Dynamic Placement in Refugee Resettlement

NARGES AHANI, Worcester Polytechnic Institute
PAUL GÖLZ, Carnegie Mellon University
ARIEL D. PROCACCIA, Harvard University
ALEXANDER TEYTELBOYM, University of Oxford
ANDREW C. TRAPP, Worcester Polytechnic Institute

Employment outcomes of resettled refugees depend strongly on where they are placed inside the host country. Each week, a resettlement agency is assigned a batch of refugees by the United States government. The agency must place these refugees in its local affiliates, while respecting the affiliates' yearly capacities. We develop an allocation system that suggests where to place an incoming refugee, in order to improve total employment success. Our algorithm is based on two-stage stochastic programming and achieves over 98 percent of the hindsight-optimal employment, compared to under 90 percent of current greedy-like approaches. This dramatic improvement persists even when we incorporate a vast array of practical features of the refugee resettlement process including indivisible families, batching, and uncertainty with respect to the number of future arrivals. Our algorithm is now part of the *Annie*™ Moore optimization software used by a leading American refugee resettlement agency.

1 INTRODUCTION

There are 26 million refugees around the world [32]. The United Nations High Commissioner for Refugees (UNHCR) considers over 1.4 million of them to be in need of *resettlement*, that is, permanent relocation from a temporary country of asylum to the country of resettlement [31]. Resettlement is mainly targeted at the most vulnerable refugees, such as children at risk, survivors of violence and torture, and those with urgent medical needs. Dozens of countries around the world resettle refugees; in 2019, for example, around 63 000 refugees were resettled [31]. Still, the number of refugees in need of resettlement far exceeds the number that is actually resettled in every year.

Historically, most countries taking in resettled refugees have paid little attention to where inside the country these refugees are placed. This policy might be worth reconsidering, however, since there is ample evidence that the initial local resettlement destination dramatically affects the outcomes of refugees [7–10, 13, 15, 18, 28]. One specific variable impacted by community placement is whether and when resettled refugees find employment. Employment plays a key role in the successful integration of a refugee by "promoting economic independence, planning for the future, meeting members of the host society, providing opportunity to develop language skills, restoring self-esteem and encouraging self-reliance" [1].

Since promoting employment is so crucial, the American resettlement agency HIAS began in 2017 to match refugees to communities using the matching software *Annie*™ Moore (Matching and Outcome Optimization for Refugee Empowerment), which is designed to maximize the total number of refugees who obtain employment soon after arrival [3]. Each week, the US goverment assigns a new batch of refugees to HIAS, and *Annie*™ suggests which community each refugee in the batch should be placed in. Before this work, *Annie*™ made its suggestions using a greedy algorithmic approach, that is, each batch of arrivals was allocated by separately maximizing the expected employment of this batch (subject to the remaining community capacities and ensuring that refugees have access to necessary services). Allocating affiliate capacity in such a greedy way will likely lead to suboptimal employment, however: A placement algorithm could achieve better employment by weighing in each placement decision whether a slot of capacity is more beneficial when used by a refugee in the current batch or when saved up for some refugee potentially arriving later in the year.

In this paper, we improve the optimization approach of *Annie*™ by intentionally incorporating the dynamic nature of the matching problem. For this, we design two closely related algorithms—one based on stochastic programming and another based on Walrasian equilibrium—that optimize the dynamic matching of refugees to communities in the United States. Our focus is to study these algorithms in a rich model that captures all of the relevant practical features of the refugee resettlement process, including indivisible families of refugees, batching, and unknown numbers of refugee arrivals. We evaluate the performance of our algorithms on HIAS data from 2014 until 2019. We show that both algorithms achieve over 98 percent of the hindsight-optimal employment in all years whereas the greedy baseline achieves only around 90 percent. We then describe how we implemented our algorithms within *Annie*™ to create *Annie*™ 2.0, which has been well-received by HIAS leadership: "*Annie*™ 2.0 is a game-changer for our pre-arrivals processes, allowing us to plan and optimize our pre-arrival strategy a year rather than a week ahead."

1.1 Related Work

Our paper extends a line of work initiated by Bansak et al. [13], which aims to increase refugees' employment outcomes through data- and optimization-driven placement. This approach consists of two components: using machine learning to estimate the probability that a given refugee placed at a given community would find employment, and using mathematical programming to perform the optimization. Ahani et al. [3] adopted a similar approach to develop *Annie*™; they also pointed out the practical relevance of indivisible families and the possibility of batching. Both papers seek to maximize employment with respect to a current batch of refugees, without considering future arrivals; it is in this sense that we think of deployed algorithms as *greedy*, and that is indeed our benchmark in this paper.

Our dynamic refugee placement problem generalizes the classic *edge-weighted online bipartite matching* problem, but most algorithms in the theoretical literature are not promising for our application since they are optimized for overly pessimistic arrival scenarios. Whereas competitive analysis was quite successful for unweighted online bipartite matching [24], no constant-factor approximation algorithm is possible for the weighted setting if arrivals are adversarial [17]. In the random-order arrival model, a $1/e$-approximation is possible [25], but the algorithm is impractical; in particular, it leaves the first 37% of arrivals entirely unmatched. Even if arrivals are drawn i.i.d. from a known distribution, Manshadi et al. [27] show that no online algorithm can obtain a better approximation ratio than 0.823, far below the performance of even the greedy baseline in our setting. Since this impossibility is based on contrived arrival distributions, many papers additionally assume that arrivals belong to finitely many types determining their edge weights. In this setting, constructing matchings that are optimal up to lower-order terms (with high probability) is not difficult [see 5], and multiple papers obtain such results, often in generalizations of edge-weighted online bipartite matching [4, 5, 35]. What limits the applicability of these algorithms to our setting, however, is that these algorithms require the distribution over types explicitly in their input, and are often constructed based on the assumption that multiple arrivals of each type will occur in a single run of the algorithm. By contrast, we estimate employment scores based on 20 independent features, which means the number of refugee "types" is too large to enumerate and we do not expect to see identical refugees.

Our algorithmic approach can be seen as an instantiation of the *Bayes Selector*, an algorithmic paradigm that has yielded impressive theoretical and empirical results across various problems with stochastic online arrivals [11, 19, 29, 35, 36]. Conceptually, the Bayes Selector takes in a prediction of future arrivals and then performs the action (in our setting: chooses the affiliate for the current arrival) that seems most likely to coincide with the action taken by an optimal benchmark. Under some regularity conditions on the arrivals, algorithms following this methodology have *constant regret*, that is, the expected difference between the algorithm's performance and that of the optimal benchmark does not grow with the size of the problem. The prediction of future arrivals often takes other shapes, but can be a sampled trajectory of arrivals as in our work [11]. In most papers, the choice of

action is based on how *often* the optimal benchmark would take an action in the simulated future rather than, as in our work, on the marginal effect of an action on the optimal value. Very recently, however, Sinclair et al. [29] analyze the same variant of the Bayes Selector (the "hindsight planning policy") as our Eq. (1), and show that it gives constant regret for the problem of stochastic online bin packing. Even though we do not provide theoretical guarantees in this paper, the success of the Bayes Selector across related settings partially explains our good empirical performance.

Shadow prices have been used to guide decisions in online settings in a variety of contexts, including advertising [16, 33, 34], revenue management [30], worker assignment [20, 22], and resource allocation [6]. [2] develop a dynamic learning approach where prices are calculated in a similar manner to ours; while they update their match scores upon every doubling of the arrival history, we update our match scores upon every batch. [20] extend the advertising context of [16] to assign workers to tasks when match scores are initially unknown and must be learned. Like [20], [22] also consider the worker-to-job context, but learn scores while matching via an explore-then-exploit approach. In our setting, our scores are known in advance independent of arrivals [3].

In independent and concurrent work, Bansak [12] also considers dynamic refugee resettlement; the algorithm obtaining the highest employment in that study is equivalent to our two-stage stochastic programming formulation in the simplest setting. Our model is much richer as we include non-unit family sizes, incompatibilities between families and communities, and allow for uncertain arrival numbers. A second limitation of their work is that their best algorithm is prohibitively slow. This lack of computational efficiency pushes them to consider algorithms with worse employment outcomes, and it limits their empirical evaluation to a single month of arrivals. By contrast, we leverage multiple algorithmic insights to speed up the algorithm by two orders of magnitude without substantially trading off employment. This speed-up allows us to empirically evaluate our algorithms for realistic matching periods, which last for an entire fiscal year. We compare the running times of both algorithms in Section 6. Finally, very recent work by Bansak and Paulson [14] extends the earlier Bansak [12] by incorporating a secondary objective that seeks to consume capacity at similar rates across affiliates, improving case wait times across affiliates without sacrificing much employment.

1.2   Organization of the Paper

In Section 2, we provide an overview of the US refugee resettlement process. In Section 3, we outline our model of dynamic refugee matching. In Section 4, we propose our two algorithms and show that they obtain near-optimal employment in a baseline setting that ignores the indivisibility of families, batching, and uncertainty about the total number of arrivals. In the next three sections, we layer on complexity toward the setting encountered in practice: indivisible families (Section 5), batching (Section 6), and unknown arrival numbers (Section 7). In these sections, we demonstrate that indivisible families and batching do not substantially change our algorithms' employment performance, and that employment remains high unless the number of arrivals widely deviates from the numbers announced by the government. In Section 8, we then explain how we implemented our approach within *Annie*™ and conclude in Section 9. In the appendix, we provide deferred details and additional empirical analyses.

2   INSTITUTIONAL BACKGROUND

The federal Office of Refugee Resettlement was created by the Refugees Act in 1980. The Act established funding rules and authorized the President of the United States to set annual capacities for resettlement. The resettlement process is managed by the US Refugee Admissions Program (USRAP) of the US Department of State, in conjunction with a number of federal agencies across federal departments as well as the International Organization for Migration and the UNHCR.

Applications for the resettlement program take place from outside of the US, typically in refugee camps. The US government conducts security checks, medical screening, and performs cultural orientation, which can take upwards of 18 months [23]. After clearance, USRAP decentralizes the process of welcoming refugees to nine NGOs known as *resettlement agencies*, of which one is HIAS. Each agency works with their own network of local *affiliates*, each supported by local offices as well as religious entities like churches, synagogues, or mosques, which serve as community liaisons for refugees. Each agency typically works with dozens of affiliates, though the number of affiliates can fluctuate over time. Some affiliates lack services to host certain kinds of refugees. For example, certain affiliates do not have translators for non-English-speaking refugees or lack support for single-parent families.

Agencies have no influence on which refugees are cleared for resettlement by USRAP or on when the refugees might arrive. Resettlement agencies meet on a weekly or fortnightly basis in order to allocate among themselves the refugees that have been cleared by USRAP. Refugees are usually resettled with members of their family. Such an indivisible group of refugees is referred to as a *case*. As a family can split when its members are fleeing their home country, some refugees who are applying for resettlement might already have existing relatives or connections in the US. Such cases *with US ties* are automatically resettled near their existing ties. All other refugees, referred to as *free cases*, can be resettled by any agency into any of the agency's affiliates.

Each affiliate has an assigned annual capacity for the number of refugees (rather than cases) it can admit in a given fiscal year.[1] These capacities are approved by USRAP and, in theory, agencies cannot exceed them. In practice, capacities can be slightly adjusted towards the end of the year or, as in recent years, substantially revised in the course of the year. Since capacities limit the number of refugees *arriving* in a fiscal year rather than *allocated* in it, and since there is typically a delay of multiple months between the two events, the Department of State tells the resettlement agencies an estimated arrival date for each cleared case. Agencies are assessed annually by USRAP on their performance in finding employment for refugees within 90 days of their arrival. Data on 90-day employment is therefore diligently collected by the affiliates and monitored by the agencies.

3   MODEL

An instance of the matching problem first defines a set L of *affiliates*, and each affiliate ℓ has a capacity $c_\ell \in \mathbb{N}_{\geq 0} \cup \{\infty\}$ of how many refugees it can host. We call a collection $\{c_\ell\}_{\ell \in L}$ of capacities for all affiliates a *capacity profile c*. To describe changes in capacity, it will be useful to manipulate the capacity profiles as vectors. Specifically, we write $c - e_\ell$ to describe the capacity profile obtained from c by reducing the capacity of affiliate ℓ by 1.

On the other side of the matching problem is a set $N = \{1, \ldots, n\}$ of *cases*. Each case i represents an indivisible family of $s_i \in \mathbb{N}_{\geq 1}$ refugees. Furthermore, each case i, for each affiliate ℓ, has an *employment score $u_{i,\ell}$*, which indicates the expected number of case members that will find employment if the case is allocated to ℓ. Typically, these employment scores $u_{i,\ell}$ are real numbers in $[0, s_i]$, but we will also allow to set $u_{i,\ell} = -\infty$ to express that case i is not compatible with affiliate ℓ. We will refer to the combination of a case's size and vector of employment scores as the *characteristics* of the case. To ensure that the matching problem is always feasible, we will assume that L contains a special affiliate $\perp$ that represents leaving a case unmatched, where $u_{i,\perp} = 0$ for all cases i and $c_\perp = \infty$.[2]

---

[1]Each fiscal year ranges from October 1 of the previous calendar year to September 30. For example, fiscal year 2017 ranges from October 1, 2016 to September 30, 2017.

[2]For example, allowing cases to be unmatched is necessary since an arriving case might only be compatible with affiliates whose capacity is already exhausted. When these situations occur in practice, such cases do not remain unmatched; instead, capacities can be increased or case–affiliate incompatibilities overruled manually by the arrivals officer. For our sequence of models, we report the fraction of matched refugees in Appendix D.7, and find that our algorithms do not lead to fewer refugees being matched than in the greedy baseline. To lower the number of unmatched refugees at the cost of reducing employment, one can add a constant reward per refugee to the $u_{i,\ell}$ with $\ell \neq \perp$.

We use the employment scores developed by Ahani et al. [3], and we give details on data preprocessing and training in Appendix A. Throughout this paper, we consider these employment scores as ground truth, which means that we evaluate algorithms directly based on the employment scores. An evaluation of how accurately the employment scores predict employment outcomes is outside of the scope of this paper, and has already been performed by Ahani et al.

The goal of the matching problem is to allocate cases to affiliates such that the *total employment*, that is, the sum of employment scores, is maximized, subject to not exceeding capacities. For a set $I \subseteq N$ and a capacity profile $c = \{c_\ell\}_{\ell \in L}$, define MATCHING$(I, c)$ as the matching integer linear program (ILP) below, where variables $x_{i,\ell}$ indicate whether case $i \in I$ is matched to affiliate $\ell \in L$:

$$\text{maximize} \quad \sum_{i \in I} \sum_{\ell \in L} u_{i,\ell} \, x_{i,\ell}$$

$$\text{subject to} \quad \sum_{\ell \in L} x_{i,\ell} = 1 \qquad\qquad \forall i \in I$$

$$\sum_{i \in I} s_i \, x_{i,\ell} \leq c_\ell \qquad\qquad \forall \ell \in L$$

$$x_{i,\ell} \in \{0, 1\} \qquad\qquad \forall i \in I, \ell \in L.$$

Let OPT(I, c) denote the optimal objective value of MATCHING(I, c). The *linear programming* (LP) *relaxation* of MATCHING(I, c) is obtained by replacing the constraint $x_{i,\ell} \in \{0, 1\}$ by $0 \leq x_{i,\ell} \leq 1$ for all $i \in I, \ell \in L$. For a fixed matching, we define the *match score* of a case i as its employment score u_{i,ℓ_i} at the affiliate ℓ_i where it is allocated; we will also refer to its *match score per refugee*, $u_{i,\ell_i}/s_i$.

Finally, cases arrive *online*, that is, they arrive one by one and, when case i arrives, the decision of which affiliate to place it in must be made irrevocably, before the characteristics of the subsequent arrivals $i+1, \ldots, n$ are known.[3] Thus, although an online matching algorithm must still produce a matching whose indicator variables $x_{i,\ell}$ satisfy the constraints of MATCHING(N, c), the total employment $\sum_{i \in N, \ell \in L} u_{i,\ell} \, x_{i,\ell}$ typically will not attain the benchmark OPT(N, c) of the *optimal matching in hindsight*. While we will not commit to a specific model of how the characteristics of arriving cases are generated, these arrivals should be thought of as *stochastic* rather than worst-case, and the distribution of case characteristics as changing slowly enough that sampling from recent arrivals is a reasonable proxy for the distribution of future arrivals.

Throughout the following sections, we will consider a sequence of models, which incorporate an increasing number of features of the real-world refugee allocation problem: in Section 4, we consider traditional online bipartite matching, which results from requiring $s_i = 1$ in the above model; from Section 5 onward, we allow cases to have arbitrary size; from Section 6 onward, we also allow cases to arrive in batches rather than one by one; in Section 7, we no longer assume that the total number n of arriving cases is known to the algorithm.

4 ONLINE BIPARTITE MATCHING ($s_i = 1$)

In this section, we will consider the special case of online bipartite (weighted) matching. We stress that this classic problem does not capture key features of the refugee-allocation problem in practice, which we will add in later sections. Instead, online bipartite matching allows us to more cleanly draw connections to theoretical arguments, which help motivate our algorithm design. Later in the paper, we will empirically show that the approach continues to work well in richer and more realistic settings.

Formally, this section considers the model defined in the previous section, with the restriction that all cases consist of single refugees, that is, that $s_i = 1$ for all $i \in N$. Under this assumption, it is well-known that the optimum matching for the ILP MATCHING(I, c) can be found by solving its LP relaxation.

---

[3]From Section 6 onward, cases will instead arrive in batches, which can be allocated simultaneously.

4.1 Algorithmic Approach

To motivate our algorithmic approach, we begin by describing why matching systems currently deployed in practice lead to suboptimal employment. These systems assign cases *greedily*, which—putting aside batching for now—means that an arriving case i is matched to the affiliate ℓ with highest employment score $u_{i,\ell}$ among those that have at least s_i remaining capacity. The main problem with greedy assignment is that it exhausts the capacity of the most desirable affiliates too early. In particular, we observe on the real data that a large fraction of cases have their highest employment score in the same affiliate ℓ^*, but that the size of the employment advantage of affiliate ℓ^* over the second-best affiliate varies. Since it only considers the highest-employment affiliate for each case, greedy assignment will fill the entire capacity of ℓ^* early in the year, including with some cases that benefit little from this assignment. Consequently, cases that would particularly profit from being placed in ℓ^* but arrive later in the year no longer fit within the capacity.

Intuitively, the decision to match a case i to an affiliate ℓ has two effects: the immediate increase of the total employment by $u_{i,\ell}$ but also an opportunity cost for consuming ℓ's capacity, which might prevent profitable assignments for later arrivals. Since greedy assignment only considers the former effect, it leaves employment on the table.

A better approach is *two-stage stochastic programming*, which allocates an arriving case i to the affiliate ℓ maximizing the sum of the immediate employment $u_{i,\ell}$ and the expected optimal employment obtainable by matching the future arrivals subject to the remaining capacity. That is, if, at the time of i's arrival, the remaining capacities are given by c, two-stage stochastic programming allocates i to the affiliate

$$\underset{\ell \in L: c_\ell \geq s_i}{\operatorname{argmax}} \quad u_{i,\ell} + \mathbb{E}\Big[\textsc{Opt}\big(\{i+1,\ldots,n\}, c - s_i \cdot e_\ell\big)\Big],$$

where the expectation is taken over the characteristics of cases $j = i+1, \ldots, n$. Since adding a constant term does not change the argmax, this can be rewritten as

$$= \underset{\ell \in L: c_\ell \geq s_i}{\operatorname{argmax}} \quad u_{i,\ell} - \mathbb{E}\Big[\textsc{Opt}\big(\{i+1,\ldots,n\}, c\big)\Big] + \mathbb{E}\Big[\textsc{Opt}\big(\{i+1,\ldots,n\}, c - s_i \cdot e_\ell\big)\Big]$$

$$= \underset{\ell \in L: c_\ell \geq s_i}{\operatorname{argmax}} \quad u_{i,\ell} - \mathbb{E}\Big[\textsc{Opt}\big(\{i+1,\ldots,n\}, c\big) - \textsc{Opt}\big(\{i+1,\ldots,n\}, c - s_i \cdot e_\ell\big)\Big]. \tag{1}$$

Using our assumption that $s_i = 1$, this can be simplified to

$$= \underset{\ell \in L: c_\ell \geq 1}{\operatorname{argmax}} \quad u_{i,\ell} - \mathbb{E}\Big[\textsc{Opt}\big(\{i+1,\ldots,n\}, c\big) - \textsc{Opt}\big(\{i+1,\ldots,n\}, c - e_\ell\big)\Big].$$

Note that the expectation that is subtracted in either of the last two lines is exactly the expected opportunity cost of reducing the capacity of ℓ by placing case i there. This motivates our algorithmic approach: in every time step, we first compute a *potential* p_ℓ for each affiliate ℓ. Then, rather than myopically maximizing the utility of the match as does greedy assignment, our algorithm PM ("**p**otential **m**atch") myopically maximizes the utility of the current match minus the potential of the capacity used, as shown in Algorithm 1. (Note that an affiliate ℓ can always be defined in Line 5 as, by assumption, $c_\perp = \infty$.)

We estimate the expected value of the opportunity cost by averaging over a fixed number k of *trajectories*, each of which consists of randomly sampled characteristics of all arrivals $i + 1$ through n. As the characteristics of arriving refugees change over time, and as these changes tend to be gradual, we draw these arrival characteristics uniformly with replacement from the arrivals in the six months prior to the current allocation decision. In Appendix D.3, we evaluate different lengths of this sampling window.

ALGORITHM 1: PM(Potential)

Parameter: a subroutine Potential to determine affiliate potentials

1 initialize the capacities c_ℓ for each affiliate ℓ;

2 **for** $t = 1, \ldots, n$ **do**

3 $\quad$ observe the case size s_t and the employment scores $\{u_{t,\ell}\}_\ell$;

4 $\quad$ call Potential() to define a potential p_ℓ for each affiliate ℓ;

5 $\quad$ $\ell \leftarrow \text{argmax}_{\ell \in L : c_\ell \geq s_t} u_{t,\ell} - s_t \, p_\ell$;

6 $\quad$ allocate case t to ℓ and set $c_\ell \leftarrow c_\ell - s_t$;

For each sampled trajectory, it remains to calculate the potential, which we would like to equal the opportunity cost $\text{OPT}(\{i+1, \ldots, n\}, c) - \text{OPT}(\{i+1, \ldots, n\}, c - e_\ell)$. Clearly, this could be computed by solving $O(|L|)$ matching LPs, which is what the flagship algorithm by Bansak [12] does.

Instead, we make use of a celebrated result in matching theory [26] to compute the opportunity costs for *all* affiliates with remaining capacity as the shadow prices of a *single* LP:

FACT 1. *Fix a matching-problem instance, in which all cases i have size $s_i = 1$. In the LP relaxation of MATCHING(N, c), let $\{p_\ell\}_{\ell \in L}$ denote the unique element-wise maximal set of shadow prices for the constraints $\sum_{i \in N} s_i x_{i,\ell} \leq c_\ell$. Then, for each ℓ with $c_\ell \geq 1$,*

$$p_\ell = \text{OPT}(\{i+1, \ldots, n\}, c) - \text{OPT}(\{i+1, \ldots, n\}, c - e_\ell).$$

This suggests the procedure Pot1 for computing potentials, which is shown in Algorithm 2.

We also develop a second method Pot2 for computing potentials, which is based on a slightly different LP and has different theoretical underpinnings:

- whereas the matching LP for Pot1 does not include the current batch of arrivals, the current batch is included in the LP for Pot2,
- whereas Pot1 uses the element-wise maximal set of shadow prices, Pot2 uses the element-wise minimal one, and
- whereas Pot1 is theoretically derived from two-stage stochastic programming, Pot2 is motivated by a connection to Walrasian equilibria.

For conciseness, we defer the formal definition of Pot2 and its connection to the Walrasian equilibrium to Appendix B.

4.2 Empirical Evaluation

We evaluate the employment of our potential-based matching algorithm on real yearly arrivals at HIAS. For each fiscal year, we consider all refugees who arrived in this period, and we consider them in the order in which they were received for allocation by HIAS. For the capacities, we use the year's *final*, i.e. most revised, capacities.[4] We also immediately take into account that affiliates are restricted in which nationalities, languages, and family sizes they can accommodate, as well as in whether they can host single parents and the constraints on tied cases.

---

[4]When the number of refugees resettled in the fiscal year exceeds the official capacity, we use the number of resettled refugees instead. In these situations, HIAS negotiated an increase in capacity that is not always recorded in our data.

[5]One way of finding the maximal shadow price is to first solve the dual LP to find its objective value, then adding a constraint that constrains the objective of the dual LP to be equal to this optimal objective value, and to finally maximize the sum of dual variables p_ℓ over this new restricted LP.

ALGORITHM 2: Pot1(k)

Parameter: $k \in \mathbb{N}_{\geq 1}$, the number of trajectories per potential computation
Input: remaining capacities c, the index t of the last observed case, characteristics of cases arriving in the past 6 months
Output: a set of potentials p_ℓ for all affiliates ℓ

1 **for** $j = 1, \ldots, k$ **do**
2   for each $i = t+1, \ldots, n$, set s_i and $\{u_{i,\ell}\}_\ell$ to the size and employment scores of a random, recently arrived case;
3   solve the following bipartite-matching LP:

$$\text{maximize} \quad \sum_{i=t+1}^{n} \sum_{\ell \in L} u_{i,\ell}\, x_{i,\ell}$$
$$\text{subject to} \quad \sum_{\ell \in L} x_{i,\ell} = 1 \qquad\qquad\qquad \forall i = (t+1), \ldots, n$$
$$\sum_{i=t+1}^{n} s_i\, x_{i,\ell} \leq c_\ell \qquad\qquad \forall \ell \in L \quad (*)$$
$$0 \leq x_{i,\ell} \qquad\qquad\qquad \forall i = (t+1), \ldots, n, \forall \ell \in L.$$

4   for each ℓ, set p_ℓ^j to be the maximal shadow price[5] of the constraint $(*)$;
5 set $p_\ell \leftarrow (\sum_{j=1}^{k} p_\ell^j)/k$ for all ℓ;
6 **return** $\{p_\ell\}_{\ell \in L}$;

    The main way in which this experiment deviates from reality is the assumption (made throughout this section) that cases have unit size. To satisfy this assumption, we split each case of size $s_i > 1$ into s_i identical single-refugee cases with a $1/s_i$ fraction of the original employment scores. In subsequent sections, we will repeat the experiments without this modification.

    We study 6 fiscal years, from 2014 to 2019. As affiliates closed and opened across these years, the number of affiliates varies between 16 and 24 (not counting the unmatched affiliate $\perp$). Finally, the number of arriving refugees (respectively, cases) varies between 1 670 (resp., 640) and 4 150 (resp., 1 630) across fiscal years.
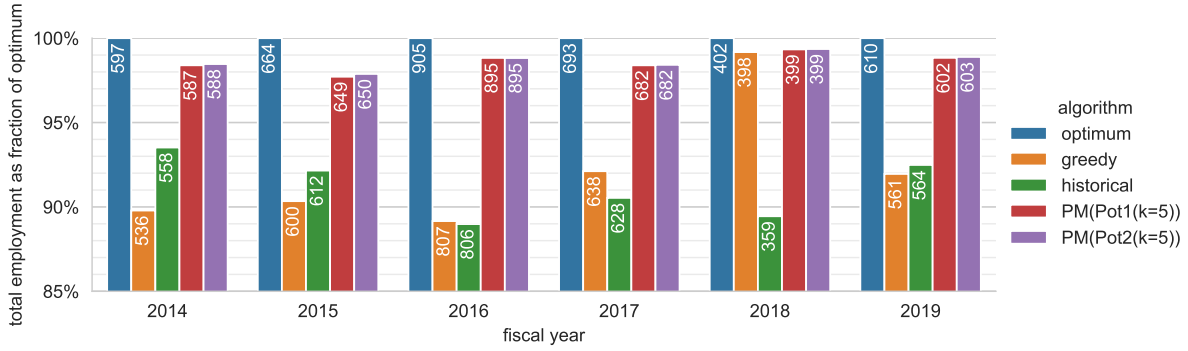

Fig. 1. Total employment obtained by different algorithms, assuming that cases are split into multiple cases of size 1. Capacities are the final capacities of the fiscal year. For the potential algorithms, total employment is averaged over 10 random runs. The numbers in the bars denote the absolute total employment; the bar height indicates the proportion of the optimum total employment in hindsight.

As shown in Fig. 1, even the greedy baseline obtains a total employment of between 89% and 92% of $\text{OPT}(N, c)$, the optimum matching in hindsight. (One outlier is the year 2018, which we discuss below.) Nevertheless, the greedy algorithm leads to between 50 and 100 fewer refugees finding employment every year compared to what would have been possible in the optimum matching. Our potential algorithms close a large fraction of this gap, obtaining between 98% and 99% of the optimal total employment, both for algorithms based on Pot1 and for those based on Pot2. Since experiments in this model take much longer to run than those in subsequent models, we defer a comparison between the two potential methods and between values of k to Section 6.1, where we can run the potential algorithms a sufficient number of times to discern smaller differences.

The fiscal year 2018 stands out from the others due to the fact that the greedy algorithm performs on par with the potential algorithms, at 99% of the hindsight-optimal total employment. This is easily explained by the fact that the capacities are much looser than in other fiscal years: whereas, in all other fiscal years between 2014 and 2019, the number of arriving refugees amounts to between 84% (2019) and 97% (2016) of the final total capacity across all affiliates, this fraction is only 48% in 2018. Since capacity is so abundant, the optimal matching will match a large fraction of cases to their maximum-score affiliate, and the greedy matching is close to optimal.

We also compare to the employment obtained by the allocation chosen by HIAS ("historical"). This comparison gives the historical matching a slight advantage, as HIAS sometimes overrides the incompatibility between an affiliate and a case, which we do not allow any other algorithm to do.[6]
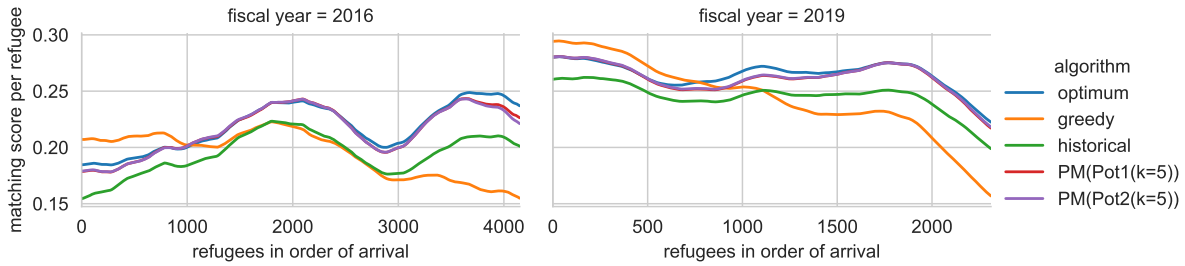

Fig. 2. Evolution of the per-refugee match score in order of arrival, for fiscal years 2016 and 2019 in the experiment of Fig. 1 (split cases, final capacities). Consecutive match scores are smoothed using triangle smoothing with width 500.

In Fig. 2, we investigate how the match score changes over the course of two fiscal years, 2016 and 2019, chosen to contain one year in which the greedy and historical baselines perform relatively poorly (2016) and one in which they perform well (2019). As the match score of subsequently arriving refugees can greatly differ, these graphs are heavily smoothed over time. If arrivals were drawn from a time-invariant distribution, we would expect the curves for the optimum matching in hindsight to be level, since how much employment the optimum matching can extract from a case would be independent of the case's arrival time. Instead, we see that the employment prospects of arrivals fluctuate noticeably over time; in particular, the early refugees in fiscal year 2016 and the late refugees in fiscal year 2019 seem to have worse employment prospects than other refugees in the plot.

The curves for both potential algorithms are nearly indistinguishable from one another, which shows that the algorithms make very similar decisions. In 2016, these curves start out closely tracking the curve of the optimal-hindsight matching, but fall behind for the last cases, which we observe in most fiscal years. The similarity of the curves over most of the year indicates that our approach of sampling trajectories from past arrivals is nearly as useful as the optimum algorithm's perfect knowledge of future arrivals and that it leads to a similar trade-off in extracting immediate employment versus preserving capacity for later arrivals. Of course,

---

[6]In these cases, we estimate the employment achieved by the case using the regression rather than using $u_{i,\ell} = -\infty$.
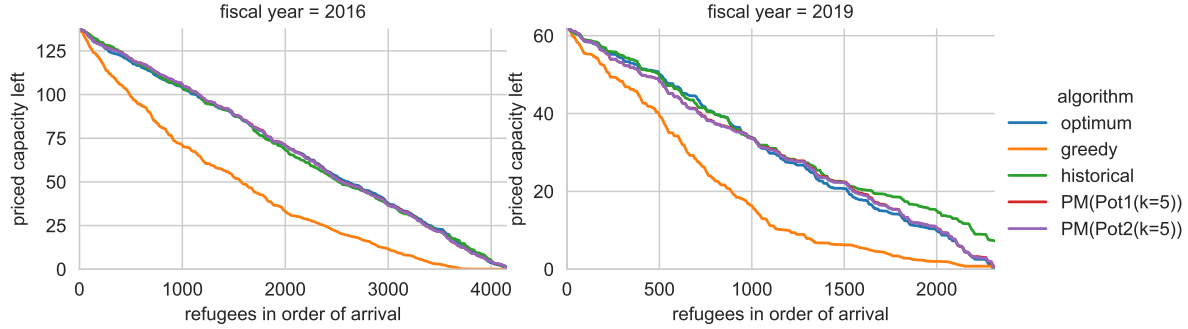
Fig. 3. Remaining priced capacity at the time of arrival of different refugees, for fiscal years 2016 and 2019 in the experiment of Fig. 1 (split cases, final capacities).

the imperfect knowledge of the future incurs a small loss towards the end of the fiscal year, likely because the amount of capacity reserved per affiliate does not perfectly match the demand, which explains the gap in total employment between the hindsight optimum and the potential algorithms. This typical end-of-year effect is not very pronounced in fiscal year 2019, likely because the final arrivals of fiscal year 2019 have lower employment probabilities than what would be expected based on past arrivals. Instead, the potential algorithms fall behind the optimum algorithm for some period in the middle of the year, perhaps because they are reserving capacity for late arrivals which the optimum already knows to hold little promise.

The most striking curve is that of the greedy algorithm, which lies above those of all other algorithms in the first quarter of arrivals, but then falls clearly below the other curves in the second half. This observation can be explained by the effect we predicted in the motivation of our potential approach: the greedy algorithm extracts small additional gains in employment early in the arrival period, at the cost of prematurely consuming the capacity of the most desirable affiliates. Then, the lack of capacity limits the match scores of later arrivals, resulting in an overall unfavorable trade-off. This effect can be directly seen in Fig. 3, in which we visualize the amount of capacity remaining in the most valuable affiliates. Specifically, looking at all arrivals of the fiscal year, we compute the shadow prices of the matching LP. At any point in time, we can then weight the remaining capacity by these prices to obtain a *priced capacity*. In Fig. 3, we see that the optimum-hindsight matching and the potential algorithms use up the priced capacity at a roughly constant pace and essentially consume it all. By contrast, the greedy algorithm uses up the capacity very quickly, such that at the median refugee, only 22% (2016) or 17% (2019) of the priced capacity is left.

The historical matching made by HIAS does not have such obvious defects, but still falls short in terms of total employment. In both reference years, the average employment moves in parallel with the optimum matching, meaning that HIAS does not overly focus on extracting employment at certain parts of the fiscal year at the expense of others. However, the average employment consistently lies below that of the optimum and of the potential algorithms. We see in Fig. 3 that, in 2019, HIAS started consuming the priced capacity at a near-constant pace very similar to that of the optimum algorithm. Around the median arrival, however, the historical matching slowed down its capacity consumption and ended up not consuming all priced capacity, which explains some loss in total employment. One reason for this behavior might be that HIAS staff treat the last 9% of the capacity as a reserve that they are more reluctant to use. In a year such as 2019, in which the overall arrivals were only 84% of the total capacity, this heuristic might have actually kept much of the reserve capacity free, including in the affiliates that could have generated higher employment. By contrast, the total arrivals in 2016 amounted to 97% of the overall capacity, which could explain why nearly all priced capacity was consumed in this year.

Despite using up priced capacity in a similar pattern as the optimum matching in 2016, the historical assignment achieved lower matching scores throughout the year. This indicates that the low employment of the historical matching is not just due to a reluctance to use the entire capacity, but that the priced capacity is furthermore inefficiently allocated.

5   NON-UNIT CASES ($s_i \geq 1$)

The most pressing aspect of refugee matching that we have ignored thus far is that many cases do not consist of individual refugees. Instead, they consist of an entire family of refugees, which has to be resettled to the same affiliate.

To accommodate cases consisting of multiple family members, we will from now drop the assumption that the s_i are 1. The main effect of this change is that the LP relaxation of the ILPs MATCHING(I, c) can now be a strict relaxation. Indeed, the LP relaxation might allow for higher objective values because it allows fractional solutions.[7] As a result, our dual prices will no longer *exactly* compute the marginal value of a unit of capacity. In any case, to retain the exact connection to stochastic programming in Eq. (1), PM would have to subtract the opportunity cost of s_i units of capacity from $u_{i,\ell}$, which might exceed s_i times the opportunity cost of a single unit of capacity.

However, as the capacity of most affiliates is much larger than the size of a typical case, both approximations can be expected to be relatively close, which is what we find empirically: we repeat the experiment of the previous section, but without splitting up cases into individual refugees. The results are nearly indistinguishable, which supports our decision to use LP relaxations even in the setting with indivisible cases. The full figures are deferred to Appendix D.1.

6   BATCHING

A second aspect that we have not considered thus far is that HIAS does not actually process arriving cases one by one, but in batches containing one or multiple cases. Most of these batches result from the weekly meetings between the resettlement agencies, but smaller batches with urgent cases are allocated between the weekly meetings.

The fact that cases arrive in batches does not make the problem harder; after all, a matching algorithm that does not support batching can still be used by presenting the cases of each batch to the algorithm one by one. As we will argue, however, batching represents an opportunity to improve on this strategy: there is a (limited) opportunity to increase total employment and a (substantial) opportunity to reduce running time.

Concerning total employment, using a non-batching algorithm in a batching setting is wasteful since it ignores potentially valuable information. Specifically, when the earliest cases of the batch are allocated, a non-batching algorithm presumes that the characteristics of the other cases in the batch are not yet known. Arguably, as the sizes of batches tend to be much smaller than the total number of cases n, the amount by which accounting for this information can increase total employment is likely to be limited.

As for running time, given that the matching algorithm receives no new information between the first and last case of a batch, it seems reasonable not to recompute potentials within a batch. As there tend to be 5 to 10 times more cases than batches and as the computation of potentials is the bottleneck in the running time of the potential algorithms, this promises to substantially speed up the algorithm.

In adapting our algorithm PM to batching, we will not change how we compute the potentials p_ℓ. However, the algorithm now allocates all cases in the batch at once, still with the objective of optimizing the immediate utility of the assignment less the sum of potentials consumed. Thus, our extended algorithm PMB ("**p**otential **m**atch with **b**atching", Algorithm 4 in Appendix C) allocates the current batch according to the solution to a

---

[7]One can always find a fractional solution that splits cases into $1/s_i$ fractions similarly to what we did in the evaluation of Section 4.2.

matching ILP, in which the utility of matching case i to affiliate ℓ is set to $u_{i,\ell} - s_i\, p_\ell$. Note that, if all batches have size $b = 1$, this algorithms coincides with our previous algorithm PM. Moreover, PMB also generalizes the greedy algorithm previously implemented in *Annie*™, which can be recovered by setting all potentials p_ℓ to zero.

We can now compare the running time of our algorithms to the flagship algorithm by Bansak [12], which is very closely related, but does not use dual prices to compute opportunity costs and handles batching in a way that does not improve running time. The computational bottleneck in both of our algorithms and theirs is the computation of bipartite-matching LPs over the trajectories of simulated future arrivals. Whereas we compute a single such program per batch of arrivals, Bansak solves $|L| \cdot b$ many such LPs per batch, where $|L|$ is the number of affiliates and b is the number of cases in the batch. In our dataset, a typical value of $|L| \cdot b$ is around 150, so these speed-ups are substantial.
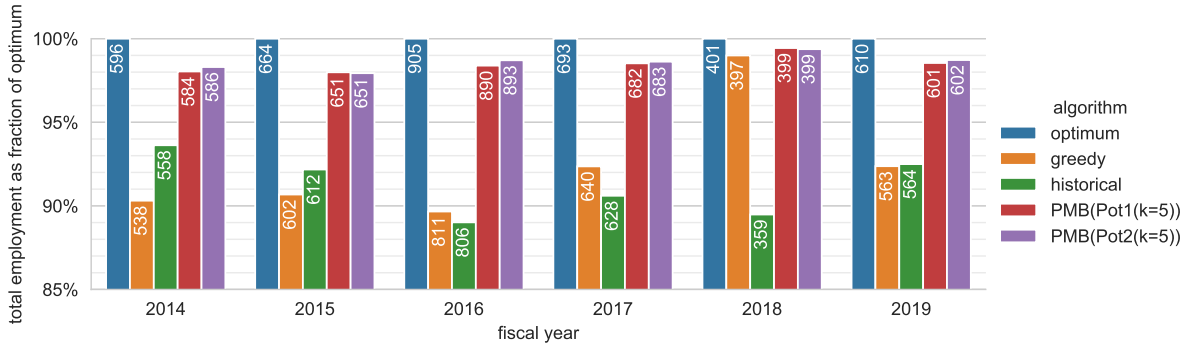
6.1   Empirical Evaluation

Fig. 4. Total employment, where cases are not split and arrive in batches. Capacities are the final fiscal year capacities. In contrast to Fig. 1, cases are treated as indivisible, cases arrive in batches, and the batching variants of greedy and the potential algorithms are used. For the potential algorithms, the mean employment across 50 random runs is shown.

We repeat the experiment measuring the total employment obtained by the algorithms, this time with the greedy algorithm and the potential algorithms allocating cases in batches. As shown in Fig. 4, the results again look very close to those in the restricted setting of online bipartite matching, confirming that our algorithmic approach generalizes well not only to non-unit case sizes but also to batching as it is used in practice.

Since processing entire cases in batches is much faster than processing cases (or individual refugees) one by one, we are now in a position to run each potential algorithm many times and analyze the distribution of total employments. As shown in Fig. 5, the total employment produced by each potential algorithm is sharply concentrated, especially when the algorithms use $k \geq 3$ trajectories to compute duals.

Running each algorithm many times enables us to compare the relative performance of the potential algorithms. Across both ways of computing potentials, and all fiscal years (with the exception of 2018, where everything is very close together), we see a clear tendency that averaging the potentials across more trajectories improves the employment outcome. These effects are somewhat limited, though, as going from a single trajectory to nine trajectories improves the median employment by less than half a percent of the hindsight optimum. As is to be expected, increasing k exhibits diminishing returns.

For k held constant, we observe that the Pot2 variants quite consistently outperform the Pot1 variants; again with the exception of 2018, in which a small inversion of this trend can be seen. While all potential algorithms perform very well, based on these results, we recommend the Pot2 potentials with a relatively large k for practical
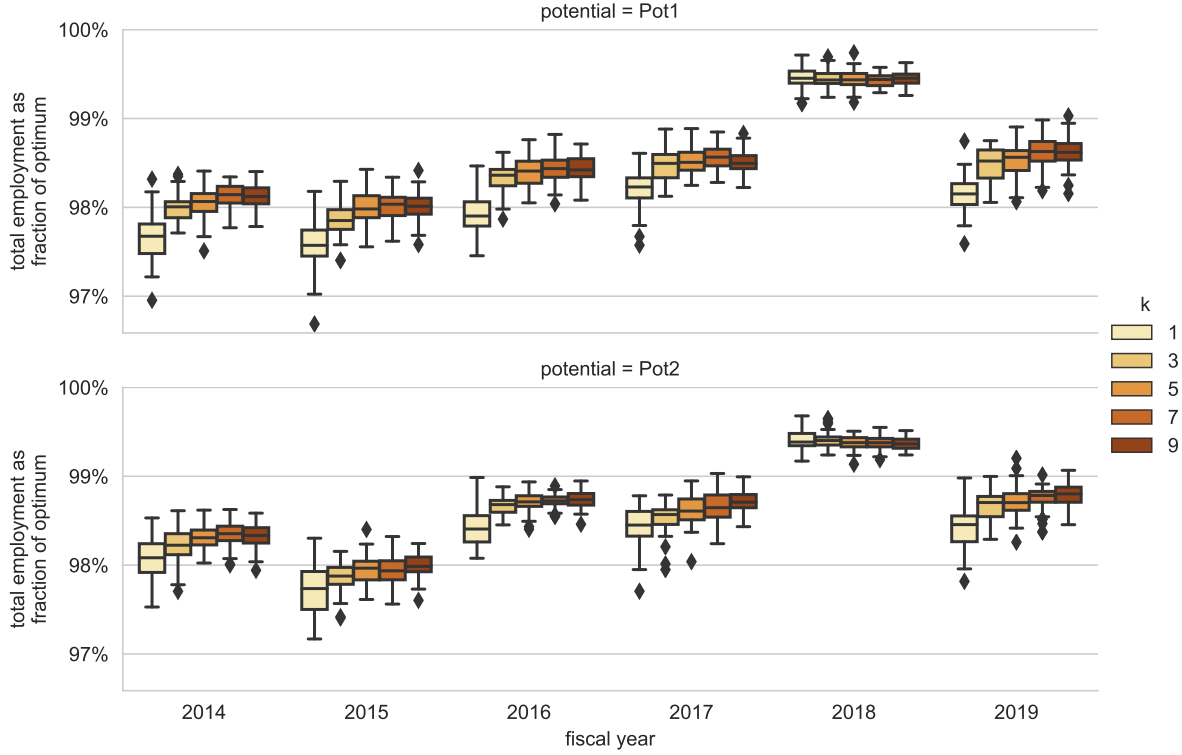
Fig. 5. Distribution of the total employment obtained by instantiating PMB with different potential methods and different k, in the experiment of Fig. 4 (whole cases, batching, final capacities) and over 50 random runs per algorithm.

implementation. Of course, increasing k increases the running time of the matching algorithm. However, since a resettlement agency computes only one set of potentials per day, the algorithm runs in few seconds even for $k = 9$ (see Appendix D.2).

To additionally support our observation that the potential algorithms outperform the greedy algorithm and the historical matching, we repeat the experiment from Fig. 4 for additional arrival sequences derived from the historical data. As we show in Appendix D.5, we obtain similar employment performance as in Fig. 4 if the arrival sequence for each year is reversed, or if we consider shifted yearly arrival periods from, say, April to the March of the following year rather than fiscal years (from October to September). In Section 7.2, we also evaluate the algorithms on bootstrapped arrivals. While we discuss more specific observations there, the potential algorithms perform similarly well or slightly better in that setting, consistently at 99% of the hindsight optimum.

7 UNCERTAINTY IN THE NUMBER OF FUTURE ARRIVALS

Given that our algorithm PMB supports non-unit sized cases and batching, it might seem that we are ready to replace the greedy algorithm in *Annie*™ by our potential algorithm. However, our algorithm crucially relies on one piece of input that the greedy algorithm did not need, namely, the total number of cases arriving in the fiscal year. This number determines the length of the sampled trajectories, which can greatly impact the shadow prices and, thus, how the algorithm allocates cases.

In principle, the information given to resettlement agencies should provide a fairly precise estimate of how many cases are expected to arrive. Indeed, before the start of each fiscal year, the US Department of State announces how many refugees it intends to resettle in that fiscal year, and resettlement agencies are instructed to prepare for a certain fraction of this total number. In fact, HIAS sets its affiliate capacities to sum up to 110% of this number of announced refugees, which is intended to give local affiliates a good idea of how many refugees they will receive while affording the resettlement agency some freedom in its allocation decisions.

7.1   Relying on Capacities
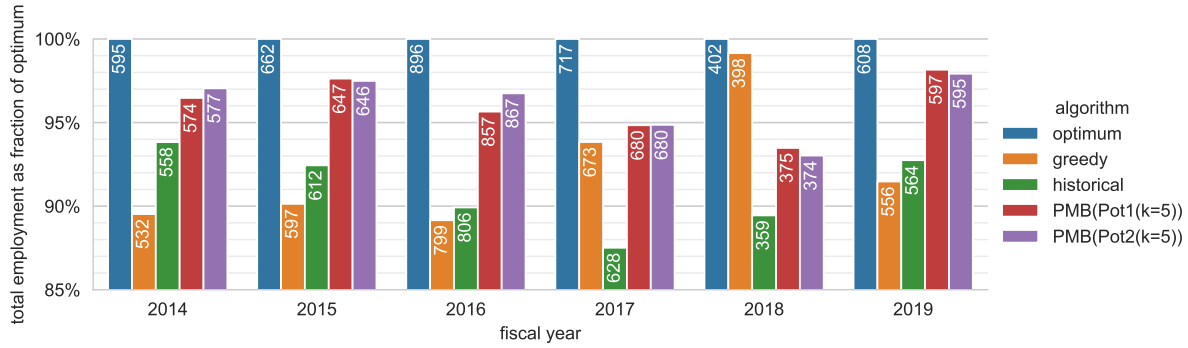
Fig. 6. Total employment, where cases are not split up and arrive in batches. The potential algorithms no longer have access to the true number of arriving cases but assume that the arriving refugees amount to 91% of the total capacity. Capacities are the *initial* capacities of the fiscal year (except for historical). For the potential algorithms, the mean employment across 50 random runs is shown.

It is thus natural to run our potential algorithms under the assumption that the number of arriving refugees will be $1/(110\%) \approx 91\%$ of the total announced capacity.[8] The result of this strategy is shown in Fig. 6. Since these experiments use the initial, unrevised capacities, the employment scores of the hindsight optimum and the greedy algorithm may differ from those in previous experiments, which used the most revised capacities.[9] In all fiscal years other than 2017 and 2018, the imprecise knowledge of future arrivals deteriorates the approximation ratio of the potential algorithms, but the potential algorithms continue to clearly outperform the greedy baseline overall, and they outperform the historical matching in every single year.

Setting aside the outlier years of 2017 and 2018 for the moment, we investigate the fiscal years 2016 and 2019, in which arrivals were otherwise highest and lowest relative to the announced capacity. In fiscal year 2016, the total arrivals were particularly large relative to the initial capacity: the arrival numbers added up to 100% of the initial capacity rather than 91%, which means that our potential algorithms expected around 3 770 refugees to arrive rather than the 4 150 that ended up arriving. As a result, the potential algorithms consume the priced capacity at an approximately constant rate, consuming it all around the expected number of expected refugees (Fig. 7, bottom left). Up to this point, the potential algorithms are more generous in consuming capacity than

---

[8]To convert the number of remaining refugees into a number of cases, we divide by the average case size of recent arrivals (over the years, this average size fluctuates between 2.4 and 2.6). While the number of refugees who have arrived is below 91% of the total capacity, this gives us a total number of cases n for the algorithms. Once the number of arrivals exceeds 91% of the total capacity, we make the algorithms assume that the current case is the last to arrive, that is, all subsequently sampled trajectories have length zero.

[9]This means that the comparison to the historical algorithm is not quite on equal terms, since the latter is constrained by a different set of capacities. In all fiscal years except for 2017 and 2018, the final capacities are element-wise larger than the original capacities.
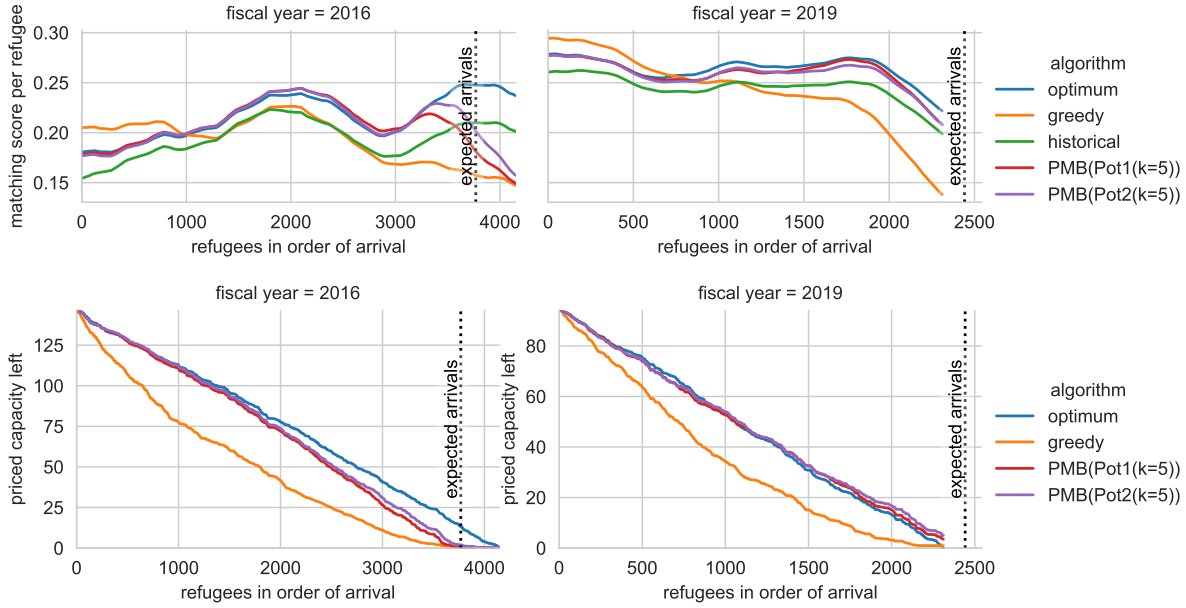
Fig. 7. Evolution of the per-refugee match score and remaining priced capacity in order of arrival, for fiscal years 2016 and 2019 and one run per algorithm in the experiment of Fig. 6 (whole cases, batches, initial capacities, potential algorithms do not know *n*). Dotted line show how many refugees the potential algorithms expect. Smoothing as in Fig. 2. Priced capacity is not shown for historical since it uses different capacities.

would be ideal given the actual number of arriving cases, which is why the potential algorithms obtain a slightly higher average employment over the first three quarters of arrivals (Fig. 7, top left) than the optimal matching in hindsight. For refugees arriving after the 3 770 expected refugees, however, the capacity in the best affiliates is used up, which is why the averaged employment sharply drops after this point.[10]

In 2019, by contrast, fewer refugees arrived than expected, only 86% of the total capacity. At the bottom right of Fig. 7, it is visible that the potential algorithms consume priced capacity at a slightly lower rate than the optimal algorithm in hindsight, as they aim to use up the capacity around 2 440 refugees rather than the 2 310 who ended up arriving. This effect is reflected in the average employment rates (top right), which lie below that of the optimal algorithm throughout most of the year.[11]

The fiscal years of 2017 and 2018 stand out due to the fact that the total number of arriving refugees fell far short of the announced number reflected in the approved capacities: in 2017, arrivals amounted to 65% of the approved capacities, while they amounted to only 46% in 2018. Both of these years fall into the beginning of the Trump administration, which not only sharply reduced the announced intake of resettled refugees, but furthermore abruptly halted the intake of refugees from six predominantly Muslim countries starting from early 2017.

As the potential algorithm depicted in Fig. 8 severely overestimates how many cases will arrive, it holds back much more priced capacity than would be optimal (bottom, solid lines). This causes the potential algorithms

---

[10]Note that, due to the triangle smoothing, the drop starts dragging down the curve 500 arrivals before its actual start.

[11]The drop in employment probabilities at the end of the fiscal year affects all algorithms including the hindsight optimum and must therefore be caused by an anomaly in arrival characteristics.
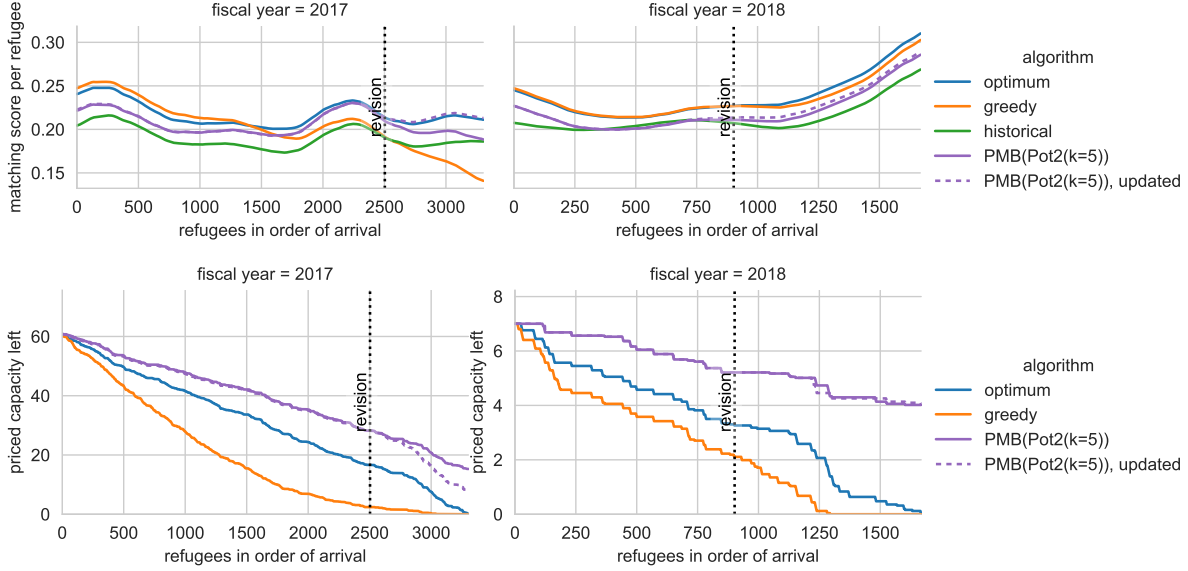
Fig. 8. Evolution of the per-refugee match score and remaining priced capacity in order of arrival, for fiscal years 2017 and 2018 in the experiment of Fig. 6 (whole cases, batches, initial capacities, potential algorithms do not know *n*). Dashed line shows evolution if potential algorithm updates its expected arrival number at time of capacity revision (dotted line).

to extract less employment throughout the year than the optimal algorithm (top, solid lines). As observed in Section 4.2, the capacities in 2018 are so loose that the greedy algorithm performs close to optimal.

In these two years, the US Department of State eventually reacted by correcting the expected arrivals downward and instructing the resettlement agencies to reduce their capacities. In fiscal year 2017, this revision came quite late and ended up underestimating the arrivals: where the arrivals amounted to only 65% of the initial capacities, they exceeded the revised total capacity at a level of 103%, rather than amounting to the 91% that was intended. Even if imperfect, this signal that arrivals are much lower than originally announced is still useful to the potential algorithms. Indeed, in Fig. 8, the dashed curve corresponds to a potential algorithm that still starts out expecting 91% of the initial capacities to arrive, but expects only 91% of the revised capacities to arrive from the point on where they were announced (vertical line). While this information comes late, the algorithm in fiscal year 2017 uses the new information to burn through the remaining priced capacity more aggressively (bottom left), which allows for higher employment among refugees arriving after the revision of arrival numbers (top left). As a result, the employment reaches 97% of the optimum in hindsight, exceeding the value of 95% without the updated information that we showed in Fig. 6.

By contrast, the revision in fiscal year 2018 did not yield much useful information; whereas the arrivals amounted to 46% of the initial capacities, they still amounted to 48% of the revised capacities. This seems to indicate that, even after half of the fiscal year's refugees had already been allocated, the administration overestimated the number of arriving refugees by a factor of two. Because the revision barely changed the number of expected arrivals, giving the potential algorithm access to this revised information does not have much effect (Fig. 8, right).

While we have considered the informational value of revisions above, our experiments have not considered that these revisions actually reduced the allowable capacities. Although we include a variant of the experiment

in Appendix D.6, it is difficult to meaningfully compare the employment achieved by different algorithms if the parameters of the matching problem are changed so drastically during the matching period. One particular challenge is that, while the amount of reduction was extraneously decided, HIAS was involved in deciding which capacities to decrease, which was done in a way that depended on previous allocation decisions.[12] Since we only know the revised capacities that were agreed upon, not the counterfactual revision of capacities that would be made, the greedy algorithm and the potential algorithms might have already exceeded a reduced capacity before it was announced. This means that the experiment rewards algorithms for greedily using up the capacity in the best affiliates before the revision, which we do not expect to be a good policy in practice. More generally, a substantial change in capacities is an exceptional situation, outside of our model, and cannot be addressed by our algorithm alone without manual intervention.
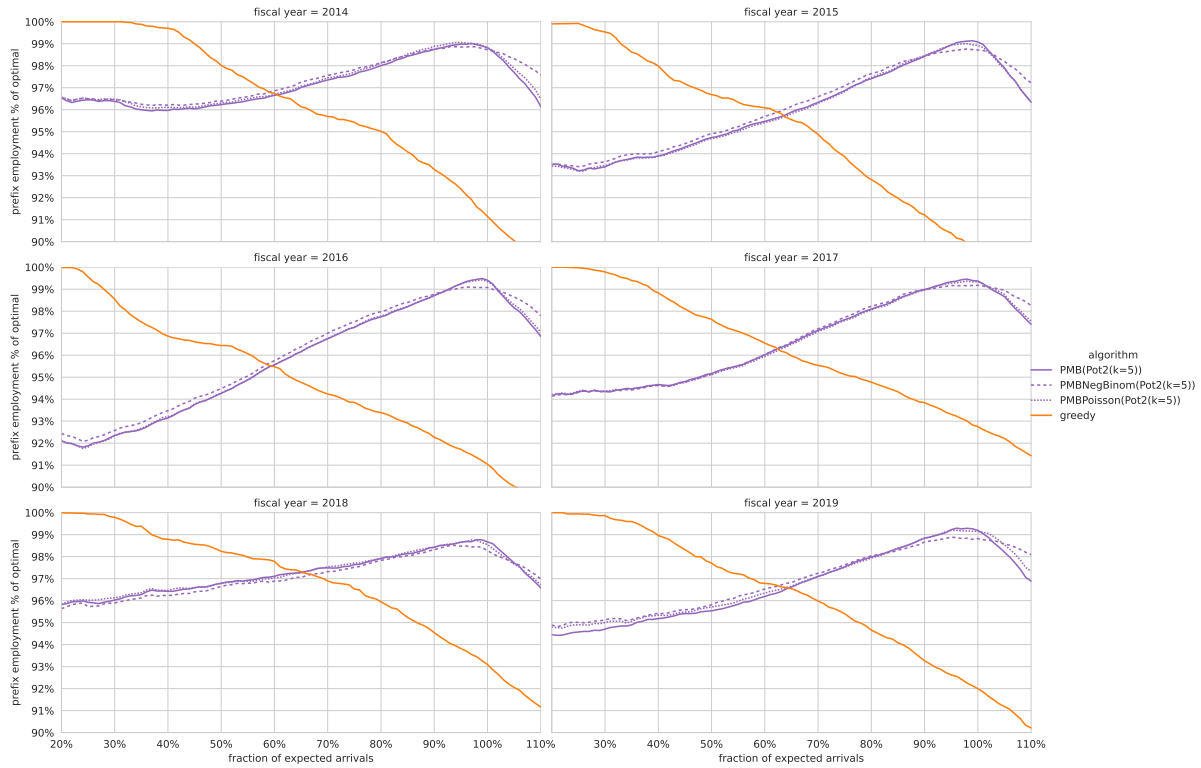

Fig. 9. Employment achieved by different algorithms as a function of how many refugees arrive. Refugee arrivals are bootstrapped over each fiscal year's historical arrivals, and the number of arriving refugees are given as a fraction of the historical arrivals. Capacities are 110% of historical matching. Employment is measured as a ratio of the optimal hindsight employment for the same set of arriving refugees. Curves are averaged over 10 arrival sequences.

---

[12]While the sum of capacities did not change much in fiscal year 2018, the capacities of some affiliates were substantially decreased and those of others were substantially increased.

7.2 Arrival Misestimation on Bootstrapped Data and Incorporating Uncertainty

To obtain more systematic insights into the robustness of potential algorithms to misestimated arrival numbers, we study bootstrapped case arrivals, which allows us to simulate varying numbers of arrivals. The results of this experiment are displayed in Fig. 9. As a baseline, consider the greedy algorithm, which obtains optimal employment when the number of arrivals is much lower than the total capacity (say, 25% of the expected arrivals, which is $\frac{25\%}{110\%} \approx 23\%$ of the capacity), but becomes more and more suboptimal the more refugees arrive.

By contrast, the potential algorithms perform best (around 99% of the optimal employment) when the number of arriving refugees matches what the algorithm expects. On average, this number is around half of a percentage point higher than in the corresponding non-bootstrapped experiments (Fig. 4). Such an increase is to be expected as the bootstrapping setup ensures that the algorithm draws trajectories from the same distribution from which the arrivals are generated. In particular, the real arrival sequence used for Fig. 4 might contain a drift in refugee characteristics or a seasonality not captured by our algorithm, and the lack of these features in the bootstrapped experiment allows for slightly higher employment. It is just as noticeable, however, that this increase is *only* half a percentage point, revealing that a drift of arrival characteristics and seasonality does not account for most of the remaining optimality gap of our algorithm.

The further the actual arrival number deviates from this expectation, the further the relative employment performance of the potential algorithm decreases. Noticeably, the performance more quickly deteriorates when the arrival numbers exceed the expectation, versus falling short. This sharp decline makes sense for two reasons. First, the algorithms aim to exploit all useful capacity exactly at the expected number of refugee arrivals; thus, only a subset of the affiliates remain available for subsequent arrivals. Second, once the number of arrivals exceeds the expectation, the trajectories in the potential algorithms add no cases beyond those that have already arrived, which means that the algorithm serves subsequent arrivals greedily. In the six fiscal years we observe, arrivals below the expectation seem like a more urgent problem than arrivals above the expectation, but over-arrivals might well become a problem under different political circumstances or when applying potential algorithms to other matching settings.

A natural way to make the potential algorithms more robust to inaccurate arrival estimates is to treat arrival estimates not as exact predictions but as subject to some uncertainty. Concretely, we adapt the potential algorithms by sampling trajectories of different lengths, each drawn from a "prior" distribution whose mean is the arrival estimate, conditioning this distribution such that trajectory lengths are never less than the number of refugees who have already been allocated. Conceivably, these adapted trajectories could generate potentials that are robust across a wider range of arrival numbers, and the adapted algorithm could therefore lead to higher employment when the official arrival numbers are inaccurate. The most obvious distribution is perhaps a Poisson distribution. As shown by the dotted line in Fig. 9, using Poisson trajectories hardly changes the employment outcomes for any of the experiments relative to the baseline of fixed trajectory sizes. This is most likely due to the low variance of the Poisson distribution. For a quite typical mean of 3 000 arriving refugees, 95% of the probability mass lies within a distance of only 3.6% of the mean. For this reason, we also try a distribution with overdispersion, specifically a negative binomial distribution parameterized to have its mean equal to the expected arrivals and its standard deviation equal to 10% of the expected arrivals. For example, if again 3 000 arrivals are expected, 95% of the probability mass deviates up to 20% from the mean. As the figure shows, negative-binomial trajectories lead to decent improvements in employment when more refugees arrive than expected. When fewer refugees arrive than expected, using random trajectory lengths helps more often than not, though with different degrees of success. Overall, negative-binomial arrivals seem to make the potential algorithms marginally more robust to misestimated arrival numbers, though not by enough to make misestimation less of an overall concern. Additionally, this additional robustness comes at a nonnegligible cost when arrival estimates are accurate.
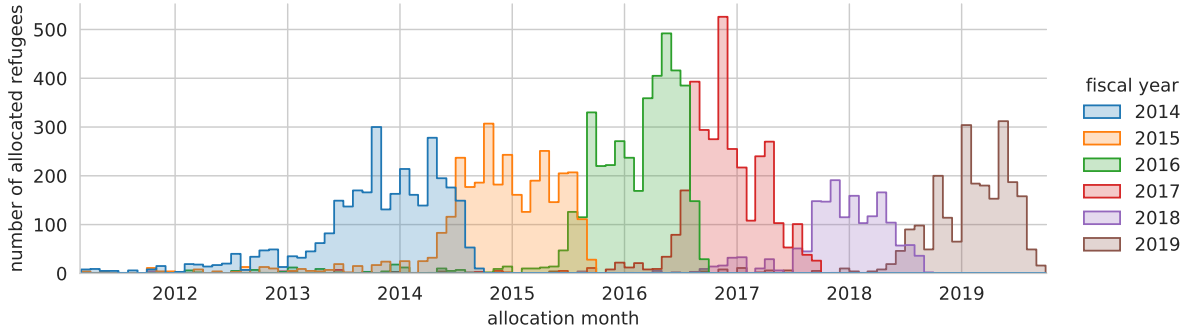
Fig. 10. Monthly number of allocated refugees, disaggregated by fiscal year of arrival.

7.3 Better Knowledge of Future Arrivals

In Section 7.1, we demonstrated that, even without outside supervision, our potential algorithms lead to substantial employment increases over the baselines, unless the announced capacities miss the eventual arrival numbers by an extreme margin. Even in these typical years, however, more accurate arrival predictions could increase the total employment on the order of percentage points of the hindsight optimum. Obviously, more accurate information about arrivals would be even more useful in years like 2017 and 2018, in which the official information is unreliable.

One approach would be to use time-series prediction to estimate the number of arrivals. For instance, when the US Department of State revised the capacities for the fiscal year 2018 in January 2018 (several months into the fiscal year), the announcement that 2.5 times more refugees were still to come than had already arrived might have raised some doubts. However, the graph of monthly arrivals in Fig. 10 shows that late increases in arrival rates may actually happen as they did in fiscal year 2016.[13]

A fundamental challenge that any data-driven approach faces is that there is very little data to learn from. Indeed, while HIAS has data on hundreds of thousand of refugees, they only have data on 15 fiscal years, which is, moreover, incomplete and smaller-scale in earlier years. Thus, there is a limited foundation to learn about how arrival patterns change between years. This task becomes especially difficult given that arrival numbers are heavily influenced by external events such as elections, the emergence of humanitarian disasters, and changes in immigration policy, which cannot be deduced from past arrival patterns. Thus, while a time-series prediction approach might lead to marginal improvements over naïvely expecting 91% of the capacity to arrive, past arrival numbers are unlikely to give enough information to accurately predict future arrival numbers.

Fortunately, resettlement agencies such as HIAS already possess much richer information and insights into the dynamics of refugee arrivals than a pure data approach would consider. In fiscal year 2017, for example, HIAS foresaw a worsening climate for refugee resettlement immediately after the November 2016 election[14] and was aware of concrete plans to drastically reduce refugee intake in January 2017,[15] both before these changes were reflected in arrival numbers and before the capacities were officially updated in March 2017. Similarly, HIAS continuously monitors domestic politics and international crises for their potential impact on resettlement, and moreover it has some limited insight into the resettlement pipeline, which allows it to prepare for changes

---

[13]In fiscal year 2016, the number of arrivals after January 2016 was 1.6 times larger than the number that had arrived so far. In the fiscal year of 2015, the number of refugees arriving after January 2015 was only 75% of that arriving before.

[14]https://www.hias.org/news/press-releases/hias-calls-president-elect-trump-respect-longstanding-refugee-policy

[15]https://www.hias.org/news/press-releases/trumps-planned-action-refugees-betrayal-american-values

in arrivals. We therefore believe that, rather than building a sophisticated tool for predicting arrivals in a fully autonomous manner, it is preferable to allow HIAS staff to override our prediction with more advanced information.

8   IMPLEMENTATION IN *ANNIE*™ MOORE

To enable HIAS to benefit from dynamic allocation via potentials, we have integrated new features into its matching software *Annie*™ MOORE. A crucial design requirement is that HIAS staff must be able to override the allocation recommendations of *Annie*™ when they are aware of requirements outside of our model. From an interface-design perspective, the challenge is to visualize the effect of such overrides on total employment, enabling HIAS staff to make informed trade-offs. In the original, static model, this was easy enough: as the quality of a matching was just the total employment of the current batch, the interface labeled each case–locality match with its associated employment score, and staff could drag the case to other localities to see the respective employment scores. In a dynamic setting, however, presenting only the employment scores may unintentionally encourage HIAS staff to greedily use capacity in their overrides, at the expense of future arrivals.

As we illustrate in Fig. 11, the new interface of *Annie*™ augments the original interface with information about affiliate potentials, thereby taking future arrivals into account. Specifically, the background color of the tile for case i encodes the *adjusted* employment score, that is, the original employment score $u_{i,\ell}$ less the potential $s_i\, p_\ell$ of the capacity consumed in affiliate ℓ.[16] The fact that the algorithm PMB always maximizes the sum of adjusted employment scores in its allocation of the current batch means that the algorithm is explainable in terms of the information presented to the user. In the interface, the *green* color spectrum indicates positive adjusted employment scores (meaning that the employment score of the case outweighs the loss in future employment), while the *red* color spectrum highlights negative adjusted scores (where a placement reduces future employment by more than its employment score). Darker colors signify greater magnitudes.

In overriding the allocation recommended by *Annie*™, HIAS staff should be able to quickly find alternative placements for a case that do not reduce immediate and future employment by more than necessary. To support this workflow, our interface shows the adjusted employment scores of a case across all affiliates at a glance: as shown in Fig. 12, upon dragging a particular case tile from its current placement, all other case tiles temporarily fade in appearance, and the shading of every affiliate tile temporarily assumes the adjusted employment score relative to the selected case. By hovering a selected case tile over a new affiliate, the original (numeric) employment score and the adjusted match score (background color of the case tile) dynamically update. Moreover, incompatibilities with affiliates due to nationality, language, family size, and single parent households can be seen via an exclamation mark in the lower left corner of the affiliate tile. After dropping the case tile in a new affiliate, the background color for each affiliate returns to its original blue shade, and all affiliate-tile exclamation marks disappear.

On a separate screen (not shown), *Annie*™ enables the entry of a prediction for total refugee arrivals, as mentioned in Section 7.3. This estimate can be critical to inform the process of estimating proper shadow prices, as at times HIAS is in a better position to give more accurate case arrival predictions than officially announced capacities.

9   CONCLUSION

We have developed and implemented algorithms for dynamically allocating refugees in a way that promotes refugees' prospects of finding employment. These algorithms outperform the baselines, even when taking into account how refugee placement in practice deviates from a classic matching setting.

While we have tested the algorithms as an autonomous system, the success of *Annie*™ in increasing employment outcomes in practice will depend on how it performs in interaction with HIAS resettlement staff. In Section 7.3,

---

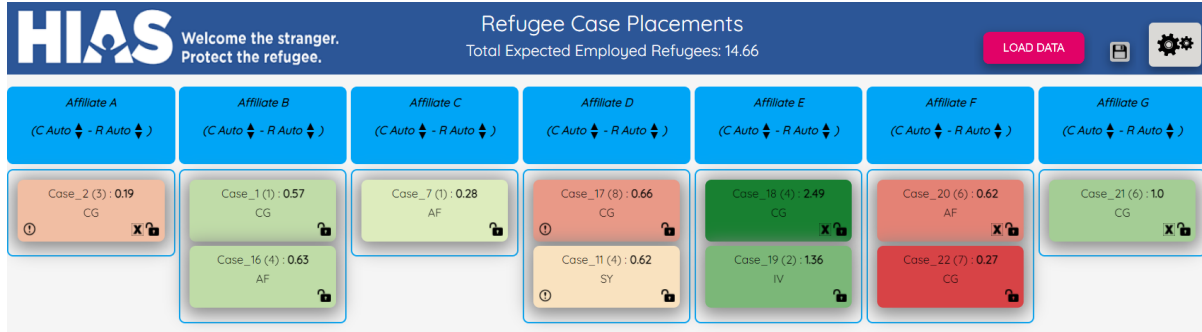[16]The employment scores of cases in affiliates are prominently retained in a text label.

Fig. 11. Updated *Annie*™ Interface. Family tiles now show both original numerical employment scores of families in affiliates, as well as the *adjusted* employment score by its shading. Green indicates positive adjusted scores, red negative scores, and darker colors represent greater magnitudes.
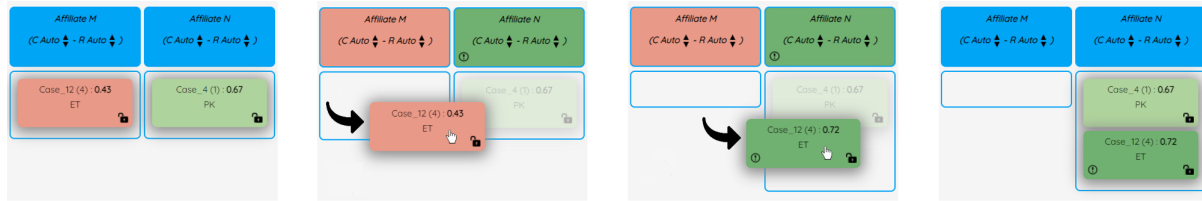

Fig. 12. Moving a family tile. Other case tiles fade, and affiliate tiles are colored as per their adjusted employment scores in shades of green (positive) or red (negative). Exclamation marks indicate incompatibilities.

we already saw that the allocation decisions of *Annie*™ can greatly profit from human decision makers providing better estimates of future arrivals. Human input is equally crucial in dealing with uncertainty in several other places: for example, HIAS staff might intervene by correcting the arrival year of a case should the Department of State's estimate seem off, or they might increase some affiliate capacities late in the year if they anticipate that these capacities can be increased. By allowing all parameters of the matching problem to be changed, *Annie*™ allows HIAS resettlement staff to improve the matching using all available information.

Ideally, the human-in-the-loop system consisting of the matching algorithm and HIAS staff can combine the strengths of both of its parts: On the one hand, the algorithms in *Annie*™ capitalize on subtle patterns in employment data and manage capacity more effectively over the course of the fiscal year. On the other hand, the expert knowledge of HIAS staff enables the system to handle the uncertainty that is inherent in a matching problem involving the actions of multiple government agencies, dozens of affiliates, and thousands of refugees. In light of the administration's recent increase of the total resettlement capacity from 15 000 to 125 000,[17] we foresee both parts playing a crucial role: the increasing scale of the problem will make data-based algorithms more effective, and human guidance will be necessary to navigate the evolving environment of a rapidly growing operation.

ACKNOWLEDGMENTS

---

[17]https://www.hias.org/news/press-releases/refugee-cap-fy2022-set-125000

REFERENCES

[1] A. Ager and A. Strang. 2008. Understanding Integration: A Conceptual Framework. *Journal of refugee studies* 21, 2 (2008), 166–191.

[2] S. Agrawal, Z. Wang, and Y. Ye. 2014. A dynamic near-optimal algorithm for online linear programming. *Operations Research* 62, 4 (2014), 876–890.

[3] N. Ahani, T. Andersson, A. Martinello, A. Teytelboym, and A. C. Trapp. 2021. Placement Optimization in Refugee Resettlement. *Operations Research* 69, 5 (2021), 1468–1486.

[4] S. Alaei, M. Hajiaghayi, and V. Liaghat. 2012. Online Prophet-Inequality Matching with Applications to Ad Allocation. In *Proceedings of the 13th ACM Conference on Electronic Commerce*. 18.

[5] S. Alaei, M. Hajiaghayi, and V. Liaghat. 2013. The Online Stochastic Generalized Assignment Problem. In *Proceedings of the 16th International Workshop on Randomization and Approximation Techniques in Computer Science*. 11–25.

[6] A. Asadpour, X. Wang, and J. Zhang. 2020. Online resource allocation with limited flexibility. *Management Science* 66, 2 (2020), 642–666.

[7] O. Åslund, P.-A. Edin, P. Fredriksson, and H. Grönqvist. 2011. Peers, Neighborhoods, and Immigrant Student Achievement: Evidence from a Placement Policy. *American Economic Journal: Applied Economics* 3, 2 (2011), 67–95.

[8] O. Åslund and P. Fredriksson. 2009. Peer Effects in Welfare Dependence: Quasi-Experimental Evidence. *Journal of Human Resources* 44, 3 (2009), 798–825.

[9] O. Åslund, J. Östh, and Y. Zenou. 2010. How Important is Access to Jobs? Old Question, Improved Answer. *Journal of Economic Geography* 10, 3 (2010), 389–422.

[10] O. Åslund and D.-O. Rooth. 2007. Do When and Where Matter? Initial Labour Market Conditions and Immigrant Earnings. *The Economic Journal* 117, 518 (2007), 422–448.

[11] S. Banerjee, I. Gurvich, and A. Vera. 2020. *Constant Regret in Online Allocation: On the sufficiency of a single historical trace*. Working Paper.

[12] K. Bansak. 2020. A Minimum-Risk Dynamic Assignment Mechanism Along with an Approximation, Heuristics, and Extension from Single to Batch Assignments. *arXiv preprint arXiv:2007.03069v2* (2020).

[13] K. Bansak, J. Ferwerda, J. Hainmueller, A. Dillon, D. Hangartner, D. Lawrence, and J. Weinstein. 2018. Improving Refugee Integration Through Data-Driven Algorithmic Assignment. *Science* 359, 6373 (2018), 325–329.

[14] K. Bansak and E. Paulson. 2022. Outcome-Driven Dynamic Refugee Assignment with Allocation Balancing. *arXiv preprint arXiv:2007.03069v4* (2022).

[15] A. P. Damm. 2014. Neighborhood Quality and Labor Market Outcomes: Evidence from Quasi-Random Neighborhood Assignment of Immigrants. *Journal of Urban Economics* 79 (2014), 139–166.

[16] N. R. Devanur and T. P. Hayes. 2009. The Adwords Problem: Online Keyword Matching with Budgeted Bidders under Random Permutations. In *Proceedings of the 10th ACM Conference on Electronic Commerce*. 71–78.

[17] M. Fahrbach, Z. Huang, R. Tao, and M. Zadimoghaddam. 2020. Edge-Weighted Online Bipartite Matching. In *Proceedings of the 61st Symposium on Foundations of Computer Science*. 412–423.

[18] J. Feywerda and J. Gest. 2016. *Location, Location: Refugee Resettlement and Integration Outcomes in the United States*. Mimeo.

[19] D. Freund and S. Banerjee. 2019. *Good Prophets Know When the End Is Near*. SSRN.

[20] C.-J. Ho and J. Vaughan. 2012. Online task assignment in crowdsourcing markets. In *Proceedings of the 26th AAAI Conference on Artificial Intelligence*, Vol. 26. 45–51.

[21] J. Hsu, J. Morgenstern, R. Rogers, A. Roth, and R. Vohra. 2016. Do Prices Coordinate Markets?. In *Proceedings of the 48th Annual ACM Symposium on Theory of Computing*. 440–453.

[22] R. Johari, V. Kamble, and Y. Kanoria. 2021. Matching while learning. *Operations Research* 69, 2 (2021), 655–681.

[23] I. Jones. 2015. Home Away From Home. *State Magazine* (Dec. 2015), 17–20.

[24] R. M. Karp, U. V. Vazirani, and V. V. Vazirani. 1990. An Optimal Algorithm for On-Line Bipartite Matching. In *Proceedings of the 22nd Annual ACM Symposium on Theory of Computing*. 352–358.

[25] T. Kesselheim, K. Radke, A. Tönnis, and B. Vöcking. 2013. An Optimal Online Algorithm for Weighted Bipartite Matching and Extensions to Combinatorial Auctions. In *Proceedings of the 21st Annual European Symposium on Algorithms*. 589–600.

[26] H. B. Leonard. 1983. Elicitation of Honest Preferences for the Assignment of Individuals to Positions. *Journal of Political Economy* 91, 3 (1983), 461–479.

[27] V. H. Manshadi, S. O. Gharan, and A. Saberi. 2012. Online Stochastic Matching: Online Actions Based on Offline Statistics. *Mathematics of Operations Research* 37, 4 (2012), 559–573.

[28] L. Martén, J. Hainmueller, and D. Hangartner. 2019. Ethnic Networks Can Foster the Economic Integration of Refugees. *Proceedings of the National Academy of Sciences* 116, 33 (2019), 16280–16285.

[29] S. R. Sinclair, F. Frujeri, C.-A. Cheng, and A. Swaminathan. 2022. *Hindsight learning in MDPs with exogenous inputs*. Working paper.

[30] K. T. Talluri and G. J. van Ryzin. 2004. *The theory and practice of revenue management*. Vol. 1. Springer.

[31] UNHCR. 2020. *Global Resettlement Needs 2021*. Technical Report. United Nations High Commissioner for Refugees.

[32] UNHCR. 2020. *Global Trends: Forced Displacement in 2019*. Technical Report. United Nations High Commissioner for Refugees.

[33] U Vazirani, V Vazirani, A Mehta, and A Saberi. 2005. Adwords and generalized on-line matching. In *Proceedings of the 46th Symposium on Foundations of Computer Science*.

[34] E. Vee, S. Vassilvitskii, and J. Shanmugasundaram. 2010. Optimal Online Assignment with Forecasts. In *Proceedings of the 11th ACM Conference on Electronic Commerce*. 109.

[35] A. Vera and S. Banerjee. 2021. The Bayesian Prophet: A Low-Regret Framework for Online Decision Making. *Management Science* 67, 3 (2021), 1368–1391.

[36] A. Vera, S. Banerjee, and I. Gurvich. 2021. Online Allocation and Pricing: Constant Regret via Bellman Inequalities. *Operations Research* 69, 3 (2021), 821–840.

APPENDIX

A DATA PREPROCESSING AND EMPLOYMENT PREDICTION

We use arrival data directly collected by HIAS, recent as of Summer 2020. We drop entries in the database missing crucial information: agent and case identifiers, the case's pool (free or with US ties), and the dates of birth, of allocation, and of arrival. We furthermore remove all cases that were allocated before March 2011 since our employment prediction uses local unemployment data, which is incomplete before then. This last step removes 3.5% of refugees from fiscal year 2014, and less than 1% of refugees from the subsequent fiscal years. Due to this removal, we do not include fiscal years before 2014 in our analysis, for which a higher percentage of arrivals would have been removed.

To predict employment scores, we train two separate LASSO models following the methodology of Ahani et al. [3], one model for free cases and one for tied cases. Training separate models is helpful since, as Ahani et al. note, tied cases find employment in different patterns due to existing support networks. The regression is trained on cases allocated in fiscal years 2011 to 2019, where the hyper-parameters are determined via cross validation before retraining on the whole time range. Note that training on years that we evaluate on would be problematic for evaluating how well the employment scores match the ground-truth employment. However, since our evaluations measure how well the matching algorithms do relative to the given employment scores, training on a wide range of years ensures that the employment scores are as accurate as possible.

B ALTERNATIVE POTENTIAL APPROACH

In Section 4.1, we presented a procedure for calculating potentials, termed Pot1, which was based on the shadow prices of a matching LP. We also outlined a second potential procedure, Pot2, in the body, which we formally define and justify here: the procedure is given in Algorithm 3, and it can be immediately plugged into PMB. Note

---

ALGORITHM 3: Pot2

Parameter: $k \in \mathbb{N}_{\geq 1}$, the number of trajectories per potential computation

Input: remaining capacities c, the index $t' + 1, \ldots, t$ of the cases in the current batch (see Algorithm 4), characteristics of cases arriving in the 6 past months

Output: a set of potentials p_ℓ for all affiliates ℓ

1 **for** $j = 1, \ldots, k$ **do**

2   for each $i = t + 1, \ldots, n$, set s_i and $\{u_{i,\ell}\}_\ell$ to the size and employment scores of a random, recently arrived case;

3   solve the following bipartite-matching LP:

$$
\begin{aligned}
\text{maximize} \quad & \sum_{i=t'+1}^{n} \sum_{\ell \in L} u_{i,\ell}\, x_{i,\ell} \\
\text{subject to} \quad & \sum_{\ell \in L} x_{i,\ell} = 1 && \forall i = (t'+1), \ldots, n \\
& \sum_{i=t'+1}^{n} s_i\, x_{i,\ell} \leq c_\ell && \forall \ell \in L \quad (*) \\
& x_{i,\ell} \geq 0 && \forall i = (t'+1), \ldots, n, \forall \ell \in L.
\end{aligned}
$$

4   for each ℓ, set p_ℓ^j to be the the minimal shadow price of the constraint $(*)$;

5 set $p_\ell \leftarrow (\sum_{j=1}^{k} p_\ell^j)/k$ for all ℓ;

6 **return** $\{p_\ell\}_{\ell \in L}$;

that the dual of the linear program in Line 3 looks as follows:

$$\begin{aligned}
\text{minimize} \quad & \sum_{i \in t'+1}^{n} y_i + \sum_{\ell \in L} p_\ell \\
\text{subject to} \quad & y_i \geq u_{i,\ell} - s_i\, p_\ell && \forall i = (t'+1), \ldots, n, \forall \ell \in L \\
& p_\ell \geq 0 && \forall \ell \in L.
\end{aligned}$$

Suppose that $s_i = 1$ for all $i \in N$, as we do in Section 4. Then, for any (non-fractional) optimal matching $x_{i,\ell}$ in the primal, and any set of optimal dual variables p_ℓ, it is well known (and follows from complementary slackness) that the two form a *Walrasian equilibrium*. That is, if we consider $u_{i,\ell}$ as case i's utility for being matched to affiliate ℓ, and if we imagine charging case i a price of p_ℓ for being matched to affiliate ℓ, the optimal matching matches each case i to an affiliate ℓ maximizing the profit $u_{i,\ell} - p_\ell$. Thus, if all trajectories perfectly predicted future arrivals, and if ties were broken in a specific way in each step, equipping PM with the potentials of Pot2 would lead to the optimal-employment matching.

Note that this justification also extends to batching (still assuming $s_i = 1$), since the Walrasian equilibrium shows that it is possible to maximize $u_{i,\ell} - p_i$ for all cases at once. Thus, if the trajectory perfectly anticipates future arrivals, each optimal hindsight matching maximizes the matching ILP allocating each batch in PMB.

When the shadow prices are chosen as the element-wise minimal set of shadow prices, work by Hsu, Morgenstern, Rogers, Roth, and Vohra [21] gives arguments that PM with Pot2 should be somewhat robust to tie breaking, assuming that employment scores satisfy a genericity condition. In addition, they show that, for large matching problems, this matching algorithm should also be robust if trajectories are drawn from the same i.i.d. distribution as the real set of future arrivals.[18]

Like the justification of the Pot2 potentials, this justification does not cleanly extend to cases of non-unit size. Furthermore, there is no immediate theoretical justification for averaging these prices over multiple trajectories, which has to be evaluated empirically.

---

[18]This mapping is not perfect since Hsu et al. show that capacities are not exceeded by much, rather than considering cascading effects of refugees switching affiliates once their most preferred affiliate is at capacity. This problem could be remedied by slightly reducing capacities in the computation of shadow prices.

C PSEUDOCODE FOR ALGORITHM PMB

ALGORITHM 4: PMB(Potential)

Parameter: a subroutine Potential to determine affiliate potentials

1 initialize the capacities c_ℓ for each affiliate ℓ;

2 $t_{last} \leftarrow 0$;   // index of last case in previous batch

3 **while** $t_{last} < n$ **do**

4   observe the size b of the current batch;

5      $t \leftarrow t_{last} + b$;   // index of last case in current batch

6   observe the case size s_i and the employment scores $\{u_{i,\ell}\}_\ell$ for all $i = (t_{last}+1), \ldots, t$;

7   call Potential() to define a potential p_ℓ for each affiliate ℓ;

8   let $\{\hat{x}_{i,\ell}\}$ be an optimal solution to the following bipartite-matching ILP with knapsack constraints:

$$
\begin{aligned}
\text{maximize} \quad & \sum_{i=t_{last}+1}^{t} \sum_{\ell \in L} (u_{i,\ell} - s_i \cdot p_\ell)\, x_{i,\ell} \\
\text{subject to} \quad & \sum_{\ell \in L} x_{i,\ell} = 1 && \forall i = (t_{last}+1), \ldots, t \\
& \sum_{i=t_{last}+1}^{t} s_i\, x_{i,\ell} \le c_\ell && \forall \ell \in L \\
& x_{i,\ell} \in \{0,1\} && \forall i = (t_{last}+1), \ldots, t, \forall \ell \in L.
\end{aligned}
$$

9   **for** $i = t_{last} + 1, \ldots, t$ **do**

10   allocate case i to unique affiliate ℓ where $\hat{x}_{i,\ell} = 1$;

11          $c_\ell \leftarrow c_\ell - s_i$;

12      $t_{last} \leftarrow t$;

D   ADDITIONAL EXPERIMENTS

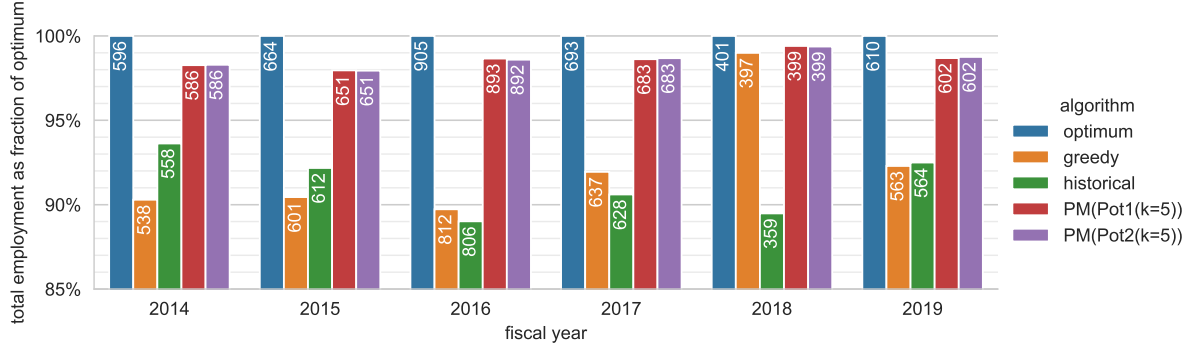## D.1   Employment Statistics for Non-Unit Cases without Batching

Fig. 13. Total employment obtained by different algorithms, with whole cases arriving rather than being split up. Capacities are the final capacities of the fiscal year. For the potential algorithms, employment is averaged over 10 random runs. The numbers in the bars denote the absolute total employment; the bar height indicates the proportion of the optimum total employment in hindsight.
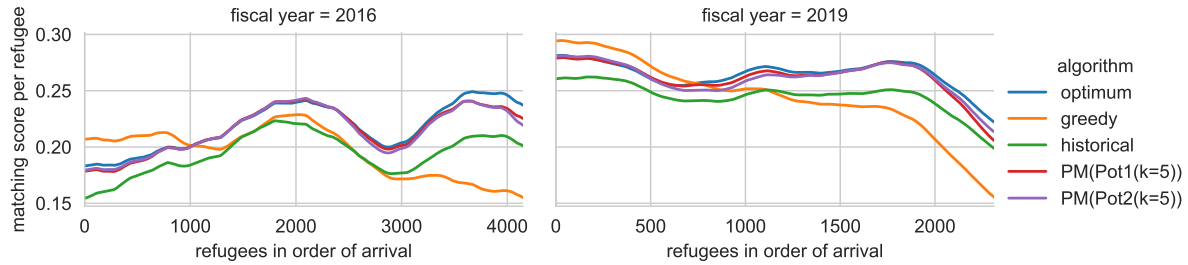

Fig. 14. Evolution of the per-refugee match score in order of arrival, for fiscal years 2016 and 2019 in the experiment of Fig. 13 (whole cases, final capacities). Match scores are smoothed using triangle smoothing with width 500.
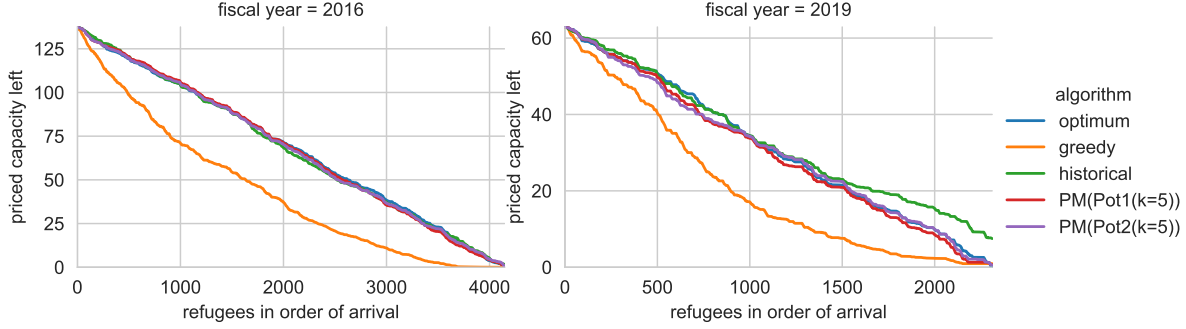
Fig. 15. Remaining priced capacity at the time of arrival of different refugees, for fiscal years 2016 and 2019 and one random run per algorithm in the experiment of Fig. 13 (whole cases, final capacities).

D.2 Timing

In the experiment of Section 6.1, we measure the running time for a call to PMB, which mainly consists of the time of sampling the k random trajectories, of computing the appropriate shadow prices for each trajectory, and of solving the matching ILP. This time will generally increase the more arrivals are expected in the remainder of the fiscal year since this determines the length of the trajectory and the size of the LP whose duals must be computed. Additionally, batches consisting of more cases should also increase running time since the matching ILP is larger for such batches.

Therefore, we benchmark the algorithms on fiscal year 2016, which saw the largest number of arriving cases (1 628). Within this fiscal year, we aim to pick a batch that is both early and large, which is a bit complicated by the fact that the earliest batches of the year are small. We choose a batch of 31 cases, which comes early enough that 1 486 of the 1 628 cases of the fiscal year come after it. This batch is the 17th largest out of the 168 batches of the year, and all preceding batches are less than half as many cases.

For this batch, we measure the running time for PMB with potentials Pot1 and Pot2, each with $k \in \{1, 5, 9\}$. Each reported running time is the average of 50 iterations and is measured on a 2017 MacBook Pro with a 3.1 GHz Dual-Core Intel Core i5 processor using Gurobi for solving LPs and ILPs:

Matching algorithm	Running time
PMB(Pot1($k = 1$))	0.5 s
PMB(Pot1($k = 5$))	2.4 s
PMB(Pot1($k = 9$))	4.0 s
PMB(Pot2($k = 1$))	0.5 s
PMB(Pot2($k = 5$))	2.6 s
PMB(Pot2($k = 9$))	4.6 s

Given that a resettlement agency needs to execute the matching algorithm at most once per day, these times are negligible. For low k and an optimized implementation, it might even be possible to support near real-time experimentation; for example, an *Annie*™ user could drag a slider to indicate the expected number of arriving refugees, and could see how this would impact the recommended allocation of the current batch.

As we mention in the conclusion of the paper, we anticipate that refugee arrival rates will substantially increase in the future. With this scenario in mind, we investigate how the runtime of potential algorithms scales for much larger arrival numbers than those currently encountered. Below, we measure the runtime of our algorithm

PMB(Pot2(k = 5)) for bootstrapped arrivals based on fiscal year 2016. In each experiment, we fix a number of arrivals and measure the running time for allocating a simulated batch containing a $\frac{1}{52}$ fraction of the total arrivals, arriving right at the beginning of the fiscal year when the largest LPs must be solved. We proportionally scale the affiliate capacities based 110% of the historically allocated capacity, and average each running time over 3 runs.

Number of arriving refugees	Running time
10 000	9 s
20 000	25 s
40 000	70 s
80 000	4 min
125 000	12 min

These numbers suggest that the runtime of the algorithm does not scale unreasonably fast in the number of arrivals. It would even be feasible to allocate the planned capacity of 125 000 refugees for the entire US through our algorithm. Should runtime considerations become a problem, computation could be easily sped up by computing the duals for the different trajectories in parallel.

D.3 Length of sampling window

Throughout the paper, our potential mechanisms simulate trajectories by bootstrapping cases that were allocated by the algorithm within the previous six months, a duration to which we will refer as the algorithm's *sampling window*.

   The sampling window of six months was *not* selected based on data, which is important to avoid overfitting the data. Instead, we chose the duration of six months as a reasonable middle ground, which we hoped would be short enough so that the algorithm may react to a sudden shift within the same matching period and simultaneously would be long enough that, at any point, the algorithm would have a sufficiently diverse and representative set of recently allocated cases.

   Though our concern for overfitting compels us not to optimize the sampling window for use in our main results, we can explore in Fig. 16 how different sampling windows would have impacted our algorithm's performance. As should be expected, a sampling window of length 0 harms employment performance since, in this case, trajectories are bootstrapped form only the cases in the current batch and thus prices are of little value. Already the second-smallest sampling window we evaluated, 63 days, leads to much better employment. Indeed, in the fiscal years 2014, 2017, and 2018, this performance is already close to optimal, whereas in the remaining three fiscal years, larger sampling windows further increase the employment performance by a fair amount. To our surprise, very long sampling windows between 1.5 and 3 years do not seem to lead to lower performance than intermediate sampling windows such as the one we chose. It seems that, despite dramatic shifts in refugee origin and admission policy, changes in refugee characteristics over time change little in the calculus of how valuable each affiliate is, which is in line with our observations in Appendix D.4. In light of these results, our own choice of 183 days seems reasonable, but we would cautiously correct this number upward, to perhaps one year. Even though we have not observed any case where very long sampling windows would be bad for employment, this finding based on only six years of data should not be overinterpreted—it is still plausible that future shocks in refugee characteristics will require a shorter sampling window to allow for faster adaptation.

D.4 Evolution of potentials by arrival numbers

Figure 17 shows how the number of arriving cases impacts the prices of different affiliates. The figure shows data for fiscal year 2016 and the potential Pot2; the corresponding plots for other fiscal years and for Pot1 look
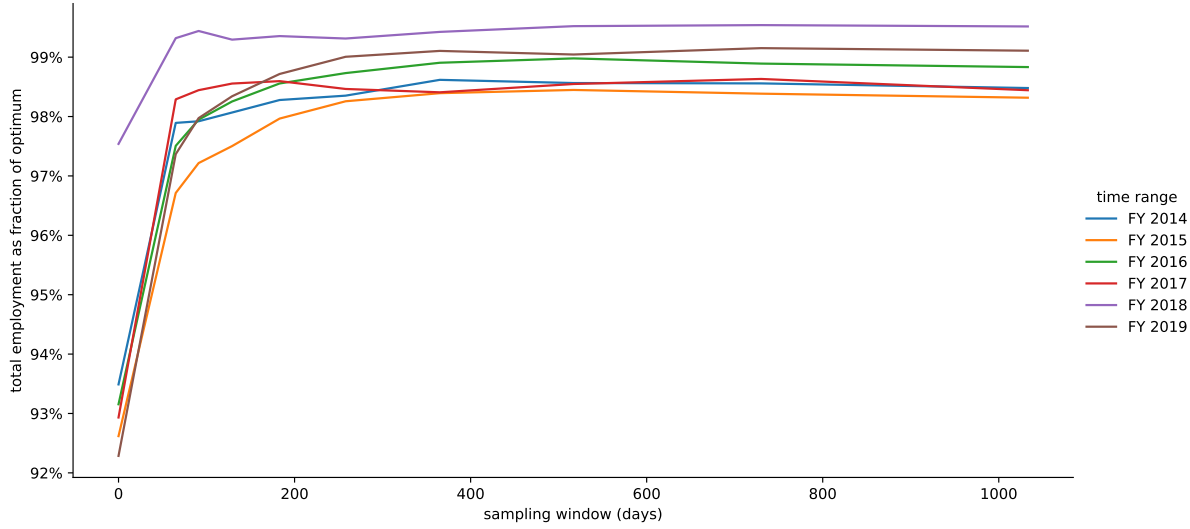
Fig. 16. Employment of algorithm PMB(Pot2($k = 5$)) as a function of the sampling window. Capacities are the final fisyear capacities, and the experiment is comparable to Fig. 4. Each datapoint is averaged over 10 random runs.

quite similar. For very short trajectories, all potentials are zero, which reflects that there is no scarcity of affiliate capacity and that greedy assignment is optimal. As the trajectory length increases, the potentials of more and more affiliates become positive. Generally speaking, affiliate potentials increase with the trajectory length, and they rarely cross, which means that, even across very different arrival numbers, which affiliates' capacities are most valuable remains stable and roughly corresponds to the average employment over all refugees.

One affiliate deviates from this pattern across fiscal years, namely the one represented by the solid red line in Fig. 17. This affiliate has a low potential up to the historical trajectory length, but, for large trajectory lengths, its potential increases steeply and becomes the largest overall. This affiliate stands out in three ways: (1) a large number of refugees (in 2016, amounting to 82% of the affiliate's capacity) are compatible with only this affiliate, (2) these tied refugees have very high per-person employment prospects (36%), and a vast majority of the other cases are not compatible with this affiliate. Thus, when the trajectory length is short, all tied cases fit in the affiliate and the compatibility constraints mean that there is little competition for the remaining capacity among non-tied cases. This changes when the trajectory length is long enough that the number of tied refugees exceeds the capacity: given that not all tied refugees can be matched, a high potential ensures that the slots are used for the refugees with largest employment opportunities. While the above observations are technically interesting and might be useful in other applications of online matching, we stress that they refer to extreme scenarios in which much more refugees arrive than in any of our data, and in which the affiliate's capacity is dramatically at odds with the demand from tied cases.

D.5 Robustness checks using other arrival sequences

Figure 4 shows that the potential algorithms consistently achieve between 98% and 99% of the optimal employment in hindsight, a clear step up from the greedy matching and the matching historically chosen by HIAS. That being said, the evaluation in the main text is limited to six different arrival sequences, one per fiscal year observed.
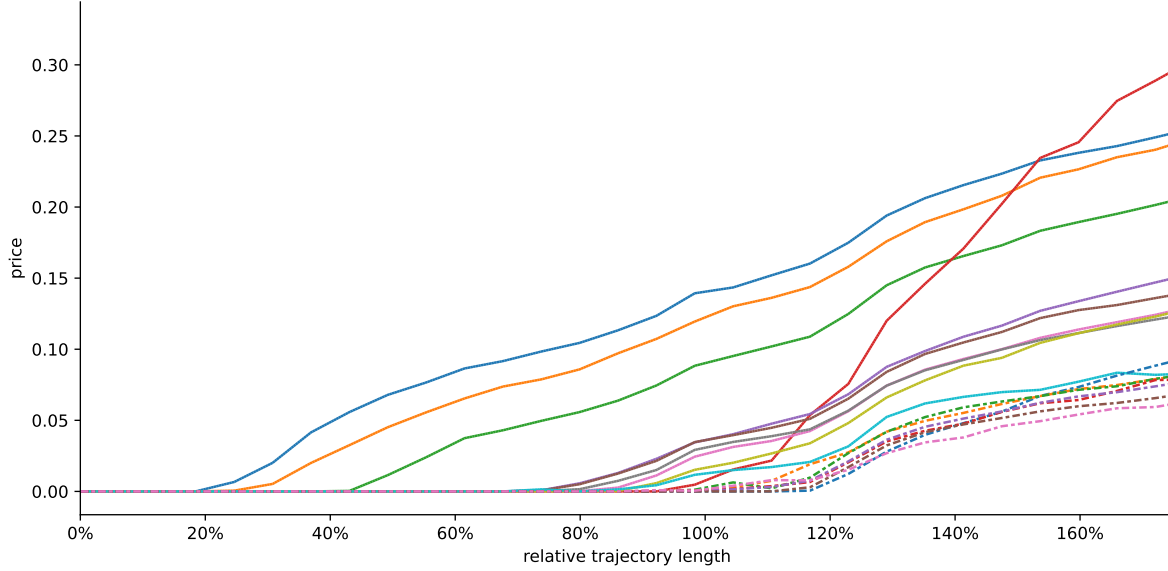
Fig. 17. At the beginning of fiscal year 2016, the Pot2 potentials of different affiliates depending on the length of simulated trajectory. The trajectory length is measured as a number of cases and given as a percentage of the actual number of case arrivals in fiscal year 2016. Each line corresponds to an affiliate, which is not identified due to privacy considerations. Trajectories are bootstrapped over the fiscal-year 2016 arrivals, and the lines averaged over 50 random trajectories. Five affiliates with capacities below 15 are omitted.
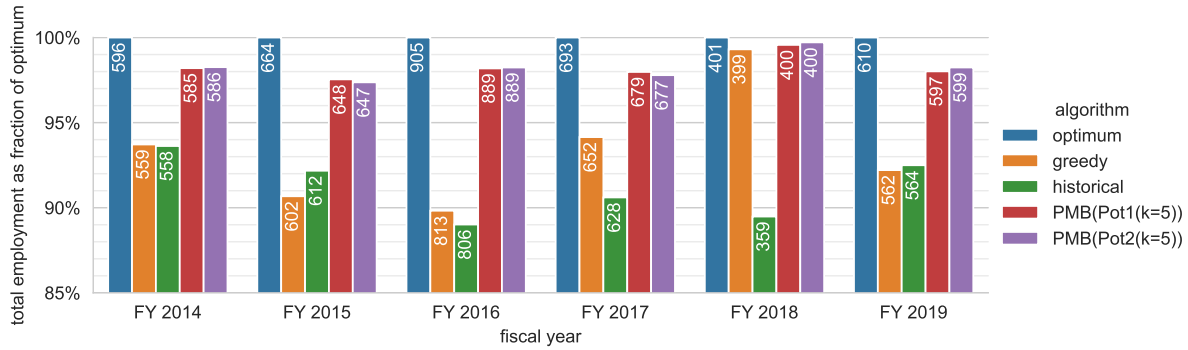
Fig. 18. Version of the experiment in Fig. 4, where each fiscal year's arrival sequence is reversed. For the potential algorithms, total employment is averaged over 10 random runs.

To demonstrate the robustness of our findings, we will repeat the experiment with two other forms of arrival sequences, derived from but not identical to the historical arrival sequences.

First, in Fig. 18, we reverse the order in which each fiscal year's arrivals are presented to the algorithm. Note that the optimal and historical matching do not change in this case. Remarkably, the performance of the greedy
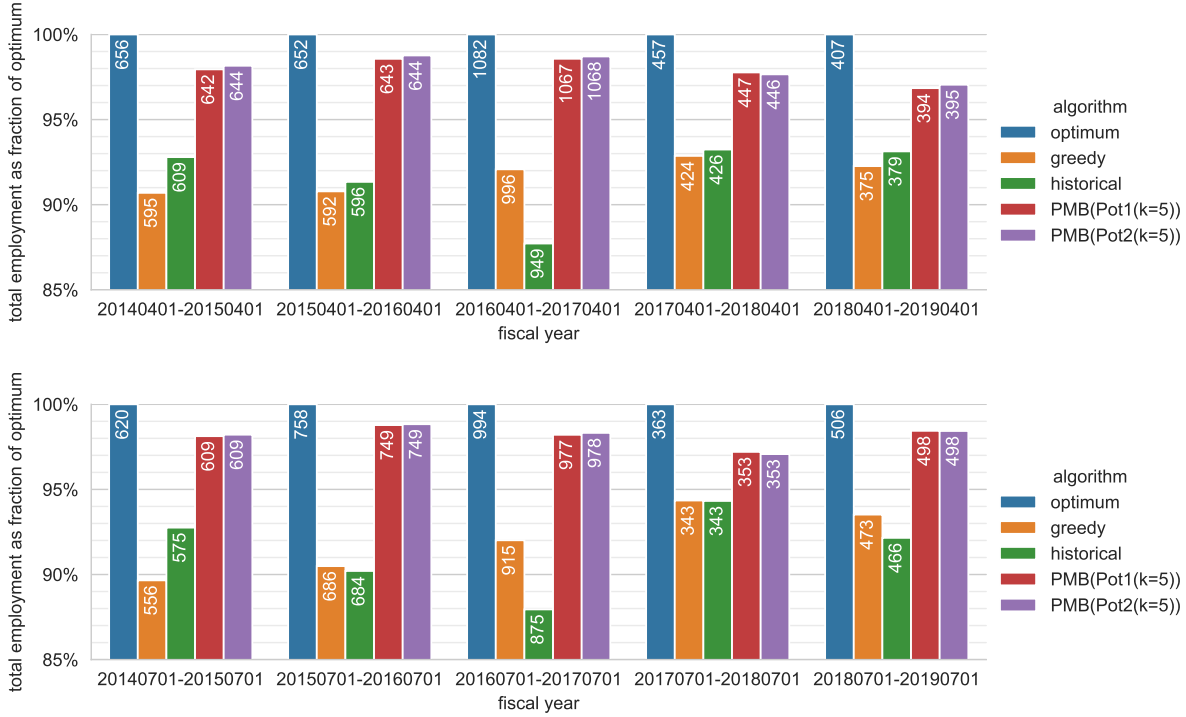
Fig. 19. Total employment, where cases are not split and arrive in batches, but grouping arrivals into "fiscal years" lasting from April to March or from July to June rather than from October to September. Capacities are 110% of the historical allocation of these cases. For the potential algorithms, total employment is averaged over 10 random runs.

algorithm improves overall, by as much as 3.5% in 2014 and 1.7% in 2017. In contrast to greedy, the performance of the potential algorithms seems to change much less. While there seems to be a trend of the potential algorithms performing slightly better forward than backward, these decreases are clearly below one percentage point.

Second, in Fig. 19, we group historic arrivals not by which fiscal year (October to September) they arrive in, but make an assumption that the matching periods instead last from April to March or from July to June. Since the capacities are not derived from any official fiscal-year capacities, but just set to 110% of the historical arrivals during the matching period, the employment numbers of this experiment are not immediately comparable to the numbers displayed in other figures. Again, the potential algorithms clearly outperform the historical and greedy matching. In the matching periods April 2018–March 2019 and July 2017–June 2018, the employment of the potential algorithms slightly drops relative to what we saw in the main text's experiments, to around 97% of the optimum. In both of these periods, the optimal employment is particularly low, which most likely reflects low numbers of arriving cases and makes larger relative deviations more likely.

D.6 Employment statistics if capacities are changed during the year

For the fiscal years 2017 and 2018, we run a version of the experiments in Section 7.1, in which the capacities are actually reduced at the time they were revised. For the greedy algorithm and the potential algorithms, this means that they are initially run using the initially announced capacities (and the potential algorithms expect 91% of

the initial total capacity to arrive). At the time the capacity was revised, the capacity for the greedy algorithm and the potential algorithms is updated to the revised capacity. If the revised capacity of an affiliate lies below what the algorithm already used before the point of revision, the capacity is frozen at current occupancy. As we argue in the body of the paper, this gives the greedy algorithm an unfair advantage and makes the resulting total employment hard to compare. The historical matching is roughly comparable since HIAS also started out aiming for the full capacities and later aimed for the revised capacities. However, the historical matching in 2017 actually made use of a fairly large end-of-year increase in capacity, which the other algorithms do not have access to, and it profits from the fact that HIAS influenced the change in capacity to match its prior allocation decisions. As a reference, we show optimal matchings in hindsight both for the initial and for the revised capacities.
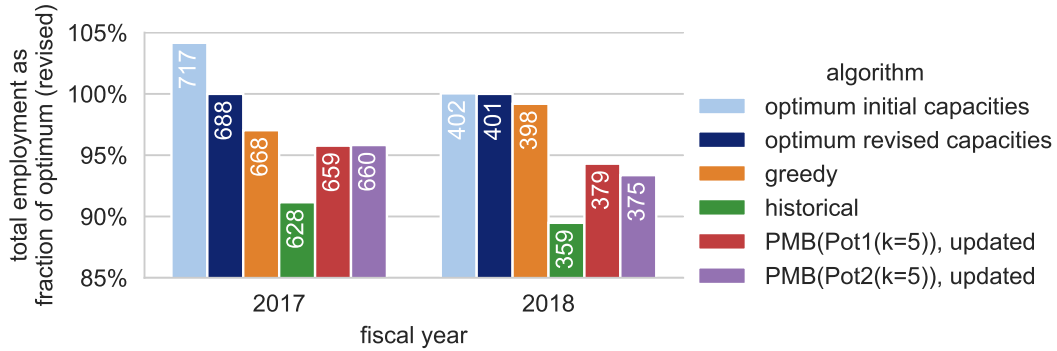

Fig. 20. Total employment, where cases are not split up and arrive in batches. Dynamic algorithms start respecting initial capacities, but capacities are changed at time of revision (except for historical). The potential algorithms do not have access to the true number of arriving cases but assume that the arriving refugees amount to 91% of the initial, then the revised capacity. For potential algorithms, a single random run is shown.

D.7 Percentage of Matched Refugees

Here, we report the percentage of refugees (i.e., cases weighted by case size) that gets matched to a real affiliate rather than the unmatched affiliate $\perp$. In our implementation of PMB (Algorithm 4), for a small constant ϵ, we add a weight of $\epsilon \cdot s_i$ for all variables $x_{i,\ell}$ indicating that a case i is matched to an affiliate $\ell \neq \perp$, with the intention of breaking ties between optimal solutions in favor of solutions that don't leave more cases unmatched than necessary. This is particularly relevant since some cases, for example those consisting of unaccompanied minors, have an employment score of zero for all affiliates. This added constant (and the equivalent modification to PM) ensures that whether these cases ever get matched does not depend on the implementation of the ILP solver. Since we implement the greedy algorithm by setting potentials in PMB to zero, the same holds true for greedy. Finally, a similar issue arises for the optimum matching in hindsight. For this, we optimize total employment without further constraints on the number of matched refugees, but then greedily add cases with zero unemployment where they fit throughout the year.

In particular, this means that the optimum is not an upper bound on how many people are matched. Instead, it is the historical matching that matches 100% of refugees, but this comparison is not on equal terms since the historical matching conforms to final capacities that were specifically increased to fit all refugees and since it ignores some case–affiliate incompatibilities.

For the setting of online bipartite matching, we find in Fig. 21 that the potential algorithms are roughly on par with the greedy algorithm when it comes to matching many refugees. The fact that both greedy and potential

algorithms tend to leave a single-digit percentage of refugees unmatched can probably be explained by the many cases with family ties, which can only go in a single affiliate and have to remain unmatched if this affiliate is full, which is hard to avoid without knowing future arrivals. The optimum-hindsight matching does not have this problem, and therefore matches around 2% of refugees more on average. The remaining gap to 100% can be explained by the greedy matching of zero-employment cases, by the fact that the case–affiliate incompatibilities might not allow to match all cases (which the historical matching is not bound by), and by genuine trade-offs between matching more refugees and obtaining higher total employment.
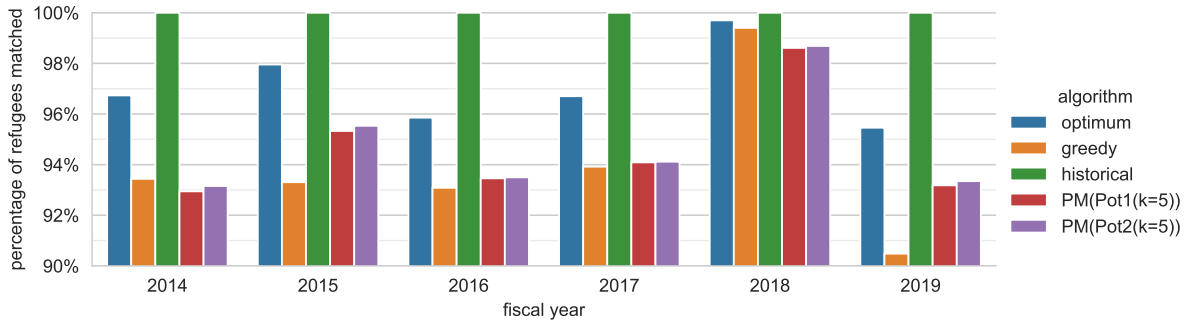

Fig. 21. Match percentages for the experiment in Fig. 1 (split cases, final capacities).

As for the total employment, the addition of non-unit case sizes (Fig. 22) and of batching (Fig. 23) barely has any effect on the percentage of matched refugees.
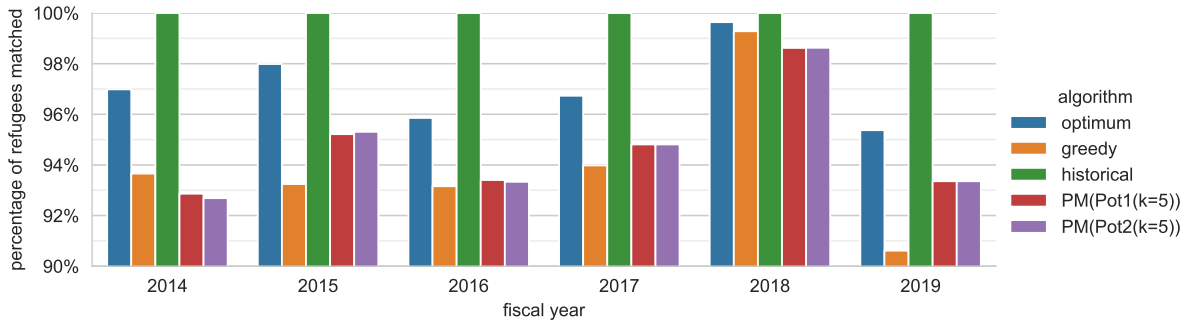

Fig. 22. Match percentages for the experiment in Fig. 13 (whole cases, final capacities).
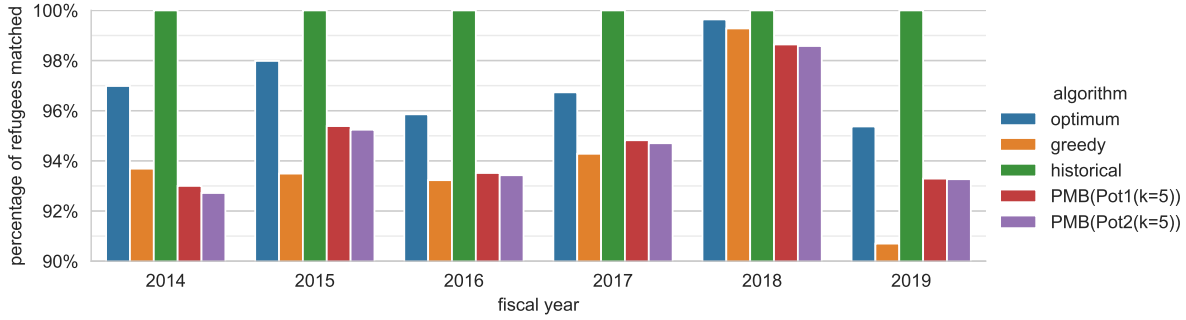
Fig. 23. Match percentages for the experiment in Fig. 4 (whole cases, batching, final capacities).

In Fig. 24, the percentage of matched refugees by the optimum matching and the greedy algorithm is lower in most years, which is due to the initial capacities being strictly tighter than the final capacities on all years other than 2017 and 2018. The potential algorithms follow the same trend, but visibly match particularly few refugees in fiscal years 2017 and 2018. Since these algorithms vastly overestimate the arrival numbers due to the overly large capacities, they leave some refugees unmatched in the expectation that this will benefit (spurious) later arrivals.
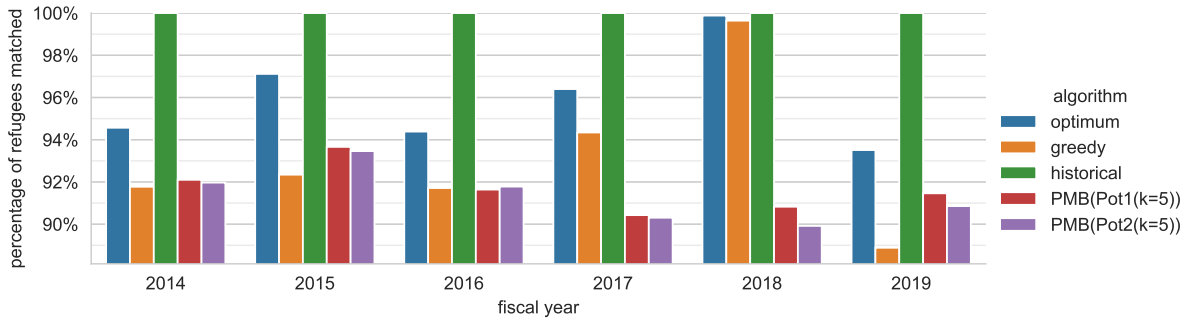

Fig. 24. Match percentages for the experiment in Fig. 6 (whole cases, batches, initial capacities, potential algorithms do not know *n*).