrictions. *Naval Research Logistics*, 54(3):250–257, 2007.

[19] F. Gurski, C. Rehs, and J. Rethmann. Knapsack problems: A parameterized point of view. *Theoretical Computer Science*, 775:93–108, 2019.

[20] K. Hamada, K. Iwama, and S. Miyazaki. The Hospitals/Residents problem with lower quotas. *Algorithmica*, 74(1):440–465, 2016.

[21] R. Hemmecke, S. Onn, and L. Romanchuk. n-fold integer programming in cubic time. *Mathematical Programming, Series A*, 137:325–341, 2013.

[22] K. Jansen, S. Kratsch, D. Marx, and I. Schlotter. Bin packing with fixed number of bins revisited. *Journal of Computer and System Sciences*, 79 (1):39–49, 2013.

[23] D. Knop and S. Schierreich. Host community respecting refugee housing. In *Proceedings of the 2023 International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2023)*, page 966–975, Richland, SC, 2023.

[24] D. Knop, M. Koutecký, and M. Mnich. Combinatorial n-fold Integer Programming and applications. *Math. Program.*, 184(1):1–34, 2020.

[25] B. Kuckuck, J. Rothe, and A. Weißenfeld. Refugee allocation in the setting of hedonic games. In *Proceedings of the 6th International Conference on Algorithmic Decision Theory (ADT 2019)*, pages 65–80. Springer, 2019.

[26] R. Kurata, N. Hamada, A. Iwasaki, and M. Yokoo. Controlled school choice with soft bounds and overlapping types. *Journal of Artificial Intelligence Research*, 58:153–184, 2017.

[27] H. W. Lenstra Jr. Integer programming with a fixed number of variables. *Mathematics of Operations Research*, 8(4):538–548, 1983.

[28] G. Lisowski and Š. Schierreich. Swap stability in refugee housing: A story about anonymous preferences. *The 10th European Starting AI Researchers' Symposium (STAIRS)*, 2023.

[29] E. J. McDermid and D. F. Manlove. Keeping partners together: algorithmic results for the hospitals/residents problem with couples. *Journal of Combinatorial Optimization*, 19:279–303, 2010.

[30] H. Nguyen, T. Nguyen, and A. Teytelboym. Stability in matching markets with complex constraints. *Management Science*, 67(12):7438–7454, 2021.

[31] K. Pietrzak. On the parameterized complexity of the fixed alphabet shortest common supersequence and longest common subsequence problems. *Journal of Computer and System Sciences*, 67(4):757–771, 2003.

[32] Š. Schierreich. Anonymous refugee housing with upper-bounds, 2023. arXiv preprint, arXiv:2308.09501.

[33] P. Xepapadeas and I. Mourtos. Refugee allocation mechanisms: Theory and applications for the European Union. *Operational Research*, 22(4): 4557–4584, 2022.

Appendix A Additional material for Section 2

A.1 *Example for* Refugee Resettlement

Consider an instance of Refugee Resettlement with four families, f_1, f_2, f_3, and f_4, two places p_1 and p_2. Families may require two kinds of services: housing, which is determined directly by the number of people in the family, and school seats for their children.

The requirement of the families is as follows.

f_1: $\boldsymbol{r}_1 = (4, 2)$, i.e., a family of four with two school-age children;
f_2: $\boldsymbol{r}_2 = (2, 0)$, i.e., a family of two, no children;
f_3: $\boldsymbol{r}_3 = (6, 2)$, i.e., a family of six with two school-age children;
f_4: $\boldsymbol{r}_4 = (3, 1)$, i.e., a family of three with one school-age child.

Place p_1 offers housing for at most 10 people, while place p_2 can house at most 8 people. To promote diversity, both places aim to enroll at least two refugee children in their schools, and both have three school seats to offer. Hence, the lower and upper quotas are as follows:

p_1: $\underline{\boldsymbol{c}}_1 = (0, 2)$ and $\bar{\boldsymbol{c}}_1 = (10, 3)$;
p_2: $\underline{\boldsymbol{c}}_2 = (0, 2)$ and $\bar{\boldsymbol{c}}_2 = (8, 3)$.

Both places are acceptable to each refugee family, and the preferences of the families are defined as

f_1: $p_1 \succ p_2$;
f_2: $p_2 \succ p_1$;
f_3: $p_2 \succ p_1$;
f_4: $p_2 \succ p_1$.

Alternatively, families may have the following utility vectors:

f_1: $\boldsymbol{u}_1 = (2, 1)$;
f_2: $\boldsymbol{u}_2 = (1, 2)$;
f_3: $\boldsymbol{u}_3 = (1, 2)$;
f_4: $\boldsymbol{u}_4 = (1, 2)$.

Consider the assignment σ that assigns families f_2 and f_3 to p_1, and assigns families f_1 and f_4 to p_2. It is clear that

$$(0, 2) = \underline{\boldsymbol{c}}_1 \leq \sum_{f_i \in \sigma^{-1}(p_1)} \boldsymbol{r}_i = (8, 2) \leq \bar{\boldsymbol{c}}_1 = (10, 3) \text{ and}$$
$$(0, 2) = \underline{\boldsymbol{c}}_2 \leq \sum_{f_i \in \sigma^{-1}(p_1)} \boldsymbol{r}_i = (7, 3) \leq \bar{\boldsymbol{c}}_2 = (8, 3),$$

implying that σ is a feasible assignment.

It is also not hard to see that σ is Pareto-optimal as well. To see this, consider first family f_4 who is assigned to its favorite place. Notice that once we decide to assign f_4 to p_2, neither f_2 nor f_3 may be additionally assigned to p_2: the former would violate the upper quota of p_2 to house at most 8 people, while the latter would violate the lower quota of p_2 to enroll at least two children in its school (unless some further family is added, violating the upper quota). Notice also that once f_2 and f_3 are both assigned to p_1, it is not possible to accommodate f_1 alongside them at p_1, as that would violate the upper quota of p_1 to house at most 10 people. Therefore, without disimproving the situation of f_4, we cannot improve the situation of any other family in any feasible assignment, showing the Pareto-optimality of σ.

It is also clear that in the setting with utilities, the total utility of σ is $\mathrm{util}(\sigma) = 3 \cdot 1 + 2 = 5$.

Consider now the assignment σ' that assigns families f_1 and f_4 to p_1, and assigns families f_2 and f_3 to p_2. It is clear that

$$(0, 2) = \underline{\boldsymbol{c}}_1 \leq \sum_{f_i \in \sigma'^{-1}(p_1)} \boldsymbol{r}_i = (7, 3) \leq \bar{\boldsymbol{c}}_1 = (10, 3) \text{ and}$$
$$(0, 2) = \underline{\boldsymbol{c}}_2 \leq \sum_{f_i \in \sigma'^{-1}(p_1)} \boldsymbol{r}_i = (8, 2) \leq \bar{\boldsymbol{c}}_2 = (8, 3),$$

implying that σ' is a feasible assignment.

We claim that σ' is a maximum-utility feasible assignment. Clearly, its utility is $\mathrm{util}(\sigma') = 3 \cdot 2 + 1 = 7$. Thus, any assignment whose utility exceeds $\mathrm{util}(\sigma')$ must assign all families to the place for which they have utility 2. However, such an assignment is not feasible, as assigning each of f_2, f_3, and f_4 to p_2 would violate its upper quota to house at most 8 people.

A.2 *Proof of Observation 1*

Observation 1 (⋆)**.** *Given an instance I of* Pareto-RR*, construct an instance I' of* MaxUtil-RR *as follows. For each family $f_i \in F$:*
- *for every place p_j that f_i finds acceptable, set $\boldsymbol{u}_i[j] = |\{p_{j'} \in P \mid p_j \succeq_i p_{j'}\}|$;*
- *for each place p_j that f_i finds unacceptable, set $\boldsymbol{u}_i[j] = -m \cdot n$.*

Let σ be a maximum-utility feasible assignment for I'. If $\mathrm{util}(\sigma) > 0$*, then σ is a feasible, acceptable, and Pareto-optimal assignment for I; otherwise there is no feasible and acceptable assignment for I.*

Proof. Clearly, σ has positive utility if σ is an acceptable assignment for I, and it has negative utility if it is unacceptable for I (since already one utility value of $-m \cdot n$ guarantees $\mathrm{util}(\sigma) < 0$). Hence, if $\mathrm{util}(\sigma) < 0$, then no feasible assignment exists for I. Otherwise, we know that σ is feasible for I. Since σ has maximum utility among all assignments, it is necessarily Pareto-optimal, because a Pareto-improvement would imply an assignment with greater utility value. □

Appendix B Additional material for Section 3

B.1 *Proof of Proposition 1*

Proposition 1 (⋆)**.** *The following problems are W[1]-hard w.r.t. m for $t = 1$:*
- Feasible-RR*;*
- Pareto-RR *with no lower quotas and equal preferences;*
- Pareto-RR *when all families have strict preferences;*
- MaxUtil-RR *with no lower quotas and equal utilities;*
- MaxUtil-RR *with $u^* = 0$.*

Proof. We give a reduction from the following variant of Bin Packing. The input instance I_{BP} contains item sizes $a_1, \dots, a_n$ and an integer k, and the question is whether these items can be allocated into k bins such that the total size of items allocated to each bin is exactly $B = (\sum_{i=1}^{n} a_i)/k$, the bin size. This problem is known to be W[1]-hard w.r.t. k [22].

Let us construct an instance I_{FRR} of Feasible-RR with $t = 1$. We define families $f_1, \dots, f_n$ and set the requirement of family f_i as a_i for each $i \in [n]$. We also define places $p_1, \dots, p_k$, each with its lower and upper quota both set to B. Then I_{FRR} admits a feasible assignment if and only if our Bin Packing instance I_{BP} is solvable; this proves the result for Feasible-RR.

To obtain a Pareto-RR instance I_{PRR}, we can reset the lower quotas to zero, and set equal preferences for each family. Then I_{PRR} admits a feasible and complete assignment if and only if I_{BP} admits a solution, because the total requirement of the families is kB which equals the total upper quota of the places. It remains to recall that by Observation 3, we can decide whether a feasible and complete assignment exists for I_{PRR} by solving PARETO-RR on I_{PRR}. This proves the first result for PARETO-RR.

Next we show that PARETO-RR is W[1]-hard even when all families have strict preferences. Assume, towards a contradiction, that there is an algorithm that can find a Pareto-optimal feasible and acceptable assignment in FPT time w.r.t. m. We show that such an algorithm could be used to solve FEASIBLE-RR in FPT time w.r.t. m, contradicting the first result unless FPT = W[1]. Let I be an instance of FEASIBLE-RR. We reduce it to an instance $I^{\succ}$ of PARETO-RR by giving every family arbitrary complete strict preferences over the places. The instances are otherwise identical. Observe that any assignment of $I^{\succ}$ is acceptable, and an assignment σ is feasible on $I^{\succ}$ if and only if it is feasible on I. If an algorithm for PARETO-RR finds a Pareto-optimal feasible and acceptable assignment σ for $I^{\succ}$, then σ is also a feasible assignment of I, and we can report I is a yes-instance. Correspondingly, if an algorithm for PARETO-RR reports there is no feasible and acceptable Pareto-optimal assignment σ of $I^{\succ}$, then there is no feasible and acceptable assignment of $I^{\succ}$, and thus no feasible assignment of I, and we can report I is a no-instance.

It is straightforward to check that by setting unit utilities instead of equal preferences we obtain an instance I_{MURR} of MAXUTIL-RR that is equivalent with I_{PRR} in the sense that I_{MURR} admits a feasible assignment with utility at least kB if and only if I_{PRR} admits a complete and feasible assignment, proving the fourth statement of the theorem. Finally, the hardness results for FEASIBLE-RR also hold for MAXUTIL-RR even if we set all utilities as zero and $u^* = 0$, proving the last statement. ☐

B.2 *Proof of Observation 2*

Observation 2 (⋆). *Suppose that the number of services is $t = 1$. If $F' \subseteq F$ is a set of families such that $\sum_{f_i \in F'} \boldsymbol{r}_i > r_{\max}(\rho - 1)$, then F' contains a homogeneous ρ-block.*

Proof. Group families in F' according to their requirements so that families with the same requirement are contained in one group. Clearly, if there is a group whose total requirement is at least ρ, then this group contains a homogeneous ρ-block, because ρ is divisible by the (common) requirement of the families in the group. If each group has total requirement at most $\rho - 1$, then the total requirement of all families in F' is at most $r_{\max}(\rho - 1)$, as required. ☐

B.3 *Proof of Claim 1*

Claim 1 (⋆). *ILP$_1$ admits a solution with value u^* if and only if there is an assignment of families with utility u^*.*

Proof. We prove the two directions of the claim separately.

Claim 4. *If ILP$_1$ admits a solution with value u^*, then there is an assignment of families with utility u^*.*

Proof. Assume there is a solution to the constructed ILP-instance with value u^*. We construct an assignment σ. Start with $\sigma(f_i) = \perp$ for every $f_i \in F$.

Let us create a counter β_r for each $r \in [r_{\max}]$ and initialize it by setting $\beta_r := \underline{b}_r$. Similarly, let us create a counter ψ_j for each place $p_j \in P$ and initialize it by setting $\psi_j := \arg\operatorname{res}(\underline{\boldsymbol{c}}_j)$.

Step 1. We build σ as follows: While there exists some $r \in [r_{\max}]$ such that $\beta_r > 0$,

(i) choose $p_j \in P$ such that $\psi_j > 0$,
(ii) assign a homogeneous ρ-block with requirement r to p_j, and
(iii) update $\beta_r := \beta_r - 1$ and $\psi_j := \psi_j - 1$.

Since the solution is valid, by Constraint (3) we know that $\sum_{r \in [r_{\max}]} \underline{b}_r = \underline{B} = \sum_{p_j \in P} \arg\operatorname{res}(\underline{\boldsymbol{c}}_j)$. Therefore at the end of the process, $\sum_{r \in [r_{\max}]} \beta_r = \sum_{p_j \in P} \psi_j = 0$.

Step 2. Next we assign the families that are matched in optional homogeneous ρ-blocks. Let us create a counter $\bar{\beta}_r$ for each $r \in [r_{\max}]$ and initialize $\bar{\beta}_r := \bar{b}_r$. Similarly, let us create a counter $\bar{\psi}_j$ for each $p_j \in P$ and initialize $\bar{\psi}_j := \arg\operatorname{res}(\max(\underline{\boldsymbol{c}} - \underline{\boldsymbol{c}}_j - r_{\max} + 1, 0))$.

We build σ as follows: While there exists some $r \in [r_{\max}]$ such that $\bar{\beta}_r > 0$,

(i) choose $p_j \in P$ such that $\bar{\psi}_j > 0$,
(ii) assign a homogeneous ρ-block with requirement r to p_j, and
(iii) update $\bar{\beta}_r := \bar{\beta}_r - 1$ and $\bar{\psi}_j := \bar{\psi}_j - 1$.

Since the solution is valid, by Constraint (3) we know that $\sum_{r \in [r_{\max}]} \bar{b}_r \leq \overline{B} = \sum_{p_j \in P} \arg\operatorname{res}(\max(\underline{\boldsymbol{c}} - \underline{\boldsymbol{c}}_j - r_{\max} + 1, 0))$. Therefore while $\sum_{r \in [r_{\max}]} \bar{\beta}_r \geq 0$, we can always find $p_j \in P$ such that $\psi_j > 0$.

Step 3. Finally, we look at every place type $\tau^P \in \mathcal{T}^P$. We have

$$\sum_{\substack{c^F \in C^F \\ c^F \text{ is suitable for } \tau^P}} x(\tau^P, c^F) = m_{\tau^P} \tag{5}$$

due to Constraint (2), where m_{τ^P} is the number of places of type τ^P. For each place type τ^P do the following: for each $c^F \in C^F$, assign $x(\tau^P, c^F)$ many families to places of type τ^P according to configuration c^F: for each place p_j of type τ^P, we assign a set of families having configuration c^F to p_j, i.e., for each $r \in [r_{\max}]$, we assign $c^F[r]$ many families with requirement r to the place p_j. By the equality in (5), every place of type τ^P will be treated exactly once in this step.

Correctness. This concludes the description of the assignment. It remains to show that this assignment is feasible and matches u^* many agents. We start by showing that the quotas of the places are respected. Let $p_j \in P$ be an arbitrary place and let τ_j^P be the type of p_j. In Step 1, we assign $\arg\operatorname{res}(\underline{\boldsymbol{c}}_j)$ many homogeneous ρ-blocks to p_j. In Step 2, we assign at most $\arg\operatorname{res}(\max(\underline{\boldsymbol{c}} - \underline{\boldsymbol{c}}_j - r_{\max} + 1, 0))$ many homogeneous ρ-blocks to p_j. In Step 3, we choose some $c^F \in C^F$ that is suitable for τ_j^P and assign families accordingly. Thus, $\tau_j^P[1] \leq R \leq \tau_j^P[1] + \tau_j^P[2]$ holds for the total requirement R of the families assigned to p_j in Step 3. Thus $R \geq \operatorname{res}(\underline{\boldsymbol{c}}_j)$.

Therefore, the total requirement of the families assigned to p_j is at least $\rho \cdot \arg\operatorname{res}(\underline{\boldsymbol{c}}_j) + \operatorname{res}(\underline{\boldsymbol{c}}_j) \overset{(1)}{=} \underline{\boldsymbol{c}}_j$, as required.

If $\bar{\boldsymbol{c}} - \underline{\boldsymbol{c}}_j \geq r_{\max} - 1$, then we know

$$R \leq \operatorname{res}(\underline{\boldsymbol{c}}_j) + \operatorname{res}(\bar{\boldsymbol{c}}_j - \underline{\boldsymbol{c}}_j - r_{\max} + 1) + r_{\max} - 1.$$

Otherwise, $R \leq \operatorname{res}(\underline{\boldsymbol{c}}_j) + \bar{\boldsymbol{c}}_j - \underline{\boldsymbol{c}}_j$.

If $\bar{\boldsymbol{c}} - \underline{\boldsymbol{c}}_j \geq r_{\max} - 1$, then the total requirement of the families assigned to p_j is at most

$$\begin{aligned}\rho \cdot \arg\operatorname{res}(\underline{\boldsymbol{c}}_j) + \rho \cdot \arg\operatorname{res}(\bar{\boldsymbol{c}}_j - \underline{\boldsymbol{c}}_j - r_{\max} + 1) + \operatorname{res}(\underline{\boldsymbol{c}}_j) \\ + \operatorname{res}(\bar{\boldsymbol{c}}_j - \underline{\boldsymbol{c}}_j - r_{\max} + 1) + r_{\max} - 1 \\ \overset{(1)}{=} \underline{\boldsymbol{c}}_j + \bar{\boldsymbol{c}}_j - \underline{\boldsymbol{c}}_j - r_{\max} + 1 + r_{\max} - 1 = \bar{\boldsymbol{c}}_j,\end{aligned}$$

as required.

Otherwise, the total requirement of the families assigned to p_j is at most $\rho \cdot \arg\operatorname{res}(\underline{\boldsymbol{c}}_j) + 0 + \operatorname{res}(\underline{\boldsymbol{c}}_j) + \bar{\boldsymbol{c}}_j - \underline{\boldsymbol{c}}_j \overset{(1)}{=} \underline{\boldsymbol{c}}_j + \bar{\boldsymbol{c}}_j - \underline{\boldsymbol{c}}_j = \bar{\boldsymbol{c}}_j$, as required.

Next, we show that all the families that are assigned in σ exist. Let $r \in [r_{\max}]$ be an arbitrary service requirement. Whenever we match a homogeneous ρ-block of requirement r, we match $\frac{\rho}{r}$ many families of this requirement. Therefore in Step 1 we match $\frac{\rho}{r} \cdot \underline{b}_r$ families of requirement i. Similarly, in Step 2 we match $\frac{\rho}{r} \cdot \bar{b}_r$ families of requirement r. In Step 3 we assign $c^F[r] \cdot x(\tau^P, c^F)$ many families for each $\tau^P \in \mathcal{T}^P, c^F \in C^F$. Thus the total number of families of requirement r matched is

$$\frac{\rho}{r}(\underline{b}_r + \bar{b}_r) + \sum_{\substack{\tau^P \in \mathcal{T}^P \\ c^F \in C^F}} c^F[r] x(\tau^P, c^F).$$

By Constraint (4) this is at most n_r, which is the number of families of requirement r.

With the same reasoning, the total number of families matched is

$$\sum_{r \in [r_{\max}]} \frac{\rho}{r}(\underline{b}_r + \bar{b}_r) + \sum_{\substack{\tau^P \in \mathcal{T}^P \\ c^F \in C^F}} c^F[r] x(\tau^P, c^F),$$

which is exactly the value of our solution to ILP_1, and thus equals u^*. ◁

Claim 5. *If there is a feasible assignment that matches u^* many agents, then ILP_1 admits a solution of value u^*.*

Proof. Let σ be a feasible assignment that matches u^* many agents.

We start by initializing $\underline{b}_r := 0$ and $\bar{b}_r := 0$ for every $r \in [r_{\max}]$. For each $r \in [r_{\max}]$, we also construct sets A^ρ_r, $\bar{A}^\rho_r$ and $A_r(c^F)$ for each $c^F \in C^F$; each of these sets will contain families with requirement r. The set A^ρ_r will contain families that are matched to compulsory homogeneous ρ-blocks, $\bar{A}^\rho_r$ the families matched to optional homogeneous ρ-blocks, and $A_r(c^F)$ will contain families that are part of the configurations.

For each place $p_j \in P$, choose an arbitrary subset of families $F^1_j \subseteq \sigma^{-1}(p_j)$ such that $\sum_{f_i \in F^1_j} \boldsymbol{r}_i \geq \underline{\boldsymbol{c}}_j$ and moreover, for each family $f_{i^*} \in F^1_j$ we have $\sum_{f_i \in F^1_j \setminus \{f_{i^*}\}} \boldsymbol{r}_i < \underline{\boldsymbol{c}}_j$; that is, the families in F^1_j satisfy the lower quota of p_j, and removing any family from F^1_j means the lower quota is no longer satisfied. Since σ is feasible, such a set must exist. Let $d := \sum_{f_i \in F^1_j} \boldsymbol{r}_i - \underline{\boldsymbol{c}}_j$; this is the amount by which the total requirement of the families in F^1_j exceeds $\underline{\boldsymbol{c}}_j$. Observe that by construction, $0 \leq d \leq r_{\max} - 1$.

If $\sum_{f_i \in F^1_j} \boldsymbol{r}_i - d > r_{\max}(\rho - 1)$, then we start an iteration as follows. Clearly, $\sum_{f_i \in F^1_j} \boldsymbol{r}_i > r_{\max}(\rho - 1)$ holds, so by Observation 2, there must be a homogeneous ρ-block F'_j contained in F^1_j. Let the requirement of the families in F'_j be $r \in [r_{\max}]$. We create a new subset $F^2_j := F^1_j \setminus F'_j$ and increment the value of the variable $\underline{b}_r$ by one. We also add the families in F'_j to A^ρ_r. Observe that there are precisely $\frac{\rho}{r}$ families in F'_j. We repeat this process until we reach k such that $\sum_{f_i \in F^k_j} \boldsymbol{r}_i - d \leq r_{\max}(\rho - 1)$.

Since $\sum_{f_i \in F^1_j} \boldsymbol{r}_i - d = \underline{\boldsymbol{c}}_j$ by construction, we stop when $\sum_{f_i \in F^k_j} \boldsymbol{r}_i - d \leq r_{\max}(\rho - 1)$, and in each iteration we remove ρ from the total requirement, we observe that the number k of iterations is precisely $\arg\operatorname{res}(\underline{\boldsymbol{c}}_j)$, and $\sum_{f_i \in F^k_j} \boldsymbol{r}_i - d = \operatorname{res}(\underline{\boldsymbol{c}}_j)$. Therefore we increment the variables $\{\underline{b}_r : r \in [r_{\max}]\}$ precisely $\sum_{p_j \in P} \arg\operatorname{res}(\underline{\boldsymbol{c}}_j)$ times. This shows the first part of Constraint (3).

Similarly, consider $\bar{F}^1_j := \sigma^{-1}(p_j) \setminus F^1_j$. Intuitively, these families are assigned to p_j but they are not necessary for satisfying the lower quota of p_j.

If $\sum_{f_i \in \bar{F}^1_j} \boldsymbol{r}_i - r_{\max} + d + 1 > r_{\max}(\rho - 1)$, then we start a second iteration as follows. Clearly, $\sum_{f_i \in \bar{F}^1_j} \boldsymbol{r}_i > r_{\max}(\rho - 1)$ holds, so by Observation 2, there must be a homogeneous ρ-block F'_j contained in $\bar{F}^1_j$. Let the requirement of the families in F'_j be $r \in [r_{\max}]$. We create a new subset $\bar{F}^2_j := \bar{F}^1_j \setminus F'_j$ and increment the value of the variable $\bar{b}_r$ by one. We repeat this process until we reach k' such that $\sum_{f_i \in \bar{F}^{k'}_j} \boldsymbol{r}_i - r_{\max} + d + 1 \leq r_{\max}(\rho - 1)$.

Observe that by the definition of d,

$$\begin{aligned}\sum_{f_i \in \bar{F}^1_j} \boldsymbol{r}_i - r_{\max} + d + 1 &= \sum_{f_i \in \bar{F}^1_j} \boldsymbol{r}_i + \sum_{f_i \in F^1_j} \boldsymbol{r}_i - \underline{\boldsymbol{c}}_j - r_{\max} + 1 \\ &\leq \bar{\boldsymbol{c}}_j - \underline{\boldsymbol{c}}_j - r_{\max} + 1.\end{aligned}$$

If $\bar{\boldsymbol{c}}_j - \underline{\boldsymbol{c}}_j \geq r_{\max} - 1$, then

$$k' - 1 \leq \arg\operatorname{res}(\bar{\boldsymbol{c}}_j - \underline{\boldsymbol{c}}_j - r_{\max} + 1),$$

and thus

$$\sum_{f_i \in \bar{F}^k_j} \boldsymbol{r}_i - r_{\max} + d + 1 \leq \operatorname{res}(\bar{\boldsymbol{c}}_j - \underline{\boldsymbol{c}}_j - r_{\max} + 1).$$

By contrast, if $\bar{\boldsymbol{c}}_j - \underline{\boldsymbol{c}}_j < r_{\max} - 1$, then we never remove homogeneous ρ-blocks from p_j, and $\bar{F}^{k'}_j = \bar{F}^1_j$. Thus, we increment the variables $\{\bar{b}_r : r \in [r_{\max}]\}$ throughout the whole process performed for each place $p_j \in P$ at most

$$\sum_{p_j \in P} \arg\operatorname{res}(\max(\underline{\boldsymbol{c}}_j - \bar{\boldsymbol{c}}_j - r_{\max} + 1, 0))$$

times, which shows the second part of Constraint (3).

After finishing the two iterations for finding F^k_j and $\bar{F}^{k'}_j$, we identify the configuration $c^F_j \in C^F$ whose r-th coordinate satisfies

$$c^F_j[r] = |\{f_h \in F^k_j \cup \bar{F}^{k'}_j : \boldsymbol{r}_h = r\}|$$

for each $r \in [r_{\max}]$. We then add to $A_r(c^F_j)$ the $c^F_j[r]$ families in the set $\{f_h \in F^k_j \cup \bar{F}^{k'}_j : \boldsymbol{r}_h = r\}$. By construction, $\sum_{f_i \in F^k_j \cup \bar{F}^{k'}_j} \boldsymbol{r}_i = \sum_{r \in [r_{\max}]} c^F_j[r] \cdot r$. Let τ^P_j be the type of p_j. We increment the variable $x(\tau^P_j, c^F_j)$ by one. We now show that τ^P_j is suitable for c^F_j.

Recall that $\sum_{f_i \in F^k_j} \boldsymbol{r}_i - d = \operatorname{res}(\underline{\boldsymbol{c}}_j) = \tau^P_j[1]$ for each $p_j \in P$. Therefore,

$$\tau^P_j[1] = \sum_{f_i \in F^k_j \cup \bar{F}^{k'}_j} \boldsymbol{r}_i - d \leq \sum_{f_i \in F^k_j \cup \bar{F}^{k'}_j} \boldsymbol{r}_i \leq \sum_{r \in [r_{\max}]} c^F[r] \cdot r,$$

as required. It remains to show that

$$\sum_{r \in [r_{\max}]} c^F_j[r] \cdot r \leq \tau^P[1] + \tau^P[2].$$

We distinguish between two cases.

Case 1: $\bar{\boldsymbol{c}}_j - \underline{\boldsymbol{c}}_j \geq r_{\max} - 1$. Then

$$\sum_{f_i \in \bar{F}_j^{k'}} \boldsymbol{r}_i + d \leq \text{res}(\bar{\boldsymbol{c}}_j - \underline{\boldsymbol{c}}_j - r_{\max} + 1) + r_{\max} - 1 = \tau^P[2].$$

Therefore,

$$\begin{aligned}\sum_{r \in [r_{\max}]} c^F[r] \cdot r &= \sum_{f_i \in F_j^k \cup \bar{F}_j^{k'}} \boldsymbol{r}_i = \sum_{f_i \in F_j^k} \boldsymbol{r}_i - d + \sum_{f_i \in \bar{F}_j^{k'}} \boldsymbol{r}_i + d \\ &\leq \text{res}(\underline{\boldsymbol{c}}_j) + \text{res}(\bar{\boldsymbol{c}} - \underline{\boldsymbol{c}}_j - r_{\max} + 1) + r_{\max} - 1 \\ &= \tau_j^P[1] + \tau_j^P[2],\end{aligned}$$

as required.

Case 2: $\bar{\boldsymbol{c}}_j - \underline{\boldsymbol{c}}_j < r_{\max} - 1$. In this case, $\tau_j^P[2] = \bar{\boldsymbol{c}}_j - \underline{\boldsymbol{c}}_j$. Recall that σ is a valid assignment, so $\sum_{f_i \in F_j^1} \boldsymbol{r}_i + \sum_{f_i \in \bar{F}_j^1} \boldsymbol{r}_i \leq \bar{\boldsymbol{c}}_j$. Therefore we obtain that

$$\sum_{f_i \in \bar{F}_j^{k'}} \boldsymbol{r}_i = \sum_{f_i \in \bar{F}_j^1} \boldsymbol{r}_i \leq \bar{\boldsymbol{c}}_j - \sum_{f_i \in F_j^1} \boldsymbol{r}_i = \bar{\boldsymbol{c}}_j - \underline{\boldsymbol{c}}_j - d = \tau_j^P[2] - d$$

which is equivalent to

$$\sum_{f_i \in \bar{F}_j^{k'}} \boldsymbol{r}_i + d \leq \tau_j^P[2].$$

Thus

$$\begin{aligned}\sum_{r \in [r_{\max}]} c^F[r] \cdot r &= \sum_{f_i \in F_j^k \cup \bar{F}_j^{k'}} \boldsymbol{r}_i = \sum_{f_i \in F_j^k} \boldsymbol{r}_i - d + \sum_{f_i \in \bar{F}_j^{k'}} \boldsymbol{r}_i + d \\ &\leq \tau_j^P[1] + \tau_j^P[2],\end{aligned}$$

as required.

For each place type τ^P, every place of type τ^P increments a single variable in $\{x(\tau^P, c^F) : c^F \in C^F, c^F \text{ is suitable for } \tau^P\}$. Thus $\sum\{x(\tau^P, c^F) : c^F \in C^F, c^F \text{ is suitable for } \tau^P\} = m_{\tau^P}$, which proves Constraint (2).

Next, let us prove Constraint (4). Let $r \in [r_{\max}]$ be a service requirement. Observe that the sets $A_r^\rho, \bar{A}_r^\rho$, and $A_r(c^F)$ for $c^F \in C^F$ are all subsets of $\{f_h \in F : \boldsymbol{r}_h = r\}$ as only families of requirement r are added to these sets. Moreover, they are pairwise disjoint, as any family is added to at most one set. Finally, the set of families matched to some place under σ is precisely $A_r^\rho \cup \bar{A}_r^\rho \cup \bigcup_{c^F \in C^F} A_r(c^F)$, as every family that is matched to a place is added to one of these sets.

Each time we incremented variable $\underline{b}_r$ by one, we added $\frac{\rho}{r}$ families to A_r^ρ. Thus $|A_r^\rho| = \frac{\rho}{r}\underline{b}_r$. Similarly, each time we incremented variable $\bar{b}_r$ by one, we added $\frac{\rho}{r}$ families to $\bar{A}_r^\rho$. Thus $|\bar{A}_r^\rho| = \frac{\rho}{r}\bar{b}_r$. Finally, for every $c^F \in C^F$, anytime we incremented a variable in $\{x(\tau^P, c^F) : \tau^P \in \mathcal{T}^P\}$ by one, we added $c^F[r]$ families to $A_r(c^F)$. Thus $|A_r(c^F)| = \sum_{\tau^P \in \mathcal{T}^P} c^F[r] x(\tau^P, c^F)$.

Therefore we can conclude that the number of families of requirement r matched under σ is precisely

$$\begin{aligned}&|A_r^\rho| + |\bar{A}_r^\rho| + \sum_{c^F \in C^F} |A_r(c^F)| \\ &= \frac{\rho}{r}\underline{b}_r + \frac{\rho}{r}\bar{b}_r + \sum_{c^F \in C^F} \sum_{\tau^P \in \mathcal{T}^P} c^F[r] x(\tau^P, c^F) \\ &= \frac{\rho}{r}(\underline{b}_r + \bar{b}_r) + \sum_{\substack{\tau^P \in \mathcal{T}^P \\ c^F \in C^F}} c^F[r] x(\tau^P, c^F).\end{aligned}$$

Since there are n_r families of requirement r, we get

$$\frac{\rho}{r}(\underline{b}_r + \bar{b}_r) + \sum_{\substack{\tau^P \in \mathcal{T}^P \\ c^F \in C^F}} c^F[r] x(\tau^P, c^F) \leq n_r.$$

This proves Constraint (4).

We deduce that the number of families matched under σ is

$$\sum_{r \in [r_{\max}]} \frac{\rho}{r}(\underline{b}_r + \bar{b}_r) + \sum_{\substack{\tau^P \in \mathcal{T}^P \\ c^F \in C^F}} c^F[r] x(\tau^P, c^F).$$

Because σ matches u^* many families, the value of the constructed solution for ILP_1 is exactly u^*. ◁

Claims 4 and 5 together prove the correctness of our ILP. □

B.4 Proof of Theorem 2

Theorem 2 (★). *P*ARETO-RR *and* MAXUTIL-RR *for $t = 1$ are NP-hard even when $r_{\max} = c_{\max} = 2$ and there are no lower quotas. The result holds for* PARETO-RR *even if all families have dichotomous preferences and find exactly two places acceptable, and for* MAXUTIL-RR *even if utilities are binary and each family has positive utilities for exactly two places.*

Proof. Here we present a reduction from the variant of 3-SAT where each literal appears exactly twice. Let our input be a formula $\varphi = \bigwedge_{C \in \mathcal{C}} C$ over a set X of variables where both literals x and $\overline{x}$ appear exactly twice in φ for each $x \in X$, and each clause $C \in \mathcal{C}$ contains exactly three distinct literals. This problem is NP-hard [10]. We construct an instance I of PARETO-RR with $t = 1$ and without lower quotas as follows.

We set $\{f_x, x^1, x^2, \overline{x}^1, \overline{x}^2 : x \in X\}$ as the set of families; we set the requirement of all families in $F_X := \{f_x : x \in X\}$ as two, and the requirement of all remaining families as one. We introduce two places, p_x and $p_{\overline{x}}$, for each variable $x \in X$ and a place p_C for each clause $C \in \mathcal{C}$. We set the upper quota of every place as two. Note that $r_{\max} = c_{\max} = 2$, as promised.

We next define the set of acceptable places for each family. First, for each literal ℓ (that is, for each $\ell \in \{x, \overline{x} : x \in X\}$), the family ℓ^h for some $h \in [2]$ corresponds to the h-th occurrence of the literal ℓ, and hence finds two places acceptable: p_ℓ and $p_{C(\ell,h)}$ where $C(\ell, h)$ is the clause containing the h-th occurrence of literal ℓ. Second, for each variable $x \in X$, the family f_x finds p_x and $p_{\overline{x}}$ acceptable. We set dichotomous preferences for all families, so each family is indifferent between the two places they find acceptable.

Claim 6. *I admits a feasible, acceptable and complete assignment if and only if φ is satisfiable.*

Proof. Suppose first that φ admits a satisfying truth assignment. We define an assignment σ for I as follows. For each variable $x \in X$ set to `true`, the family f_x is assigned to the place $p_{\overline{x}}$, the two families $\overline{x}^1$ and $\overline{x}^2$ that correspond to false literals are assigned to the places $p_{C(\overline{x},1)}$ and $p_{C(\overline{x},2)}$, respectively, while the two families x^1 and x^2 that correspond to true literals are assigned to p_x. Similarly, For each variable $y \in X$ set to `false`, the family f_y is assigned to the place p_y, the two families y^1 and y^2 that correspond to false literals are assigned to the places $p_{C(y,1)}$ and $p_{C(y,2)}$, respectively, while the two families $\overline{y}^1$ and $\overline{y}^2$ that correspond to true literals are assigned to $p_{\overline{y}}$. Observe that σ never assigns families with a total requirement more than two to a place, because each clause contains at most two literals that evaluate to `false`, and thus each place p_C, $C \in \mathcal{C}$ accommodates at most two families under σ, both with requirement one. Therefore, σ is feasible, and it is straightforward to verify that it is complete and acceptable as well.

Suppose now that σ is an assignment for I that is feasible, acceptable, and complete. We set each variable $x \in X$ to `true` if and only if $\sigma(f_x) = p_{\overline{x}}$. In other words, we set a literal ℓ to `false` exactly if the place p_ℓ is occupied by a family in F_X. To show that each clause is satisfied, let us assume for the sake of contradiction that we set all three literals in some clause C to `false`. This means that for each literal ℓ appearing in C, the place p_ℓ accommodates a family in F_X. Since each such family has requirement two, which equals the upper quota of p_ℓ, we get that σ must assign the families ℓ^1 and ℓ^2 elsewhere due to its feasibility. Since σ respects acceptability and is complete, it must assign ℓ^1 and ℓ^2 to the clauses $C(\ell, 1)$ and $C(\ell, 2)$, respectively. In particular, the three families that correspond to the literals appearing in C (that is, the families $\ell_1^{h_1}, \ell_2^{h_2}$, and $\ell_3^{h_3}$ for which $C(\ell_j, h_j) = C$ for $j \in [3]$) can only be assigned to p_C, contradicting the feasibility of σ. This shows that the constructed truth assignment satisfies φ, and hence, the claim holds. $\triangleleft$

The result for PARETO-RR follows from Claim 6 due to Observation 3, and from that, the NP-hardness for MAXUTIL-RR follows from Observation 1. $\square$

B.5 *Proof of Claim 2*

Claim 2 ($\star$). *Suppose that $i \in [n-1]$, and let $\sigma_i : F_i \to P$ denote an optimal allocation for I_i. Then there exists an optimal allocation $\sigma_{i+1} : F_{i+1} \to P$ for I_{i+1} such that $\Delta(\sigma_i, \sigma_{i+1}) \le m \cdot \rho_m^\star$.*

Proof. It will be useful for us to define an integer ρ_j recursively for each $j \in [m]$ by setting

$$\rho_j = \begin{cases} r_{\max}(\rho - 1) & \text{for } j = 1 \\ \rho_{j-1} \cdot (m-1) + \rho + r_{\max} & \text{for } j = 2, \dots, m. \end{cases} \tag{6}$$

It is straightforward to verify by simple calculus that $\rho_m \le \rho_m^\star$ (relying also on our assumption that $m \ge 2$).

Let σ_{i+1} be an optimal allocation for I_{i+1} that minimizes $\Delta(\sigma_i, \sigma_{i+1})$. Consider the the movement of families when the allocation changes from σ_i to σ_{i+1}; we will refer to this as *the relocation*. For two distinct places p_j and $p_{j'}$, let $d(p_j, p_{j'})$ be the total requirement of all families moving from p_j into $p_{j'}$ under the relocation. Define also $d^{\text{in}}(p_j) := \sum_{p_{j'} \in P \setminus \{p_j\}} d(p_{j'}, p_j)$ as the total requirement of families in F_i moving into p_j under the relocation. As both σ_i and σ_{i+1} are complete, we know $\Delta(\sigma_i, \sigma_{i+1}) = \sum_{p_j \in P} d^{\text{in}}(p_j)$.

We will prove that $d^{\text{in}}(p_j) \le \rho_m$ for each $p_j \in P$. From this $\Delta(\sigma_i, \sigma_{i+1}) = \sum_{p_j \in P} d^{\text{in}}(p_j) \le m\rho_m \le m\rho_m^\star$ follows, proving the claim. Assume for the sake of contradiction that there exists some $p_{j_0} \in P$ with $d^{\text{in}}(p_{j_0}) > \rho_m$.

An acyclic auxiliary graph G_ρ. We proceed with defining an auxiliary digraph G_ρ defined over P in which $(p_j, p_{j'})$ is an arc if and only if $d(p_j, p_{j'}) > \rho_1$; recall that $\rho_1 = r_{\max}(\rho - 1)$. Notice that by Observation 2, for each arc e in G_ρ we know that the set of families moving from the tail of e to the head of e under the relocation contains a homogeneous ρ-block; let B_e be such a homogeneous ρ-block corresponding to e.

We claim that G_ρ is acyclic. Suppose for the sake of contradiction that a set C of arcs forms a directed cycle in G_ρ. Consider the assignment σ'_{i+1} for I_{i+1} obtained from σ_{i+1} by moving all families contained in $F_C := \cup_{e \in C} B_e$ to the place they were assigned under σ_i, i.e., "backward" along the cycle C. Since all homogeneous ρ-blocks have the same total requirement ρ, the assignment σ'_{i+1} is feasible for I_{i+1}. Moreover, since $\Delta(\sigma_i, \sigma'_{i+1}) < \Delta(\sigma_i, \sigma_{i+1})$, we know that $\text{util}(\sigma'_{i+1}) < \text{util}(\sigma_{i+1})$, due to our choice of σ_{i+1}. This means

$$\sum_{f_h \in F_C, p_j = \sigma_i(f_h)} \boldsymbol{u}_h[j] < \sum_{f_h \in F_C, p_j = \sigma_{i+1}(f_h)} \boldsymbol{u}_h[j]. \tag{7}$$

Let us now construct an assignment σ'_i from σ_i by moving all families contained in F_C to the place they are assigned under σ_{i+1}, i.e., "forward" along the cycle C. Again, σ'_i is feasible for I_i. Moreover, by (7) we know that $\text{util}(\sigma'_i) > \text{util}(\sigma_i)$, which contradicts the optimality of σ_i. We obtain that G_ρ is indeed acyclic.

Finding a path $\mathcal{P}^\star$ in G_ρ. Let $P_\dashv$ denote the set of places with *free capacity* at least ρ under σ_i, i.e. $P_\dashv = \{p_j : \bar{\boldsymbol{c}}_j \ge \rho + \text{load}(p_j, \sigma_i)\}$. We define $P_\vdash$ analogously, to contain places with free capacity at least ρ under σ_{i+1}, so $P_\vdash = \{p_j : \bar{\boldsymbol{c}}_j \ge \rho + \text{load}(p_j, \sigma_{i+1})\}$. We are going to construct a path from $P_\vdash$ to $P_\dashv$ in G_ρ.

Define a *chain* from p_{j_0} to $P_\dashv$ as a sequence $p_{j_0}, p_{j_1}, \dots, p_{j_{\widetilde{m}}}$ such that (i) $p_{j_{\widetilde{m}}} \in P_\dashv$, and (ii) $d(p_{j_{h-1}}, p_{j_h}) > \rho_{m-h}$ for each integer h with $0 < h \le \widetilde{m}$. We build such a chain by induction. We start from the sequence containing only p_{j_0}, and maintain condition (ii) as an invariant; note that (ii) holds trivially for the sequence p_{j_0}. So suppose that we have already built a sequence $p_{j_0}, p_{j_1}, \dots, p_{j_h}$ for which condition (ii) holds. If $p_{j_h} \in P_\dashv$, then we are done, as the sequence fulfills condition (i) as well, and thus is a chain. If $p_{j_h} \notin P_\dashv$ then, by the definition of $P_\dashv$, the free capacity of p_{j_h} under σ_i is less than ρ. This implies that the families moving out from p_{j_h} under the relocation must have total requirement greater than $d^{\text{in}}(p_{j_h}) - \rho$ (as otherwise the families moving into p_{j_h} during the relocation would not fit). Since these families must move into some place in $P \setminus \{p_{j_h}\}$, there must exist some $p_{j_{h+1}} \in P \setminus \{p_{j_h}\}$ for which

$$d(p_{j_h}, p_{j_{h+1}}) \ge \frac{d^{\text{in}}(p_{j_h}) - \rho}{m-1} > \frac{\rho_{m-h} - \rho}{m-1} > \rho_{m-h-1}$$

where the second inequality follows from condition (ii) if $h > 0$, and from the assumption that $d^{\text{in}}(p_{j_0}) > \rho_m$ in the case $h = 0$; the third equality follows from the definition of ρ_{m-h} which satisfies

$$\frac{\rho_{m-h} - \rho - r_{\max}}{m-1} = \rho_{m-h-1}.$$

Hence, we can pick $p_{j_{h+1}}$ as the next place in the chain, since the sequence $p_{j_0}, p_{j_1}, \dots, p_{j_h}, p_{j_{h+1}}$ fulfills condition (ii). Since G_ρ is acyclic and $|P|$ is finite, the existence of a chain from p_{j_0} to $P_\dashv$ follows.

Next, we similarly build a *back-chain* from $P_\vdash$ to $p_{j'_0} = p_{j_0}$, which is defined as a sequence $p_{j'_{\widetilde{m}'}}, \dots, p_{j'_1}, p_{j'_0}$ such that (i') $p_{j'_{\widetilde{m}'}} \in P_\vdash$, and (ii') $d(p_{j'_h}, p_{j'_{h-1}}) > \rho_{m-h}$ for each integer h with $0 < h \le \widetilde{m}'$. Because $d^{\text{in}}(p_{j_0}) > \rho_m$, we know that there exists some place $p_{j'_1}$ for which $d(p_{j'_1}, p_{j_0}) \ge d^{\text{in}}(p_{j_0})/(m-1) > \rho_m/(m-1) > \rho_{m-1}$. We build our back-chain starting from the sequence $p_{j'_1}, p_{j_0}$ by induction, using the same technique we applied to build our chain in the previous paragraph.

Namely, suppose that we already have a sequence $p_{j'_h}, \dots, p_{j'_1}, p_{j_0}$ for which condition (ii') holds. If $p_{j'_h} \in P_\vdash$, then we are done, as the sequence fulfills condition (i') as well, and thus is a back-chain. If $p_{j'_h} \notin P_\vdash$ then, by the definition of $P_\vdash$, the free capacity of $p_{j'_h}$ under σ_{i+1} is less than ρ.

However, $d(p_{j'_h}, p_{j'_{h-1}}) > \rho_{m-h}$ by condition (ii'); therefore, taking into account that f_{i+1} might be assigned to $p_{j'_h}$ by σ_{i+1}, we obtain that the total requirement of families moving into $p_{j'_h}$ under the relocation (recall that this excludes f_{i+1}) is $d^{\text{in}}(p_{j'_h}) > d(p_{j'_h}, p_{j'_{h-1}}) - \rho - \boldsymbol{r}_{i+1} > \rho_{m-h} - \rho - r_{\max}$. Since these families must have come from some place in $P \setminus \{p_{j'_h}\}$, we know that there exists some place $p_{j'_{h+1}} \in P$ for which

$$d(p_{j'_{h+1}}, p_{j'_h}) \geq \frac{d^{\text{in}}(p_{j'_h})}{m-1} > \frac{\rho_{m-h} - \rho - r_{\max}}{m-1} = \rho_{m-h-1}.$$

Hence, we can pick $p_{j'_{h+1}}$ as the next place in the back-chain, because the sequence $p_{j'_{h+1}}, p_{j'_h}, \dots, p_{j'_1}, p_{j_0}$ fulfills condition (ii'). Since G_ρ is acyclic and $|P|$ is finite, the existence of a back-chain from $P_\vdash$ to p_{j_0} follows.

Consider the sequence $\mathcal{P}^\star$ of places obtained by concatenating our back-chain from $P_\vdash$ to p_{j_0} with the chain from p_{j_0} to $P_\dashv$. Observe that by conditions (ii) and (ii'), there is an arc in G_ρ from each place in $\mathcal{P}^\star$ to the next place in $\mathcal{P}^\star$. Since G_ρ is acyclic, this means that $\mathcal{P}^\star$ forms a path in G_ρ.

The contradiction implied by our path $\mathcal{P}^\star$. It remains to show that the existence of our path $\mathcal{P}^\star$ from $P_\vdash$ to $P_\dashv$ in G_ρ leads to a contradiction. Let $E(\mathcal{P}^\star)$ denote the set of arcs on this path, and define $F_{\mathcal{P}^\star} = \bigcup_{e \in E(\mathcal{P}^\star)} B_e$, that is, $F_{\mathcal{P}^\star}$ is the union of homogeneous ρ-blocks corresponding to the arcs on $\mathcal{P}^\star$. Define the assignment σ'_{i+1} for I_{i+1} obtained from σ_{i+1} by moving all families contained in $F_{\mathcal{P}^\star}$ to the place they were assigned under σ_i, i.e., "backward" along the path $\mathcal{P}^\star$. Since all homogeneous ρ-blocks have the same total requirement ρ and, in addition, the first place on $\mathcal{P}^\star$ belongs to $P_\vdash$ and thus has free capacity at least ρ under σ_{i+1} that can be used to accommodate the superblock corresponding to the first arc of $\mathcal{P}^\star$, we get that σ'_{i+1} is feasible for I_{i+1}. Moreover, since $\Delta(\sigma_i, \sigma'_{i+1}) < \Delta(\sigma_i, \sigma_{i+1})$, we know that $\text{util}(\sigma'_{i+1}) < \text{util}(\sigma_{i+1})$ due to our choice of σ_{i+1}, which yields

$$\sum_{f_h \in F_{\mathcal{P}^\star}, p_j = \sigma_i(f_h)} \boldsymbol{u}_h[j] < \sum_{f_h \in F_{\mathcal{P}^\star}, p_j = \sigma_{i+1}(f_h)} \boldsymbol{u}_h[j]. \tag{8}$$

Let us now construct an assignment σ'_i from σ_i by moving all families contained in $F_{\mathcal{P}^\star}$ to the place they are assigned under σ_{i+1}, i.e., "forward" along the path $\mathcal{P}^\star$. Again, σ'_i is feasible for I_i, because the last place on $\mathcal{P}^\star$ belongs to $P_\dashv$. Due to (8), $\text{util}(\sigma'_i) > \text{util}(\sigma_i)$, which contradicts to the optimality of σ. This contradiction proves the claim. $\square$

B.6 *Proof of Claim 3*

Claim 3 (★)**.** *Suppose that $q \in [\underline{c}_\Sigma]$, and let $\hat{\sigma}_q : F \to P$ denote an optimal allocation for $\hat{I}_q$. Then there exists an optimal allocation $\hat{\sigma}_{q+1} : F \to P$ for $\hat{I}_{q+1}$ such that $\Delta(\hat{\sigma}_q, \hat{\sigma}_{q+1}) \leq m \cdot \rho^\star_m$.*

Proof. The proof is similar to the proof of Claim 2. Let $\underline{\boldsymbol{c}}_j$ and $\underline{\boldsymbol{c}}'_j$ denote the lower quotas given for some location $p_j \in P$ in $\hat{I}_q$ and in $\hat{I}_{q+1}$, respectively; then $\underline{\boldsymbol{c}}_j = \underline{\boldsymbol{c}}'_j$ holds for each place p_j but one, and $\sum_{j \in [m]} \underline{\boldsymbol{c}}'_j = 1 + \sum_{j \in [m]} \underline{\boldsymbol{c}}_j$. Let $\hat{\sigma}_{q+1}$ be an optimal allocation for $\hat{I}_{q+1}$ that minimizes $\Delta(\hat{\sigma}_q, \hat{\sigma}_{q+1})$. Consider the situation where the allocation changes from $\hat{\sigma}_q$ to $\hat{\sigma}_{q+1}$; we will refer to this as *the relocation*.

For distinct places p_j and $p_{j'}$, let $\hat{d}(p_j, p_{j'})$ be the total requirement of all families moving from p_j into $p_{j'}$ under the relocation. Define also the values $\hat{d}^{\text{in}}(p_j) := \sum_{p_{j'} \in P \setminus \{p_j\}} \hat{d}(p_{j'}, p_j)$ and $\hat{d}^{\text{out}}(p_j) := \sum_{p_{j'} \in P \setminus \{p_j\}} \hat{d}(p_j, p_{j'})$ as the total requirement of families moving into p_j and out of p_j, respectively, under the relocation. As both $\hat{\sigma}_q$ and $\hat{\sigma}_{q+1}$ are complete assignments, we know that $\Delta(\hat{\sigma}_q, \hat{\sigma}_{q+1}) = \sum_{p_j \in P} \hat{d}^{\text{in}}(p_j) = \sum_{p_j \in P} \hat{d}^{\text{out}}(p_j)$.

Recall the definition of values ρ_h for $h \in [m]$ as defined by 6. We will prove that $\hat{d}^{\text{in}}(p_j) \leq \rho_m$ for each $p_j \in P$. From this $\Delta(\hat{\sigma}_q, \hat{\sigma}_{q+1}) = \sum_{p_j \in P} \hat{d}^{\text{in}}(p_j) \leq m \cdot \rho_m \leq m \cdot \rho^\star_m$ follows, implying the claim. Assume for the sake of contradiction that there exists some $p_{j_0} \in P$ with $\hat{d}^{\text{in}}(p_{j_0}) > \rho_m$.

An acyclic auxiliary graph $\hat{G}_\rho$. We proceed with defining an auxiliary digraph $\hat{G}_\rho$ defined over P in which $(p_j, p_{j'})$ is an arc exactly if $\hat{d}(p_j, p_{j'}) > \rho_1$. Notice that by Observation 2, for each arc e in $\hat{G}_\rho$ we know that the set of families moving from the tail of e to the head of e under the relocation contains a homogeneous ρ-block; let B_e be such a homogeneous ρ-block corresponding to e.

It is straightforward to see that the same argument used in the proof of Claim 2 for showing that G_ρ is acyclic can be applied to prove that $\hat{G}_\rho$ is acyclic. However, to find a path in $\hat{G}_\rho$ that leads to a contradiction, we need a more careful, somewhat different argument.

Finding a path $\hat{\mathcal{P}}^\star$ in G_ρ. Let $\hat{P}_\dashv$ contain those places $p_j \in P$ where

(a$_\dashv$) $\text{load}(p_j, \hat{\sigma}_q) \leq \bar{\boldsymbol{c}}_j - \rho$, and
(b$_\dashv$) $\text{load}(p_j, \hat{\sigma}_{q+1}) \geq \underline{\boldsymbol{c}}'_j + \rho$.

Analogously, let $\hat{P}_\vdash$ contain those places $p_j \in P$ where

(a$_\vdash$) $\text{load}(p_j, \hat{\sigma}_{q+1}) \leq \bar{\boldsymbol{c}}_j - \rho$, and
(b$_\vdash$) $\text{load}(p_j, \hat{\sigma}_q) \geq \underline{\boldsymbol{c}}_j + \rho$.

Define a *chain* from p_{j_0} to $\hat{P}_\dashv$ as a sequence $p_{j_0}, p_{j_1}, \dots, p_{j_{\tilde{m}}}$ such that (i) $p_{j_{\tilde{m}}} \in \hat{P}_\dashv$, and (ii) $\hat{d}(p_{j_{h-1}}, p_{j_h}) > \rho_{m-h}$ for each integer h with $0 < h \leq \tilde{m}$. We build such a chain by induction. We start from the sequence containing only p_{j_0}, and maintain condition (ii) as an invariant; note that (ii) holds trivially for the sequence p_{j_0}. So suppose that we have already built a sequence $p_{j_0}, p_{j_1}, \dots, p_{j_h}$ for which condition (ii) holds. If $p_{j_h} \in \hat{P}_\dashv$, then we are done, as the sequence fulfills condition (i) as well, and thus is a chain. If $p_{j_h} \notin \hat{P}_\dashv$, then either condition (a$_\dashv$) or condition (b$_\dashv$) does not hold for p_{j_h}.

First let us assume that p_{j_h} does not satisfy condition (a$_\dashv$). Then

$$\begin{aligned} \bar{\boldsymbol{c}}_{j_h} - \rho &< \text{load}(p_j, \hat{\sigma}_q) \\ &= \text{load}(p_j, \hat{\sigma}_{q+1}) - \hat{d}^{\text{in}}(p_{j_h}) + \hat{d}^{\text{out}}(p_{j_h}) \\ &\leq \bar{\boldsymbol{c}}_{j_h} - \hat{d}^{\text{in}}(p_{j_h}) + \hat{d}^{\text{out}}(p_{j_h}). \end{aligned}$$

From this, we get that

$$\hat{d}^{\text{out}}(p_{j_h}) > \hat{d}^{\text{in}}(p_{j_h}) - \rho. \tag{9}$$

Assume now that p_{j_h} violates condition (b$_\dashv$). Then

$$\begin{aligned} \underline{\boldsymbol{c}}_{j_h} + 1 \geq \underline{\boldsymbol{c}}'_{j_h} &> \text{load}(p_{j_h}, \hat{\sigma}_{q+1}) - \rho \\ &= \text{load}(p_{j_h}, \hat{\sigma}_q) + \hat{d}^{\text{in}}(p_{j_h}) - \hat{d}^{\text{out}}(p_{j_h}) - \rho \\ &\geq \underline{\boldsymbol{c}}_{j_h} + \hat{d}^{\text{in}}(p_{j_h}) - \hat{d}^{\text{out}}(p_{j_h}) - \rho, \end{aligned}$$

which implies

$$\hat{d}^{\text{out}}(p_{j_h}) > \hat{d}^{\text{in}}(p_{j_h}) - \rho - 1. \tag{10}$$

Hence, (10) holds in both cases.

Since the families moving out from p_{j_h} must move into some place in $P \setminus \{p_{j_h}\}$, there must exist some $p_{j_{h+1}} \in P \setminus \{p_{j_h}\}$ for which

$$\hat{d}(p_{j_h}, p_{j_{h+1}}) \geq \frac{\hat{d}^{\text{in}}(p_{j_h}) - \rho - 1}{m-1} > \frac{\rho_{m-h} - \rho - r_{\max}}{m-1} = \rho_{m-h-1} \tag{11}$$

where the second inequality follows from condition (ii) if $h > 0$, and from the assumption that $d^{\text{in}}(p_{j_0}) > \rho_m$ in the case $h = 0$. Hence, we can pick $p_{j_{h+1}}$ as the next place in the chain, as the sequence $p_{j_0}, p_{j_1}, \dots, p_{j_h}, p_{j_{h+1}}$ fulfills condition (ii). Since G_ρ is acyclic and $|P|$ is finite, the existence of a chain from p_{j_0} to $\hat{P}_\dashv$ follows.

Next, we similarly build a *back-chain* from $\hat{P}_\vdash$ to $p_{j'_0} = p_{j_0}$, which is a sequence $p_{j'_{\widetilde{m}'}}, \dots, p_{j'_1}, p_{j'_0}$ such that (i') $p_{j'_{\widetilde{m}'}} \in \hat{P}_\vdash$, and (ii') $\hat{d}(p_{j'_h}, p_{j'_{h-1}}) > \rho_{m-h}$ for each integer h with $0 < h \leq \widetilde{m}'$. Because $\hat{d}^{\text{in}}(p_{j_0}) > \rho_m$, there must exist some place $p_{j'_1} \in P \setminus \{p_{j_0}\}$ for which $\hat{d}(p_{j'_1}, p_{j_0}) \geq \hat{d}^{\text{in}}(p_{j_0})/(m-1) > \rho_m/(m-1) > \rho_{m-1}$. We build our back-chain starting from the sequence $p_{j'_1}, p_{j_0}$ by induction.

Suppose that we already have a sequence $p_{j'_h}, \dots, p_{j'_1}, p_{j_0}$ for which condition (ii') holds. If $p_{j'_h} \in \hat{P}_\vdash$, then we are done, as the sequence fulfills condition (i') as well, and thus is a back-chain. If $p_{j'_h} \notin \hat{P}_\vdash$, then either condition (a$_\vdash$) or condition (b$_\vdash$) fails for $p_{j'_h}$.

First, assume that condition (a$_\vdash$) fails for $p_{j'_h}$. Then

$$\begin{aligned} \bar{\mathbf{c}}_{j'_h} - \rho &< \text{load}(p_{j'_h}, \hat{\sigma}_{q+1}) \\ &= \text{load}(p_{j'_h}, \hat{\sigma}_q) + \hat{d}^{\text{in}}(p_{j'_h}) - \hat{d}^{\text{out}}(p_{j'_h}) \\ &\leq \bar{\mathbf{c}}_{j'_h} + \hat{d}^{\text{in}}(p_{j'_h}) - \hat{d}^{\text{out}}(p_{j'_h}), \end{aligned}$$

which implies

$$\hat{d}^{\text{in}}(p_{j'_h}) > \hat{d}^{\text{out}}(p_{j'_h}) - \rho. \tag{12}$$

Second, assume that condition (b$_\vdash$) fails for $p_{j'_h}$. Then

$$\begin{aligned} \underline{\mathbf{c}}_{j'_h} \leq \underline{\mathbf{c}}'_{j'_h} &\leq \text{load}(p_{j'_h}, \hat{\sigma}_{q+1}) \\ &= \text{load}(p_{j'_h}, \hat{\sigma}_q) + \hat{d}^{\text{in}}(p_{j'_h}) - \hat{d}^{\text{out}}(p_{j'_h}) \\ &< \underline{\mathbf{c}}_{j'_h} + \rho + \hat{d}^{\text{in}}(p_{j'_h}) - \hat{d}^{\text{out}}(p_{j'_h}) \end{aligned}$$

which again implies (12).

Since the families moving into $p_{j'_h}$ must have come from some place in $P \setminus \{p_{j'_h}\}$, there must exist some place $p_{j'_{h+1}} \in P \setminus \{p_{j'_h}\}$ for which

$$d(p_{j'_{h+1}}, p_{j'_h}) \geq \frac{d^{\text{in}}(p_{j'_h})}{m-1} > \frac{\rho_{m-j} - \rho}{m-1} > \rho_{m-j-1}.$$

Hence, we can pick $p_{j'_{h+1}}$ as the next place in the back-chain, because the sequence $p_{j'_{h+1}}, p_{j'_h}, \dots, p_{j'_1}, p_{j_0}$ fulfills condition (ii'). Since $\hat{G}_\rho$ is acyclic and $|P|$ is finite, the existence of a back-chain from $\hat{P}_\vdash$ to p_{j_0} follows.

Consider the sequence $\hat{\mathcal{P}}^\star$ of places obtained by concatenating our back-chain from $\hat{P}_\vdash$ to p_{j_0} with the chain from p_{j_0} to $\hat{P}_\dashv$. Observe that by conditions (ii) and (ii'), there is an arc in $\hat{G}_\rho$ from each place in $\hat{\mathcal{P}}^\star$ to the next place in $\hat{\mathcal{P}}^\star$. Since G_ρ is acyclic, this means that $\hat{\mathcal{P}}^\star$ forms a path in $\hat{G}_\rho$.

The contradiction implied by our path $\hat{\mathcal{P}}^\star$. It remains to show that the existence of our path $\hat{\mathcal{P}}^\star$ from $\hat{P}_\vdash$ to $P_\dashv$ in $\hat{G}_\rho$ leads to a contradiction. Let $E(\hat{\mathcal{P}}^\star)$ denote the set of arcs on this path, and define $F_{\hat{\mathcal{P}}^\star} = \bigcup_{e \in E(\hat{\mathcal{P}}^\star)} B_e$, that is, $F_{\hat{\mathcal{P}}^\star}$ is the union of homogeneous ρ-blocks corresponding to the arcs on $\hat{\mathcal{P}}^\star$. Define the assignment $\hat{\sigma}'_{q+1}$ for I_{i+1} obtained from $\hat{\sigma}_{q+1}$ by moving all families contained in $F_{\hat{\mathcal{P}}^\star}$ to the place they were assigned under $\hat{\sigma}_q$, i.e., "backward" along the path $\hat{\mathcal{P}}^\star$. Let us show that $\hat{\sigma}'_{q+1}$ is feasible for $\hat{I}_{q+1}$.

Since the first place $p_{p_{j'_{\widetilde{m}'}}}$ on $\hat{\mathcal{P}}^\star$ belongs to $P_\vdash$, by condition (a$_\vdash$) it has free capacity at least ρ under $\hat{\sigma}_{q+1}$ that can be used to accommodate the superblock corresponding to the first arc of $\hat{\mathcal{P}}^\star$, so the upper quota of $p_{p_{j'_{\widetilde{m}'}}}$ is not exceeded under $\hat{\sigma}'_{q+1}$. Since the last place $p_{p_{j_{\widetilde{m}}}}$ on $\hat{\mathcal{P}}^\star$ belongs to $P_\dashv$, removing a homogeneous ρ-block from the families assigned by $\hat{\sigma}_{q+1}$ to $p_{p_{j_{\widetilde{m}}}}$ does not violate its lower quota due to condition (b$_\dashv$). Since all homogeneous ρ-blocks have the same total requirement ρ, it also follows that the lower and upper quotas are respected by $\hat{\sigma}'_{q+1}$ for all remaining places as well. This proves that $\hat{\sigma}'_{q+1}$ is feasible for $\hat{I}_{q+1}$.

Moreover, since $\Delta(\hat{\sigma}_q, \hat{\sigma}'_{q+1}) < \Delta(\hat{\sigma}_q, \hat{\sigma}_{q+1})$, we know that the total utility of $\hat{\sigma}'_{q+1}$ is less than that of $\hat{\sigma}_{q+1}$, due to our choice of $\hat{\sigma}_{q+1}$, which yields

$$\sum_{f_h \in F_{\hat{\mathcal{P}}^\star}, p_j = \hat{\sigma}_q(f_h)} \mathbf{u}_h[j] < \sum_{f_h \in F_{\hat{\mathcal{P}}^\star}, p_j = \hat{\sigma}_{q+1}(f_h)} \mathbf{u}_h[j]. \tag{13}$$

Let us now construct an assignment $\hat{\sigma}'_q$ from $\hat{\sigma}_q$ by moving all families contained in $F_{\hat{\mathcal{P}}^\star}$ to the place they are assigned under $\hat{\sigma}_{q+1}$, i.e., "forward" along the path $\hat{\mathcal{P}}^\star$. Again, we can show that $\hat{\sigma}'_q$ is feasible for $\hat{I}_q$.

The last place on $\hat{\mathcal{P}}^\star$ belongs to $P_\dashv$, and therefore by condition (a$_\dashv$) can accommodate a homogeneous ρ-block besides the families assigned to it by $\hat{\sigma}_q$. The first place on $\hat{\mathcal{P}}^\star$ belongs to $P_\vdash$, and thus by condition (b$_\vdash$) we can remove a homogeneous ρ-block from among the families assigned to it by $\hat{\sigma}_q$ without violating its lower quota. Feasibility is therefore maintained at all places, as promised. Due to (13), the utility of $\hat{\sigma}'_q$ exceeds the utility of $\hat{\sigma}_q$, which contradicts the optimality of $\hat{\sigma}_q$. This contradiction proves the claim. □

B.7 *Proof of Theorem 4*

Theorem 4 (⋆). MAXUTIL- *and* PARETO-RR *for $t = 1$ are FPT w.r.t. u^*, the desired utility, if there are no lower quotas.*

Proof. Let I denote our input instance. We present an algorithm which, in polynomial time, produces an equivalent instance I' with at most $(u^*)^3$ families. Applying Proposition 6 to I' then yields fixed-parameter tractability for parameter u^*.

We start by checking whether there exists a family $f_i \in F$ and a place $p_j \in P$ such that p_j can accommodate f_i and $\mathbf{u}_i[j] \geq u^*$. If so, we output the assignment that assigns f_i to p_j and leaves every other family unassigned.

Otherwise, we proceed by greedily assigning families to places so that (i) the assignment remains feasible, and (ii) each family $f_i \in F$ assigned to some place $p_j \in P$ has positive utility for that place, i.e., $\mathbf{u}_i[j] \geq 1$. Let σ_0 be the feasible assignment obtained at the end of this greedy process.

If $\text{util}(\sigma_0) \geq u^*$, then we output σ_0. Otherwise, consider those places and families that are "relevant" according to σ_0, namely the sets $P_0 = \{p_j \in P : \sigma_0^{-1}(p_j) \neq \varnothing\}$ and $F_0 = \{f_i : \sigma_0(f_i) \neq \perp\}$. First, we *mark* each family in F_0. Next, for each $p_j \in P_0$ and each $\gamma \in [u^*]$, we order all families in $F_{j,\gamma} := \{f_i \in F : \mathbf{u}_i[j] = \gamma\}$

according to their requirement in a non-decreasing manner, and mark the first u^* families in this ordering (or all of them, if $|F_{j,\gamma}| < u^*$). We define an instance I' of MAXUTIL-RR by deleting all unmarked families.

Let F' be the set of families in I', that is, the families we have marked; we claim $|F'| \leq (u^*)^3$. First, due to condition (ii), we know that $|F_0| < u^*$ and $|P_0| < u^*$ follows from $\text{util}(\sigma_0) < u^*$. Additionally, we marked at most u^* families from $F_{j,\gamma}$ for each $p_j \in P_0$ and $\gamma \in [u^*]$. Summing this for all values of p_j and γ, we get at most $(u^* - 1)(u^*)^2$ families. Taking into account the at most u^* families in F_0, we get that there at most $(u^*)^3$ marked families, as promised.

Claim 7. *I' is equivalent with I.*

Proof. Clearly, a feasible assignment for I' is also feasible for I, and has the same utility in both instances. Suppose now that σ is a feasible assignment for I with $\text{util}(\sigma) \geq u^*$; we construct a feasible assignment for I' with utility at least u^* as follows.

First, for each family in F', we let σ' and σ coincide. Let $P^\star$ contain all places that have at least one family assigned to them by σ. Now, for each $p_j \in P^\star$ and each $\gamma \in [u^*]$, we take the ordering of $F_{j,\gamma}$ used in the marking process (recall that the families in $F_{j,\gamma}$ were ordered in a non-decreasing way according to their requirements), and we pick $n_{j,\gamma}$ families one by one from $F_{j,\gamma}$ among those that are not yet assigned to some place by σ', always picking the first family available, where

$$n_{j,\gamma} = \left|\{f_i \in F \setminus F' : f_i \in \sigma^{-1}(p_j), \boldsymbol{u}_i[j] = \gamma\}\right|.$$

We let σ' assign the picked families to p_j. This process stops once the number of assigned families reaches u^* or we have iterated through all places in $P^\star$ and all utility values $\gamma \in [u^*]$. We set $\sigma'(f_i) = \perp$ for all families of $f_i \in F'$ left unassigned thus far.

Let us show that for each $p_j \in P^\star$ and $\gamma \in [u^*]$, we are always able to pick a marked family from $F_{j,\gamma}$ during the above process. Assume first that we have marked the first u^* families from $F_{j,\gamma}$. Then due to our stopping condition, at each step there are less than u^* families in $F_{j,\gamma}$ that are already assigned by σ' to some place, so at each step when we pick the first available family from $F_{j,\gamma}$, we pick a marked family. Second, assume that $|F_{j,\gamma}| < u^*$, and thus all families in $F_{j,\gamma}$ are marked. In this case, there is no family in $F \setminus F'$ that has utility γ for p_j, due to the definition of $F_{j,\gamma}$. However, this implies $n_{j,\gamma} = 0$. This proves that all picked families are in F', and hence, σ' is an assignment for I'.

We next show that $\text{util}(\sigma') \geq u^*$. On one hand, this is clear if the algorithm stops because it has assigned at least u^* families, since all families contribute at least 1 to the total utility of σ'. On the other hand, if the algorithm stops because it has iterated over all possible places and utility values considered, then we know that at each place $p_j \in P^\star$ and for each $\gamma \in [u^*]$, there are exactly the same number of families assigned to p_j by σ and σ', due to our definition of $n_{j,\gamma}$. This implies

$$\begin{aligned}\text{util}(\sigma') &= \sum_{p_j \in P^\star} \sum_{f_i \in \sigma'^{-1}(p_j)} \boldsymbol{u}_i[j] \\ &= \sum_{p_j \in P^\star} \left(\sum_{f_i \in \sigma'^{-1}(p_j) \cap \sigma^{-1}(p_j)} \boldsymbol{u}_i[j] + \sum_{\gamma \in [u^*]} \gamma \cdot n_{j,\gamma} \right) \\ &= \sum_{p_j \in P^\star} \sum_{f_i \in \sigma^{-1}(p_j)} \boldsymbol{u}_i[j] = \text{util}(\sigma) \geq u^*.\end{aligned}$$

It remains to show that σ' is feasible. Consider some $p_j \in P^\star$. Clearly the families that both σ and σ' assign to p_j, i.e., those in $\sigma^{-1}(p_j) \cap F'$, contribute the same amount to the load of σ and σ'. Let $r_{j,\gamma}$ denote the maximum requirement of any marked family in $F_{j,\gamma}$. Consider the families in $\sigma^{-1}(p_j) \setminus F'$, and partition this set according to the utility values these families have for p_j. Consider some $\gamma \in [u^*]$. Since each family f_i in $\sigma^{-1}(p_j) \setminus F'$ with $\boldsymbol{u}_i[j] = \gamma$ is unmarked, it has requirement at least $r_{j,\gamma}$. By contrast, the $n_{j,\gamma}$ families assigned to p_j from $F_{j,\gamma}$ during the picking process are all marked, and thus have requirement at most $r_{j,\gamma}$. Hence, we get

$$\begin{aligned}\text{load}(p_j, \sigma') &= \sum_{f_i \in \sigma'^{-1}(p_j)} \boldsymbol{r}_i \\ &\leq \sum_{f_i \in \sigma'^{-1}(p_j) \cap \sigma^{-1}(p_j)} \boldsymbol{r}_i + \sum_{\gamma \in [u^*]} r_{j,\gamma} \cdot n_{j,\gamma} \\ &\leq \sum_{f_i \in \sigma^{-1}(p_j)} \boldsymbol{r}_i = \text{load}(p_j, \sigma).\end{aligned}$$

Thus, σ is a feasible assignment for I' with utility at least u^*, as required. ◁

The result for MAXUTILRR now follows from Claim 7 and Proposition 6, and the result for PARETO-RR follows by Observation 1.

We remark that it is possible to reduce the number of places as well: it suffices to keep (at most) u^* places for each family $f_i \in F'$ among those which can accommodate it: we need to pick them in non-decreasing order of f_i's utility for them. This yields a "pseudo-kernelization" algorithm for parameter u^* in the sense that it produces an equivalent instance where both the number of families and the number of places is bounded by a function of u^*; however, the requirement values and the upper quotas may be unbounded. ☐

Appendix C Additional material for Section 4

C.1 *Proof of Proposition 2*

Proposition 2 (⋆). *The following problems are NP-hard even if $c_{\max} = r_{\max} = 1$ and $m = 1$:*
- FEASIBLE-RR*;*
- PARETO-RR *with equal preferences;*
- MAXUTIL-RR *with equal utilities.*

Proof. We present a reduction from MULTICOLORED INDEPENDENT SET to FEASIBLE-RR. In this problem, we are given a graph $G = (V, E)$ and an integer k, with the vertex set of G partitioned into k sets $V_1, \ldots, V_k$; the task is to decide whether there exists an independent set of size k that contains one vertex from each set V_i, $i \in [k]$. This problem is NP-hard and, in fact, W[1]-hard when parameterized by k [31].[3]

We construct an instance of FEASIBLE-RR with a single location p as follows. We set V as the set of families, and $E \cup \{s_1, \ldots, s_k\}$ as the set of services, with p offering exactly one unit from every service. Each family $v \in V_i$ for some $i \in [k]$ requires one unit of each service associated with its incident edges, and one unit of s_i. We set the lower quota for p as 1 for each service s_i, $i \in [k]$, and as 0 for each service in E.

[3] Pietrzak dealt with the MULTICOLORED (or PARTITIONED) CLIQUE problem, but the claimed hardness results follow immediately from his results.

Notice that a feasible assignment assigns at least one vertex from each set V_i, $i \in [k]$, to p. Moreover, no two vertices assigned to p can be adjacent, as the edge connecting them corresponds to a service from which both of these two vertices (i.e., families) need one unit. Thus, a feasible assignment yields an independent set in G containing a vertex from each set V_i, $i \in [k]$. Vice versa, assigning such an independent set to p satisfies all lower and upper quotas, and thus yields a feasible assignment.

Finally, observe that adding arbitrary preferences or utilities to the constructed instance of FEASIBLE-RR we obtain an instance of PARETO- or MAXUTIL-RR respectively, that is equivalent with the original input instance. □

C.2 *Proof of Observation 3*

Observation 3 (⋆). *Given an instance I of* PARETO-RR *with dichotomous preferences, we can decide the existence of a feasible, acceptable and complete assignment for I by solving* PARETO-RR *on I.*

Proof. It suffices to observe that there exists a feasible, acceptable and complete (*fac*, for short) assignment if and only if every feasible, acceptable and Pareto-optimal (*faP*, for short) assignment is complete. Clearly, a fac assignment is necessarily Pareto-optimal, since families are indifferent between places that they find acceptable. On the other hand, if there exists a fac assignment σ, then each faP assignment must be complete, as otherwise σ would be a Pareto-improvement for it. □

C.3 *Proof of Proposition 3*

Proposition 3 (⋆). PARETO- *and* MAXUTIL-RR *are NP-hard even if $m = 2, r_{\max} = 1$, there are no lower quotas, and families have equal preferences or utilities.*

Proof. We present a reduction from INDEPENDENT SET to PARETO-RR that is similar to the proof of Proposition 2. Let (G, k) be our input instance with $G = (V, E)$.

We set V as the set of families, $E \cup \{s^\star\}$ as the set of services, and $P = \{p^\star, p\}$ as the set of places, with both places acceptable for each family. Place $p^\star$ offers exactly one unit of each service in E, and offers k units of service $s^\star$. Place p offers offers $|V| - k$ units of every service. Moreover, each family $v \in V$ requires one unit of each service associated with its incident edges, and one unit of service $s^\star$.

Note that p can accommodate an arbitrary set of $|V| - k$ families (but not more), while $p^\star$ can accommodate at most k families corresponding to an independent set. Thus, G admits an independent set of size k if and only if I admits a feasible and complete assignment. From this, the result for PARETO-RR follows by Observation 3.

By adding equal utilities to the constructed instance and setting the desired utility as $u^* = |V|$, we obtain an instance where an allocation reaches the desired utility if and only if it is complete. From this, the result for MAXUTIL-RR follows. □

C.4 *Proof of Theorem 5*

Theorem 5 (⋆). PARETO- *and* MAXUTIL-RR *are NP-hard even when $m = 3, c_{\max} = 1$, there are no lower quotas, and families have equal preferences or utilities, respectively.*

Proof. We present a reduction from 3-COLORING; let $G = (V, E)$ be the input graph. We create an instance I of PARETO-RR with equal preferences and without lower quotas as follows.

First, we set V as the set of families, E as the set of services, and $P = \{p_1, p_2, p_3\}$ as the set of places, with all places acceptable for each family. Each place offers exactly one unit of each service in E, and each family $v \in V$ requires one unit of each service associated with its incident edges.

Notice that a feasible assignment cannot assign two families corresponding to adjacent vertices to the same place, as the edge connecting them represents a service required by both of them. Thus, the families assigned to a given place must form an independent set in G.

We claim that a feasible and complete assignment for I exists if and only if G admits a proper 3-coloring. First, if σ is a complete and feasible assignment, then it yields a proper 3-coloring of G by the above reasoning. Second, if G is 3-colorable, then a 3-coloring of G yields a feasible and complete assignment, since there is no service from which a color class requires more than one unit. Since the families have equal, and hence, dichotomous preferences, by Observation 3 we can decide the 3-colorability of G by solving PARETO-RR on I.

To obtain NP-hardness for MAXUTIL-RR, we set equal utilities and $u^* = |V|$ as the desired utility, so that an assignment has utility at least u^* exactly if it is complete. The result then follows from the above reasoning. □

C.5 *Proof of Proposition 4*

Proposition 4 (⋆). PARETO-RR *for $m = 1$ is polynomial-time solvable if there are no lower quotas.*

Proof. Let $P = \{p_1\}$. The key observation is that a feasible and acceptable assignment σ is Pareto-optimal if and only if it *non-wasteful*, meaning that there exists no unassigned family $f_i \in F$ that finds p_1 acceptable and for which $\sigma^{-1}(p_1) \cup \{f_i\}$ can be accommodated at p_1. First, if σ is non-wasteful, then we cannot accommodate any more families in p_1 without disimproving some family, so σ is Pareto-optimal. Second, if σ is Pareto-optimal, then it must be non-wasteful as well: indeed, the existence of an unassigned family f_i that finds p_1 acceptable and for which $\sigma^{-1}(p_1) \cup \{f_i\}$ can be accommodated at p_1 implies that assigning f_i to p_1 alongside the families in $\sigma^{-1}(p_1)$ yields a Pareto-improvement over σ.

It remains to show that we can find a non-wasteful and feasible assignment in polynomial time. Start with an empty assignment and iterate over the families that find p_1 acceptable. If p_1 can accommodate such a family alongside everyone already assigned to p_1, assign the family to p_1. The resulting assignment is feasible and acceptable, because we never assign a family to a place it finds unacceptable. It is also non-wasteful, because if the place cannot accommodate a family f_i during the iteration when f_i is considered, it also cannot accommodate f_i at any later point during the algorithm. □

C.6 *Proof of Proposition 5*

Proposition 5 (⋆). PARETO-RR *is FPT w.r.t. the number of families with ties $n_\sim$, if there are no lower quotas.*

Proof. Our approach is to first apply serial dictatorship for each family whose preferences do not contain ties, and then try all possible assignments for the remaining families.

Let $F_\succ$ denote the set of those families in our input instance I whose preferences are a linear order over the subset of P they find acceptable (i.e., whose preference list does not contain ties). By re-indexing the set F, we may assume that $F_\succ = \{f_1, \dots, f_{n-n_\sim}\}$.

In the first phase of the algorithm, starting from an empty assignment, we iterate over $i = 1, \ldots, n - n_{\sim}$ and, at the i-th iteration, assign the family f_i to the place most preferred by f_i among all places in P that are acceptable for f_i and can accommodate f_i alongside the families already assigned to it. Let $\sigma_{\succ}$ denote the assignment obtained at the end of this iteration.

In the second phase of the algorithm, we delete the families $F_{\succ}$ from the instance, and reduce the upper quota of each place $p_j \in P$ with the load vector of $\mathrm{load}(p_j, \sigma_{\succ})$. We then solve PARETO-RR-on the obtained instance I' by applying Proposition 6; let $\sigma_{\sim}$ denote the returned assignment (note that there always exists a feasible and acceptable assignment, since there are no lower quotas for I). We return the assignment $\sigma := \sigma_{\sim} \cup \sigma_{\succ}$.[4]

The feasibility and acceptability follows from the feasibility and acceptability of $\sigma_{\succ}$ in I, and of $\sigma_{\sim}$ in I', as well as the fact that the upper quota for p_j in I' is set to $\bar{\mathbf{c}}_j - \mathrm{load}(p_j, \sigma_{\succ})$. To see that σ is Pareto-optimal as well, assume for the sake of contradiction that some assignment σ' is a Pareto-improvement over σ.

First, observe that $\sigma'(f_i) = \sigma_{\succ}(f_i)$ for each $f_i \in F_{\succ}$: indeed, assuming otherwise, we obtain that there exists some family f_i such that $\sigma'(f_{i'}) = \sigma_{\succ}(f_{i'})$ for each $i' < i$, but $\sigma'(f_i) \succ_i \sigma_{\succ}(f_i)$; however, this either contradicts the definition of $\sigma_{\succ}$ or the feasibility or acceptability of σ'. Therefore, we know that the restriction of σ' to I' must be a Pareto-improvement over $\sigma_{\sim}$ which contradicts the correctness of our algorithm for Proposition 6. Hence, our algorithm always produces a correct output. □

C.7 *Proof of Proposition 6*

Proposition 6 (⋆). *FEASIBLE-, PARETO-, and MAXUTIL-RR are FPT w.r.t. n.*

Proof. We present an algorithm for MAXUTIL-RR, the result then follows from Observation 1.

Assume that our input instance I admits a feasible assignment, and let σ be such an assignment with maximum utility. Our algorithm first guesses the set family $\mathcal{F} = \{\sigma^{-1}(p_j) : p_j \in P, \sigma(p_j) \neq \emptyset\}$; since $\mathcal{F}$ is a partition of a subset of F, there are $O(n^n)$ possible guesses to try.

The algorithm next creates the following edge-weighted bipartite graph $G = (\mathcal{F} \cup P, E)$. A set of families $\Gamma \in \mathcal{F}$ is adjacent in G to some place p_j if and only if $\underline{\mathbf{c}}_j \leq \sum_{f_i \in \Gamma} \boldsymbol{r}_i \leq \bar{\mathbf{c}}_j$, and we set the weight of the edge connecting Γ and p_j (if it exists) as $\sum_{f_i \in \Gamma} \boldsymbol{u}_i[j]$. We compute a maximum-weight matching $M^{\star}$ among those that cover the set $F^{\star} = \{f_i \in F : \underline{\mathbf{c}}_i \neq 0\}$; this can be done in $O((n+m)^3)$ time using e.g., the Hungarian algorithm. The algorithm outputs the assignment that to each place p_j covered by $M^{\star}$ assigns the families contained in $\Gamma \in \mathcal{F}$ where $\{\Gamma, p_j\} \in M^{\star}$.

The correctness of this algorithm follows from the observation that every matching M in G that covers $F^{\star}$ corresponds to a feasible assignment whose utility is the weight of M, and vice versa; note that feasibility is guaranteed by the definition of the edge set of G and the condition that the matching covers $F^{\star}$. The total running time of the algorithm is $O(n^{n+3})$. □

C.8 *Proof of Theorem 6*

Theorem 6 (⋆). FEASIBLE-RR, PARETO-RR *for equal preferences, and* MAXUTIL-RR *for equal utilities, are FPT w.r.t.* $t + r_{\max}$.

[4] Since the domains of $\sigma_{\sim}$ and $\sigma_{\succ}$ partition F, taking the union of $\sigma_{\sim}$ and $\sigma_{\succ}$ yields an assignment for I.

Proof. We present an N-fold IP for MAXUTIL-RR with equal utilities as follows. First, let $\mathcal{R} = \{\boldsymbol{r}_i : f_i \in F\}$ contain all requirement vectors associated with some family in our input instance. By possibly re-indexing the families in F, we can ensure $\mathcal{R} = \{\boldsymbol{r}_1, \ldots, \boldsymbol{r}_{|\mathcal{R}|}\}$. Note that $|\mathcal{R}| \leq (r_{\max} + 1)^t$ due to the definition of $r_{\max}$. Let also $n_{\boldsymbol{r}}$ denote the number of families with requirement vector $\boldsymbol{r} \in \mathcal{R}$.

We introduce a variable $x(p_j, \boldsymbol{r})$ for each $p_j \in P$ and $\boldsymbol{r} \in \mathcal{R}$ which is interpreted as the number of families with requirement vector $\boldsymbol{r}$ assigned to p_j. Additionally, we introduce "slack" variables $y(p_j, s_k)$ for each place $p_j \in P$ and service $s_k \in S$ interpreted as the available free capacity for service s_k at place p_j. Consider the following integer program ILP^N:

$$(\mathrm{ILP}^N) \quad \max \sum_{p_j \in P, \boldsymbol{r} \in \mathcal{R}} x(p_j, \boldsymbol{r}) \quad \text{such that}$$

$$\forall p_j \in P, s_k \in S: \; y(p_j, s_k) + \sum_{\boldsymbol{r} \in \mathcal{R}} \boldsymbol{r}[k] x(p_j, \boldsymbol{r}) = \bar{\mathbf{c}}_j[k] \quad (14)$$

$$\forall \boldsymbol{r} \in \mathcal{R}: \quad \sum_{p_j \in P} x(p_j, \boldsymbol{r}) \leq n_{\boldsymbol{r}} \quad (15)$$

$$\forall p_j \in P, \boldsymbol{r} \in \mathcal{R}: \; 0 \leq x(p_j, \boldsymbol{r}) \quad (16)$$

$$\forall p_j \in P, s_k \in S: \; 0 \leq y(p_j, s_k) \leq \bar{\mathbf{c}}_j[k] - \underline{\mathbf{c}}_j[k] \quad (17)$$

Claim 8. *There exists an integer solution to ILP^N with value u^* if and only if there exists a feasible assignments for I with utility u^*.*

Proof. Interpret $x(p_j, \boldsymbol{r})$ as the number of families with requirement vector $\boldsymbol{r}$ assigned to some place p_j, and $y(p_j, s_k)$ as the unused capacity of p_j for service s_k. Then constraints (14) and (17) for each $p_j \in P$ and $s_k \in S$ express the condition that the total requirement for service s_k of all families assigned to p_j should be at least $\underline{\mathbf{c}}_j[k]$ and at most $\bar{\mathbf{c}}_j[k]$. Constraints (15) and (16) express the condition that for each $\boldsymbol{r} \in \mathcal{R}$, the total number of families with requirement $\boldsymbol{r}$ assigned to some place in P should not exceed $n_{\boldsymbol{r}}$. This shows that constraints (14)–(17) together characterize feasibility assignments.

It remains to note that since utilities are equal, the total utility of an assignment is proportional to the number of families assigned, which is expressed by the objective function. ◁

Claim 9 (⋆). *ILP^N is an N-fold IP for $N = m$ with constraint $A^{(m)}$ where $A = \begin{pmatrix} I_{|\mathcal{R}|} & 0 \\ A_{\mathcal{R}} & I_t \end{pmatrix}$ for some $A_{\mathcal{R}}$ with $||A_{\mathcal{R}}||_{\infty} \leq r_{\max}$, and I_h denotes the identity matrix of size $h \times h$ for each $h \in \mathbb{N}$.*

Proof. Define the following matrices and vectors:

$$A_{\mathcal{R}} = (\boldsymbol{r}_1 \; \boldsymbol{r}_2 \; \ldots \; \boldsymbol{r}_{|\mathcal{R}|}) \in \mathbb{N}^{t \times |\mathcal{R}|}$$
$$\mathbf{x}_j = (x(p_j, \boldsymbol{r}_1) \; x(p_j, \boldsymbol{r}_2) \; \ldots \; x(p_j, \boldsymbol{r}_{|\mathcal{R}|}))^{\top} \in \mathbb{Z}^{|\mathcal{R}|}$$
$$\mathbf{y}_j = (y(p_j, s_1) \; y(p_j, s_2) \; \ldots \; y(p_j, s_t))^{\top} \in \mathbb{Z}^t$$

Then constraints (14) for all $\boldsymbol{r} \in \mathcal{R}$ but fixed $p_j \in P$ can be written as

$$(A_{\mathcal{R}} \; I_t) \cdot \begin{pmatrix} \mathbf{x}_j \\ \mathbf{y}_j \end{pmatrix} = \bar{\mathbf{c}}_j. \quad (18)$$

Gathering (18) for every $j \in [m]$, we obtain

$$\begin{pmatrix} A_{\mathcal{R}}\ I_t & 0 & \dots & 0 \\ 0 & A_{\mathcal{R}}\ I_t & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & A_{\mathcal{R}}\ I_t \end{pmatrix} \cdot \begin{pmatrix} \mathbf{x}_1 \\ \mathbf{y}_1 \\ \mathbf{x}_2 \\ \mathbf{y}_2 \\ \vdots \\ \mathbf{x}_m \\ \mathbf{y}_m \end{pmatrix} = \begin{pmatrix} \bar{\boldsymbol{c}}_1 \\ \bar{\boldsymbol{c}}_2 \\ \vdots \\ \bar{\boldsymbol{c}}_m \end{pmatrix}. \qquad (19)$$

Constraints (15) for all $\boldsymbol{r} \in \mathcal{R}$ can be written as

$$\begin{pmatrix} I_{|\mathcal{R}|}\ I_{|\mathcal{R}|}\ \dots,\ I_{|\mathcal{R}|} \end{pmatrix} \cdot \begin{pmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \\ \vdots \\ \mathbf{x}_m \end{pmatrix} \le \begin{pmatrix} n_{\boldsymbol{r}_1} \\ n_{\boldsymbol{r}_2} \\ \vdots \\ n_{\boldsymbol{r}_{|\mathcal{R}|}} \end{pmatrix}. \qquad (20)$$

Recall also that N-fold IPs can handle lower and upper bounds on variables, as required by constraints (16) and (17). Therefore, we can observe that, summing up equations (19) and inequalities (20), constraints (14)–(17) can be formed as an N-fold IP[5] whose coefficient matrix is

$$A^{(m)} = \begin{pmatrix} I_{|\mathcal{R}|}\ 0 & I_{|\mathcal{R}|}\ 0 & \dots, & I_{|\mathcal{R}|}\ 0 \\ A_{\mathcal{R}}\ \ I_t & 0 & \dots & 0 \\ 0 & A_{\mathcal{R}}\ \ I_t & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & A_{\mathcal{R}}\ \ I_t \end{pmatrix}.$$

for the matrix $A = \begin{pmatrix} I_{|\mathcal{R}|} & 0 \\ A_{\mathcal{R}} & I_t \end{pmatrix}$. ◁

The algorithm by Hemmecke et al [21, Theorem 6.2] solves such an N-fold IP for $N = m$ in time $||A||_\infty^{O(t\cdot|\mathcal{R}|^2+|\mathcal{R}|\cdot t^2)} \cdot m^3 \cdot L$ where L denotes the binary encoding of the constants on the right-hand side of the IP, the upper and lower quotas, and the objective function[6]; in our case $L = O(\log(n + m + c_{\max}))$. Recall that $|\mathcal{R}| \le (r_{\max} + 1)^t$, and that each entry in A is an integer at most $r_{\max}$. We can observe that the running time is fixed-parameter tractable w.r.t. parameter $t + r_{\max}$. □

C.9 *Proof of Theorem 7*

Theorem 7 (⋆). *The following problems are FPT w.r.t. parameter $m + t + r_{\max}$:*
- PARETO-RR*,*
- MAXUTIL-RR *on instances where the number of different utility values is at most $g(m+t+r_{\max})$ for some computable function g.*

Proof. Recall that the reduction from PARETO-RR to MAXUTIL-RR described in Observation 1 constructs an instance where utility values fall into the range $[m]$ with the sole exception of $-m \cdot n$; hence, the number of different utility values is at most $m + 1$. Hence, it suffices to solve MAXUTIL-RR, as the first result follows from the second one.

We are going to present an ILP for MAXUTIL-RR.

[5] The fact that (20) is not an equality but an inequality does not cause problems, as shown by Knop et al. in the full version of their paper [24].

[6] See Eisenbrand et al. [16] for a more recent, slightly faster algorithm.

First, let $\mathcal{R} = \{\boldsymbol{r}_i : f_i \in F\}$ contain all requirement vectors associated with some family in our input instance I of MAXUTIL-RR. Then $|\mathcal{R}| \le (r_{\max} + 1)^t$ due to the definition of $r_{\max}$. Second, let $\mathcal{U} = \{\boldsymbol{u}_i : f_i \in F\}$ contain all utility vectors associated with some family. Since the number of different utility values is at most $g(m + t + r_{\max})$, we know that- $|\mathcal{U}| \le (g(m + t + r_{\max}))^m$.

We define the *type* of a family $f_i \in F$ as $(\boldsymbol{r}_i, \boldsymbol{u}_i)$, so two families have the same type, if they have the same requirement and utility vectors. Let $\mathcal{T}$ denote the set of all family types appearing in the instance; then $|\mathcal{T}| \le |\mathcal{R}| \cdot |\mathcal{U}| \le (r_{\max} + 1)^t \cdot (g(m + t + r_{\max}))^m$, so the number of family types is bounded by a function of the parameter. We define the two type sets $\mathcal{T}_{(\boldsymbol{r},\cdot)} = \{(\boldsymbol{r}, \boldsymbol{u}) \in \mathcal{T} : \boldsymbol{u} \in \mathcal{U}\}$ and $\mathcal{T}_{(\cdot,\boldsymbol{u})} = \{(\boldsymbol{r}, \boldsymbol{u}) \in \mathcal{T} : \boldsymbol{r} \in \mathcal{R}\}$. Also, for each type $\tau \in \mathcal{T}$ we let n_τ denote the number of families of type τ in the instance.

We introduce a variable $x(p_j, \tau)$ for each $p_j \in P$ and $\tau \in \mathcal{T}$ which is interpreted as the number of families of type τ assigned to p_j. The number of variables is therefore $|\mathcal{T}| \cdot m$. Consider the following integer program ILP$_2$:

$$(\text{ILP}_2)\ \ \max \sum_{p_j\in P} \sum_{\boldsymbol{u}\in\mathcal{U}} \sum_{\tau\in\mathcal{T}_{(\cdot,\boldsymbol{u})}} \boldsymbol{u}[j] \cdot x(p_j, \tau) \quad \text{such that}$$

$$\forall p_j \in P: \quad \underline{\boldsymbol{c}}_j \le f \sum_{\boldsymbol{r}\in\mathcal{R}} \sum_{\tau\in\mathcal{T}_{(\boldsymbol{r},\cdot)}} x(p_j, \tau) \cdot \boldsymbol{r} \le \bar{\boldsymbol{c}}_j \qquad (21)$$

$$\forall \tau \in \mathcal{T}: \quad 0 \le \sum_{p_j\in P} x(p_j, \tau) \le n_\tau \qquad (22)$$

Claim 10 (⋆). *There exists an integer solution to ILP$_2$ with value u^* if and only if there exists a feasible assignments for I with utility u^*.*

Proof. Interpret $x(p_j, \tau)$ as the number of families of type τ that are assigned to place p_j. Then for each $p_j \in P$,

$$\sum_{\boldsymbol{r}\in\mathcal{R}} \sum_{\tau\in\mathcal{T}_{(\boldsymbol{r},\cdot)}} x(p_j, \tau) \cdot \boldsymbol{r},$$

formulates exactly the load of p_j, which implies that constraint (21) expresses the condition of feasibility. Moreover, for each family type $\tau \in \mathcal{T}$, the expression $\sum_{p_j\in P} x(p_j, \tau)$ formulates the total number families of type τ assigned to some place. Hence, constraint (22) expresses the condition that the assignment can only assign at most n_τ families in total. This means that integer solutions to ILP$_2$ correspond to feasible assignments for I, and vice versa. Finally, notice that the expression

$$\sum_{p_j\in P} \sum_{\boldsymbol{u}\in\mathcal{U}} \sum_{\tau\in\mathcal{T}_{(\cdot,\boldsymbol{u})}} \boldsymbol{u}[j] \cdot x(p_j, \tau)$$

formulates exactly the utility of the assignment, and thus a solution for ILP$_2$ with value u^* implies the existence of a feasible assignment σ with $\text{util}(\sigma) = u^*$, and vice versa. ◁

Since the number of variables in ILP$_2$ is bounded by a function of the parameter $m+t+r_{\max}$ and the number of constraints is FPT w.r.t. to the parameter, the problem can be solved in FPT time [27]. □

C.10 *Proof of Proposition 7*

Proposition 7 (⋆). *FEASIBLE-, PARETO-RR, and MAXUTIL-RR are in XP w.r.t. $m + t$ and are FPT w.r.t. $m + t + c_{\max}$.*

Proof. We present an algorithm for MAXUTIL-RR; due to Observation 1, this can be used to solve PARETO-RR as well. Our approach is a straightforward adaptation of the textbook dynamic programming for KNAPSACK.

A *load state* is a tuple $(\boldsymbol{\ell}_1, \dots, \boldsymbol{\ell}_m)$ of vectors where $\boldsymbol{\ell}_j \in \mathbb{N}^t$ satisfies $\boldsymbol{\ell}_j \le \bar{\boldsymbol{c}}_j$ for each $p_j \in P$. Define Λ as the set of all load states, and observe that $|\Lambda| \le (c_{\max})^{mt}$. Define also $F_i = \{f_1, \dots, f_i\}$ for each $i \in [n]$.

For each possible load state $\lambda = (\boldsymbol{\ell}_1, \dots, \boldsymbol{\ell}_m) \in \Lambda$ and each family $i \in [n]$ we compute the maximum utility $T(\lambda, i)$ that can be achieved by some assignment $\sigma : F_i \to P$ with $\mathrm{load}(p_j, \sigma) = \boldsymbol{\ell}_j$ for each $j \in [m]$; if no such assignment exists, then we will write $T(\lambda, i) = -\infty$.

We start by computing the values $T(\lambda, i)$, for $i = 1$ and for each load state $\lambda = (\boldsymbol{\ell}_1, \dots, \boldsymbol{\ell}_m) \in \Lambda$ as follows:

- if $\boldsymbol{\ell}_j = \boldsymbol{r}_1$ for some $j \in [m]$ and $\boldsymbol{\ell}_{j'} = 0$ for each $j' \in [m] \setminus \{j\}$, then $T(\lambda, 1) = \boldsymbol{u}_1[j]$;
- if $\boldsymbol{\ell}_j = 0$ for each $j \in [m]$, then $T(\lambda, 1) = 0$;
- otherwise $T(\lambda, 1) = -\infty$.

The correctness of these values can be seen by observing that there are exactly $m + 1$ possible assignments for F_1: either we leave family f_i unassigned, or we assign it to one of the places in P. Notice that we do not require feasibility in the definition of $T(\lambda, i)$.

After the above initialization for the case $i = 1$, we compute the values $T[\lambda, i]$ for each $i = 2, \dots, n$ and for all $\lambda \in \Lambda$ using the following recursive formula. Let $\lambda = (\boldsymbol{\ell}_1, \dots, \boldsymbol{\ell}_m)$; then

$$T(\lambda, i) = \max \left\{ T(\lambda, i-1), \max_{\substack{j \in [m] \\ \boldsymbol{\ell}_j \ge \boldsymbol{r}_i}} \left\{ T(\lambda^{(j)}, i-1) + \boldsymbol{u}_i[j] \right\} \right\} \tag{23}$$

where

$$\lambda^{(j)} = (\boldsymbol{\ell}_1, \dots, \boldsymbol{\ell}_{j-1}, \boldsymbol{\ell}_j - \boldsymbol{r}_i, \boldsymbol{\ell}_{j+1}, \boldsymbol{\ell}_m). \tag{24}$$

Finally, the algorithm returns the maximum utility achievable by some feasible load state, that is, its output is

$$\max\{T((\boldsymbol{\ell}_1, \dots, \boldsymbol{\ell}_m), n) : \underline{\boldsymbol{c}}_j \le \boldsymbol{\ell}_j \le \bar{\boldsymbol{c}}_j \text{ for each } j \in [m]\}.$$

Claim 11. *Equation (23) correctly computes $T(\lambda, i)$ for each $\lambda \in \Lambda$ and $i \ge 2$.*

Proof. We prove the claim by recursion on i, building on the fact that the algorithm computes the values for $T(\lambda, 1)$ for each $\lambda \in \Lambda$ correctly. Thus, suppose that the computations are correct for $i - 1$, and consider the formula (23) for i.

Consider some load state $\lambda = (\boldsymbol{\ell}_1, \dots, \boldsymbol{\ell}_m) \in \Lambda$. Assume first that $\sigma : F_i \to P$ is an assignment with maximum utility among those that satisfy $\mathrm{load}(p_j, \sigma) = \boldsymbol{\ell}_j$ for each $j \in [m]$; this means $T(\lambda, i) = \mathrm{util}(\sigma)$. Let σ' denote the restriction of σ to F_{i-1}. If σ leaves f_i unassigned, then the load of every place is the same under σ' and under σ, which implies $T[\lambda, i-1] \ge \mathrm{util}(\sigma)$ by our inductive hypothesis for $i - 1$. If σ assigns f_i to some place $p_j \in P$, then the load of p_j under σ' is $\boldsymbol{\ell}_j - \boldsymbol{r}_i$, while the load of every other place $p_{j'} \in P \setminus \{p_j\}$ is the same as under σ, that is, $\boldsymbol{\ell}_{j'}$. This means that $T[\lambda^{(j)}, i-1] \ge \mathrm{util}(\sigma') = \mathrm{util}(\sigma) - \boldsymbol{u}_i[j]$ where $\lambda^{(j)}$ is defined by (24), again using our hypothesis. Hence, irrespective of the value $\sigma(f_i)$, the value on the right-hand side of (23) is at least $\mathrm{util}(\sigma) = T(\lambda, i)$.

It remains to show the reverse direction, so assume that the right-hand side of (23) is $\hat{u}$; we are going create an assignment $\hat{\sigma}$ which satisfies $\mathrm{load}(p_j, \hat{\sigma}) = \boldsymbol{\ell}_j$ for each $j \in [m]$ and has utility at least $\hat{u}$.

First, if $\hat{u} = T(\lambda, i-1)$, then by our inductive hypothesis we know that there exists an assignment $\sigma : F_{i-1} \to P$ with $\mathrm{util}(\sigma) = \hat{u}$ and satisfying $\boldsymbol{\ell}_j = \mathrm{load}(p_j, \sigma)$ for each $j \in [m]$. Then setting $\hat{\sigma} = \sigma$ is sufficient. Otherwise, the right-hand side of (23) must be defined by $\hat{u} = T(\lambda^{(j)}, i-1)$ for some index $j \in [m]$. By our inductive hypothesis, there exists some assignment $\sigma' : F_{i-1} \to P$ with utility $u^\star - \boldsymbol{u}_i[j]$ that yields the load state $\lambda^{(j)}$ as defined by (24), i.e., for which the load of p_j is $\boldsymbol{\ell}_j - \boldsymbol{r}_i$ while the load of each remaining place $p_{j'}$ is $\boldsymbol{\ell}_{j'}$. In this case, we can extend σ' by assigning f_i to p_j, giving us the assignment $\hat{\sigma}$ with the required properties and utility $\hat{u}$. $\lhd$

Due to Claim 11 and by the definitions of the function $T(\cdot)$, the correctness of our algorithm follows. Each computation step described by the recursion (23) requires $O(m)$ time, and we perform it at most $|\Lambda|n$ times. By $|\Lambda| \le (1 + c_{\max})^{mt}$, the overall running time is $(1 + c_{\max})^{mt} O(nm)$.

By using standard techniques, we can not only compute the utility of an optimal, feasible assignment, but also determine a maximum-utility feasible assignment itself. $\square$

C.11 *Proof of Proposition 8*

Proposition 8 ($\star$)**.** FEASIBLE-, PARETO- *and* MAXUTIL-RR *are in XP w.r.t. the desired utility u^* if there are no lower quotas.*

Proof. Since there are no lower quotas, we have no reason to match a family $f_i \in F$ to a place $p_j \in P$ if $\boldsymbol{u}_i[j] \le 0$. Since utilities are integral, every pair $f_i \in F, p_j \in P$ such that $\sigma(f_i) = p_j$ and $\boldsymbol{u}_i[j] > 0$ contributes at least 1 to the final utility. Thus we can iterate over all subsets of family-place pairs of size at most u^* such that every pair has a positive utility, and verify whether any of them gives raise to a feasible assignment σ with $\mathrm{util}(\sigma) \ge u^*$. There are at most $O((nm)^{u^*})$ subsets of pairs. $\square$

Appendix D Further parameterizations

In this section we discuss our results regarding parameters that were not included in detail in the main paper. We show FPT results for the parameters r_Σ, c_Σ, and u_Σ, and present in Tables 2 and 3 a nearly complete complexity picture regarding the parameters $u_{\max}, u^*$, and $n_\sim$.

We define $n_\sim$ as the number of families who have ties in their preferences. That is, the number of families $f_i \in F$ such that there are two places $p_j, p_{j'} \in P$ such that f_i finds both p_j and $p_{j'}$ acceptable and is indifferent between them. Many of the hardness-results for this parameter follow from the fact that PARETO-RR is NP-hard even when there is only one place, which is why we did not include it in the main paper. We do however discover that PARETO-RR is FPT w.r.t. $n_\sim$ when there are no lower quotas (Proposition 5).

Note that families with all-zero requirement vectors can be assigned arbitrarily without affecting the feasibility, and thus we may disregard such families and assume that $n \le \sum_{f_i \in F} \sum_{s_k \in S} \boldsymbol{r}_i[k] = r_\Sigma$. Therefore, Proposition 6 implies fixed-parameter tractability of our problems when parameterized by the sum of all requirements.

Corollary 1. FEASIBLE-, PARETO-, *and* MAXUTIL-RR *are FPT w.r.t. r_Σ.*

Another simple observation yields fixed-parameter tractability for parameter u_Σ in the case when there are no lower quotas. Indeed,

Other param.	$u_{\max}$ LQ=0 / LQ≠0		u^* LQ=0 / LQ≠0	
–	NPh$^\circ$/NPh$^\circ$	[19]	**FPT/NPh**$^\circ$	[T4]/[P1]
m	**W1h**$^\circ$**/W1h**$^\circ$	[P1]	**FPT/W1h**$^\circ$	[T4]/[P1]
	XP/XP	[P7]	**FPT/XP**	[T4]/[P7]
$r_{\max}$	**NPh/NPh**	[T2]	**FPT/?, XP**$^+$	[T4]/[R3]
eq. util.	**FPT/FPT**	[T1]	**FPT/FPT**	[T1]
–	NPh$^\circ$/NPh$^\circ$	[19]	W1h$^\circ$/**NPh**$^\circ$	[19]/[P1]
			XP/NPh$^\circ$	[P8]/[P1]
t	NPh$^\circ$/NPh$^\circ$	[19]	**?,XP/NPh**$^\circ$	[P8]/[P1]
m	NPh$^\circ$/NPh$^\circ$	[19]	W1h$^\circ$/NPh$^\circ$	[19]/[P2]
			XP/NPh$^\circ$	[P8]/[P2]
$r_{\max}$	NPh$^\circ$/NPh$^\circ$	[19]	W1h$^\circ$/**NPh**$^\circ$	[19]/[P2]
			XP/NPh$^\circ$	[P8]/[P2]
$m+t$	**W1h**$^\circ$**/W1h**$^\circ$	[P1]	**?/W1h**$^\circ$	[P1]
	XP/XP	[P7]	**XP/XP**	[P7]
$m+r_{\max}$	NPh$^\circ$/NPh$^\circ$	[19]	W1h$^\circ$/**NPh**$^\circ$	[19]/[P2]
			XP/NPh$^\circ$	[P8]/[P2]
$t+r_{\max}$	**NPh/NPh**	[T2]	**?,XP/?**	[P8]
eq. util.	**FPT/FPT**	[T6]	**FPT/FPT**	[T6]
$m+t+r_{\max}$	**FPT/FPT**$^+$	[R2]	**FPT/FPT**$^+$	[R2]

Table 2. Additional results regarding parameters u^* and $u_{\max}$ for MAXUTIL-RR. Above: Results for the single-service case ($t = 1$). We skip the parameterization by n since for this case since it is FPT for the more general case. Bold faced ones are obtained in this paper. LQ=0 (resp. LQ≠0) refers to the case when lower quotas are zero (resp. may be positive). Here, NPh means that the problem remains NP-hard even if the corresponding parameter(s) are constant. All hardness results hold for binary utilities. Additionally, $^\circ$ means hardness results hold even for equal utilities, and $^+$ means the algorithm only works when the utilities are non-negative.

Other param.	$n_\sim$ LQ=0 / LQ≠0	
–	**FPT/NPh**	[P5]/[P1]
m	**FPT/W1h**	[P5]/[P1]
	FPT/XP	[P5]/[P7]
$r_{\max}$	**FPT/NPh**	[P5]/[R1]
–	**FPT/NPh**	[P5]/[P2]
m	**FPT/NPh**	[P5]/[P2]
t	**FPT/NPh**	[P5]/[P1]
$r_{\max}$	**FPT/NPh**	[P5]/[P2]
$m+t$	**FPT/W1h**	[P5]/[P1]
	FPT/XP	[P5]/[P7]
$m+r_{\max}$	**FPT/NPh**	[P5]/[P2]
$t+r_{\max}$	**FPT/NPh**	[P5]/[R1]
$m+t+r_{\max}$	**FPT/FPT**	[T7]

Table 3. Additional results regarding parameter $n_\sim$ for PARETO-RR. Since the problem is FPT w.r.t. $n_\sim$ when there are no lower quotas by Proposition 5, we only look at the general case. Above: Results for the single-service case ($t = 1$). We skip the parameterization by n since for this case since it is FPT for the more general case. Here, NPh means that the problem remains NP-hard even if the corresponding parameter(s) are constant.

in such a case we can discard all families with all-zero utility vectors, as they can be deleted from any assignment without changing its feasibility, acceptability, or total utility. Therefore, we may assume that $n \le \sum_{f_i \in F} \sum_{p_j \in P} \boldsymbol{u}_i[j] = u_\Sigma$, leading to the following consequence of Proposition 6.

Corollary 2. MAXUTIL-RR *is FPT w.r.t.* u_Σ *if there are no lower quotas.*

Finally, it is also not hard to see that setting c_Σ as the parameter also yields fixed-parameter tractability.

Proposition 9. FEASIBLE-, PARETO-, *and* MAXUTIL-RR *are FPT w.r.t.* c_Σ.

Proof. By definition, we have $c_{\max} \le c_\Sigma$. It is also clear we can discard all places with all-zero upper quotas, which means that we may assume $m \le c_\Sigma$. Finally, we can also discard all services for which every place has zero upper quota (removing also all families that require such a service), implying that we may suppose $t \le c_\Sigma$. Hence we have $m + t + c_{\max} \le 3c_\Sigma$, and the result follows from Proposition 7. □

D.1 Additonal Results

In this section, we make some remarks on how existing results can be modified to show parameterized results regarding $u_{\max}$, u^* and $n_\sim$.

Remark 1. PARETO-RR *for* $t = 1$ *is NP-hard even when* $r_{\max} = c_{\max} = 2$ *and no family has ties in its preferences.*

Proof. We modify the proof of Theorem 2. Observe that the total service requirements of the families are $R = 6|X|$, and the total upper quotas of the localities are $Q = 2|C| + 4|X|$. Since every literal appears in two clauses, and each clause has three literals, there are $\frac{4|X|}{3}$ clauses. Thus $Q = 2\frac{4|X|}{3} + 4|X| \ge 6|X| = R$. We create $Q - R$ many dummy families which each require one unit of service. They find all places acceptable, and rank them in an arbitrary order. We also modify the preferences of the existing families so that they rank all the families they find acceptable in an arbitrary order. Now, $n_\sim = 0$. We set the lower quotas of the places equal to their upper quotas.

Now there is a feasible and acceptable assignment if and only if there is an assignment that assigns every family to a place it finds acceptable. Since the dummy families find every place acceptable, it is sufficient to look into finding an assignment that places the original families to a place they find acceptable. The proof of Theorem 2 shows this is NP-hard. Since a Pareto-optimal, feasible, and acceptable assignment must assign every family to a place they find acceptable, PARETO-RR must be NP-hard even when $t = 1$, $r_{\max} = c_{\max} = 2$ and no family has ties in its preferences. □

Remark 2. *If utilities are non-negative,* MAXUTIL-RR *is FPT w.r.t.* $m + t + r_{\max} + u_{\max}$. *If there are no lower bounds,* MAXUTIL-RR *is FPT w.r.t.* $m + t + r_{\max} + u_{\max}$ *regardless of the utilities.*

Proof. If there are no lower bounds, we can set any negative utility to -1. There is never a reason to match a family to a place where it has negative utility.

We can observe the statement from the proof of Theorem 7. We know that the number of different utility vectors families may have is at most $(u_{\max} + 1)^m$ (or $(u_{\max} + 2)^m$ when there are no lower bounds) and the number of different requirement vectors is, as before, at most $(r_{\max} + 1)^t$. Thus the number of different family types is at most $u_{\max}^m (r_{\max} + 1)^t$ (or $(u_{\max} + 2)^m (r_{\max} + 1)^t$ when there are no lower bounds), which is a function of the parameter. Rest of the proof proceeds as in the proof of Theorem 7. □

Remark 3. MAXUTIL-RR *is XP w.r.t.* $r_{\max} + u^*$ *when* $t = 1$ *and the utilities are non-negative.*

Proof. Similarly to the proof of Proposition 8, we iterate over all the size at most u^* subsets of family-place pairs, and see if assigning them according to the pairs gives sufficient utility. If yes, we assign the families in the pairs to their places and update the upper and lower quotas accordingly. As we have already obtained sufficient utility, we can ignore the utilities of the families and use Theorem 1 to find a feasible assignment for the updated instance in FPT-time w.r.t. $r_{\max}$. $\square$