Summer Internship Matching with Funding Constraints

Haris Aziz
UNSW Sydney and Data61 CSIRO
Sydney
haris.aziz@unsw.edu.au

Anton Baychkov
University of Sydney
Sydney
abay3963@uni.sydney.edu.au

Péter Biró
Hungarian Academy of Sciences
Budapest
peter.biro@krtk.mta.hu

ABSTRACT

We present a novel model that captures matching markets for summer internships at universities and other organizations that involve funding constraints. For these markets, we show that standard results from the literature such as the existence of stable matchings do not extend and in fact checking whether a stable matching exists is NP-complete which answers an open problem. Because of these challenges, we investigate how far stability requirements can be satisfied. One of our contributions is presenting a polynomial-time algorithm that satisfies a weaker notion of stability and allocates the budget in a fair manner.

KEYWORDS

Matching under preferences; economic paradigms

ACM Reference Format:
Haris Aziz, Anton Baychkov, and Péter Biró. 2020. Summer Internship Matching with Funding Constraints. In *Proc. of the 19th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2020), Auckland, New Zealand, May 9–13, 2020,* IFAAMAS, 8 pages.

1 INTRODUCTION

Centralized two-sided matching market algorithms have received immense success in several application domains including matching students to schools, residents to hospitals, and projects to workers.[1] We present a novel matching market model that we refer to as the summer intern matching market. The model captures the matching of student applicants to projects proposed by supervisors in the internship program. A distinctive feature of the model is that in order for an applicant to be assigned to any project, a certain amount of money needs to be contributed from the project supervisors' funds.

Our problem is inspired by summer intern research programs in our country. It is common for undergraduate students to undertake research projects over the summer. Each project is supervised by one or more members of the faculty, with many offering multiple projects. Even though the projects may be discounted by contributions from the faculty, supervisors are often required to contribute to the funding of these positions, from their personal research budget. Alternatively, they could be constrained by the amount of time they can allocate to supervision. These supervisor-side constraints mean that not all projects can be funded.

[1]For an overview of real-life matching markets, please see http://www.matching-in-practice.eu.

Proc. of the 19th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2020), B. An, N. Yorke-Smith, A. El Fallah Seghrouchni, G. Sukthankar (eds.), May 9–13, 2020, Auckland, New Zealand. © 2020 International Foundation for Autonomous Agents and Multiagent Systems (www.ifaamas.org). All rights reserved.

Just as the standard hospital resident matching models do not just apply to matching of doctors to hospitals [22], our model also does not just apply matching of student interns. It applies to any two-sided matching model in which very widely applicable budget requirements and budget constraints are involved. For example, our problem also models hiring scenario in which different teams have their own budgets and they want to hire employees. Certain employee roles could sit across various teams. In that case, multiple teams can pool in their money to fund joint positions.

The budget constraints that we consider lead to interesting research challenges. A stable matching may not exist and the standard deferred acceptance algorithm does not work for our problem.

Contributions. In this paper, we formalize the summer internship problem with budgets. It falls within the class of models that matches applicants, projects, and supervisors. The main characteristic of our problem is that supervisors have budgets that they can spread across their projects. The model is more general than a widely-used matching model called hospital resident matching with regions in which the hospitals are partitioned into regions and regions have upper capacities [13].

We first prove that a stable matching may not exist for our model. Furthermore, even checking whether a (strongly) stable matching exists is NP-complete. The statement not just applies to our matching model but also simultaneously applies to several other matching models with distributional constraints. For a well-studied model concerning disjoint regions [13], the complexity question was open. In view of these challenges, we pursue a weaker notion of stability.

We then present a strongly polynomial-time algorithm to compute a weakly stable matching. Our algorithm uses as an oracle an algorithm based on network flows to repeatedly check whether a given matching is feasible or not. We also prove several other properties of the algorithm.

We then provide a normative criterion for fair ex-post allocation of supervisor funding among projects, and a polynomial-time algorithm to find the fairest allocation. Combined with our polynomial-time algorithm for finding a weakly stable matching, we present a compelling approach for finding a desirable solution for the summer internship problem that appeals to both stability and fairness requirements.

2 RELATED WORK

The literature on two-sided matching was inspired by the seminal paper of Gale and Shapley [8] who considered matching markets that match students to schools and hospitals to residents. The paper has spawned richer matching models and resulted in new algorithmic work (see e.g., Manlove [20]). Our work is an extension of these models and falls in the general umbrella of matching markets with various kinds of distributions constraints (see e.g., [3, 6, 7, 14, 18]).

Our concept of supervisor-feasibility is a type of feasibility constraint, as defined by Kamada and Kojima [15]. Thus, their notions of stability also apply in our model. In our paper, we focus on computational results such as establishing NP-completeness or polynomial-time solvability of stable matchings. Our model also has the additional dimension of budget allocations for which we explore fairness concepts as well as algorithms to divide the method fairly.

Our model bears some similarities with the hospital-resident matching problem with regional constraints [4, 5, 9, 13]. In these region-based problems, at most a certain number of students can be selected from given regions. On the other hand, in the summer internship problem, a supervisor's budget is divisible and can be spread partially over all of her projects. Even if regions can be overlapping, region based constraints cannot capture the feasibility constraints in our model. If the regions are disjoint (as studied by Kamada and Kojima [13]), then the region-based model is a special case of our model. The general setting with region constraints has not seen many positive results, and the more constrained hierarchical regions as studied by Goto et al. [9] can neither replicate, nor be replicated by a set of supervisors.

Abraham et al. [1] considered a different model for student-project allocation. Notable differences in their model include: (1) no project has multiple supervisors, (2) each supervisor has a universal priority list over students which is not project specific, and (3) a supervisor has a rigid capacity constraint for the number of projects to supervise. Our model allows supervisors to explore more efficient outcomes by pooling in their budgets to host a student. The universal priority list of each supervisors makes the model of Abraham et al. [1] much more restricted and different from our model.

Two other recent models are similar to our setting. Goto et al. [10] introduce the Student-Project-Room matching problem. Rooms are indivisible, and at most one room can be allocated to each project. Ismaili et al. [12] extend this to a more general Student-Project-Resource allocation problem. Resources are still indivisible, but there is no longer a restriction imposed on the number of resources that can be allocated to each project. Our model is distinct from both of these, as it allows the resources (in our case, supervisor budgets) to be divisible. This divisibility allows for better computational results. For instance, verifying the feasibility of a matching, and finding a weakly stable (and thus non-wasteful) matching can be done in polynomial time.

There is also work on matching with budget constraints (see e.g., Ismaili et al. [11], Kawase and Iwasaki [16, 17]). The models considered in these papers are different in several respects. For example, hospitals have additive utilities and each hospital gives monetary compensation to doctors.

3 MODEL

Let A be a finite set of applicants, and P a finite set of projects. Each applicant $a \in A$ has a strict preference list $\succ_a$ that ranks the subset of projects that a finds acceptable. Each project $p \in P$ has a preference list $\succ_p$ over the subset of applicants that p finds acceptable, and a maximum capacity c_p.

Furthermore, let S denote the set of project supervisors. Each supervisor $s \in S$ has a list of projects P_s that they supervise, and is endowed with a budget (e.g. quantity of funds) q_s that they can allocate among those projects. We assume that these budgets are infinitely divisible and that each applicant requires one unit of funding. Further, we assume that these endowments are publicly known, and thus supervisors cannot strategise by misreporting their budgets. Additionally, denote the list of supervisors for project p by S_p.

We say that an applicant $a \in A$ is matched to project $p \in P$ if $(a, p) \in M$. A **matching** M is a subset of $A \times P$ that satisfies the following conditions:

- Each applicant is matched to at most one project (for all $a \in A$, $|\{(a,p) \in M : p \in P\}| \leq 1$), and a finds the project they are matched to acceptable.
- The number of applicants matched to any project does not exceed that project's capacity (for all $p \in P$, $|\{(a,p) \in M : a \in A\}| \leq c_p$), and p finds all applicants matched to it acceptable.

We use $M(a)$ to refer to the project that applicant a is matched to ($M(a) = \emptyset$ if a is unmatched). Meanwhile, $M(p)$ denotes the set of applicants matched to p.

Let $x_{s,p}$ be the amount of funds a supervisor s allocates to project p. We call a matching M **feasible** (or supervisor-feasible) if there exists a set $\{x_{s,p}\}_{s \in S, p \in P_s}$ that satisfies the following conditions:

- $x_{s,p} \geq 0$ for all $s \in S, p \in P_s$
- Every project receives one unit of funding for each applicant matched to it: $\sum_{s \in S_p} x_{s,p} = |M(p)|$ for all $p \in P$
- Supervisors do not exceed their endowment: $\sum_{p \in P_s} x_{s,p} \leq q_s$ for all $s \in S$

We call any set $\{x_{s,p}\}_{s \in S, p \in P_s}$ that satisfies the above conditions for matching M a **feasible funding allocation**. The mathematical model presented exactly captures the student research internship program in our university: each supervisor can be part of multiple internship project proposals but does not necessarily have the funding to contribute to all of them.

Example 3.1 (Summer Internship Problem). Consider the following instance of the summer internship problem with 2 applicants 2 supervisors, and 2 projects.

$$\begin{array}{ccc} A = \{a_1, a_2\} & \succ_{a_1}: p_2, p_1 & \succ_{a_2}: p_1, p_2 \\ P = \{p_1, p_2\} & \succ_{p_1}: a_1, a_2 & \succ_{p_2}: a_2, a_1 \\ S = \{s_1, s_2\} & P_{s_1} = \{p_1, p_2\} & P_{s_2} = \{p_2\} \\ q_{s_1} = 0.7 & q_{s_2} = 0.5 & c_{p_1} = c_{p_2} = 1 \end{array}$$

The only three feasible matchings are the empty matching and the two matchings in which some applicant is matched to project p_2. The reason no one can be matched to project p_1 is that p_1 has a sole supervisor s_1 who does not have sufficient funding to fund p_1. On the other hand, the combined funding of s_1 and s_2 is more than 1 for project p_2 so p_2 can be funded. A feasible funding allocation x is where s_1 contributed half of the budget of project p_1 and s_2 contributes the rest: $x_{s_1,p_2} = 0.5, x_{s_2,p_2} = 0.5$.

4 STRONG STABILITY

An applicant-project pair (a, p) is a **blocking pair** for matching M if:

- applicant a prefers project p to the project they are currently matched to: $p \succ_a M(a)$, and
- either:
  - p is under capacity and finds a acceptable: $|M(p)| < c_p$ and $a \succ_p \emptyset$; or
  - p prefers a to one of its currently matched applicants: $\exists\, a' \in M(p)$ such that $a \succ_p a'$

Definition 4.1 (Strong Stability). We call a matching M strongly stable if for any blocking pair (a, p) for matching M, the following two conditions are satisfied:

- $a' \succ_p a$ for all applicants $a' \in M(p)$
- The matching $M' = (M \cup \{(a, p)\}) \setminus \{(a, M(a))\}$ is not feasible.

The first condition implies that we only allow the existence of blocking pairs involving a project p that is under its maximum capacity c_p. The second condition implies that even if slot is free at project p, adding applicant a to project p will result in a distributional constraint being violated so that $M' = (M \cup \{(a, p)\}) \setminus \{(a, M(a))\}$ is not feasible. Thus, we allow blocking pairs that cannot be satisfied without violating our feasibility constraint to exist in a strongly stable matching.

Our first observation is that for our problem, a strongly stable matching does not exist. This follows from the observation that our model is more general than the hospital resident setting with disjoint regions [13]. We provide an adaptation of an example (Example 1) of Kamada and Kojima [15] for the sake of completeness.

Example 4.2 (A strongly stable matching does not necessarily exist for the summer internship problem). Consider the following instance of the summer internship problem:

$$A = \{a_1, a_2\} \qquad \succ_{a_1}: p_2, p_1 \qquad \succ_{a_2}: p_1, p_2$$
$$P = \{p_1, p_2\} \qquad \succ_{p_1}: a_1, a_2 \qquad \succ_{p_2}: a_2, a_1$$
$$S = \{s\} \qquad P_s = \{p_1, p_2\} \qquad q_s = c_{p_1} = c_{p_2} = 1$$

It is easy to see that $|M| \leq 1$. If both applicants are unmatched then (a_1, p_1) forms a blocking pair. Suppose without loss of generality that a_1 is matched in the stable matching. If $M = \{(a_1, p_1)\}$ then (a_1, p_2) is a blocking pair that doesn't satisfy the second condition of strong stability. If $M = \{(a_1, p_2)\}$ then (a_2, p_2) is a blocking pair that doesn't satisfy the first condition of strong stability. Thus every feasible matching admits a blocking pair that is not permitted under strong stability, and is therefore not strongly stable.

Below we show that the problem of deciding the existence of a strongly stable matching is NP-complete. We reduce for Restricted MAX-SMTI, the problem of deciding whether there exists a complete stable matching for the stable marriage problem with incomplete lists and ties under the restriction that the preferences of the men are strict, and the preference list of each woman is either strict or consists solely of a tie of length two [21]. First, we introduce an instance that will serve as the core of the construction imitating an indifferent woman.

Example 4.3 (An instance with two strongly stable matchings, covering different agents). The instance consists of a preference cycle involving four applicants and four projects as follows.

$$A = \{a_1, a_2, a_3, a_4\}$$
$$\succ_{a_1}: p_2, p_1 \qquad \succ_{a_2}: p_3, p_2$$
$$\succ_{a_3}: p_4, p_3 \qquad \succ_{a_4}: p_1, p_4$$
$$P = \{p_1, p_2, p_3, p_4\}$$
$$\succ_{p_1}: a_1, a_4 \qquad \succ_{p_2}: a_2, a_1$$
$$\succ_{p_3}: a_3, a_2 \qquad \succ_{p_4}: a_4, a_3$$
$$S = \{s_1, s_2, s_3\} \qquad P_{s_1} = \{p_1, p_3\}$$
$$P_{s_2} = \{p_2\} \qquad P_{s_3} = \{p_4\}$$
$$q_{s_1} = q_{s_2} = q_{s_3} = 1 \qquad c_{p_1} = c_{p_2} = c_{p_4} = c_{p_4} = 1$$

One can check that there are two strongly stable matchings: $M_1 = \{(a_1, p_1), (a_2, p_2), (a_4, p_4)\}$ and $M_2 = \{(a_2, p_2), (a_3, p_3), (a_4, p_4)\}$. First, let us show the stability of M_1 (the argument is similar for M_2 by symmetry). The only blocking pairs for M_1 are (a_2, p_3) and (a_3, p_3), but adding any of these pairs to M_1 would make the matching infeasible. To show that M_1 and M_2 are the only strongly stable matchings we observe first that a_2 has to be matched to p_2, and a_4 has to be matched to p_4 by symmetric reasons (so we prove only the former fact). Note that a_2 cannot be unmatched, as she would block the matching with p_2. Suppose now for a contradiction that a_2 is matched to p_3. Then p_1 must be unfilled by feasibility, thus a_4 has to be matched to p_4 and a_3 remains unmatched and so forms a blocking pair with p_3, a contradiction. So we showed that (a_2, p_2) and (a_4, p_4) must be contained in any strongly stable matching. Finally we shall note that both p_1 and p_3 cannot remain unfilled, as (a_4, p_1) would form a blocking pair. Therefore, in a strongly stable matching we must also have either (a_1, p_1) or (a_3, p_3), resulting in M_1 and M_2, respectively.

Using the example as a gadget, we will show that deciding whether an instance of our problem has a strongly stable solution is an NP-complete problem.

Theorem 4.4. *Checking the existence of a strongly stable matching is NP-complete, even if all the supervisors have capacity one and each is responsible for at most two distinct projects.*

Proof. Given a solution, we can check whether it is strongly stable by considering each potential blocking pair in polynomial time, so the problem is in NP. For proving NP-hardness, we reduce from the special version of the MAX-SMTI problem [21]. Here, we are given an instance of a stable marriage problem with incomplete lists and ties, where $U = \{u_1, u_2, \ldots, u_n\}$ is the set of men, each having a strict preference list over the women acceptable for him, and $W = W^s \cup W^t$ is the set of women, where $W^s = \{w_1, \ldots, w_k\}$ are the women with strict preference lists and $W^t = \{w_{k+1}, \ldots, w_n\}$ are the women, where each has a single tie of length two in her preference list (i.e., she finds two men acceptable, and she is indifferent between them). A matching is said to be weakly stable if it is not blocked by a pair where both parties strictly prefer each other to their current partners. Let us denote the restricted instance of SMTI by I, where the problem of deciding the existence of a complete weakly stable matching is NP-complete [21].

We construct the corresponding internship problem I' as follows. For every man u_i we create a project p_i with a single supervisor s_i

each with capacity one. We denote this set of projects by P^u. For every woman in $w_j \in W^s$ we create an applicant a_j, and we denote this set of applicants by A^s. Here the preference list of a_j over the projects in I' is identical to the preference list of w_j over the men in I. Now, for each woman $w_j \in W^t$, we introduce a gadget G^j that is identical to the instance in Example 4.3, so let G^j be constructed as follows.

$$A^j = \{a^j_1, a^j_2, a^j_3, a^j_4\}$$
$$\succ_{a^j_1}: p^j_2, p^j_1 \qquad \succ_{a^j_2}: p^j_3, p^j_2$$
$$\succ_{a^j_3}: p^j_4, p^j_3 \qquad \succ_{a^j_4}: p^j_1, p^j_4$$
$$P^j = \{p^j_1, p^j_2, p^j_3, p^j_4\}$$
$$\succ_{p^j_1}: a^j_1, a^j_4 \qquad \succ_{p^j_2}: a^j_2, a^j_1$$
$$\succ_{p^j_3}: a^j_3, a^j_2 \qquad \succ_{p^j_4}: a^j_4, a^j_3$$
$$S = \{s^j_1, s^j_2, s^j_3\} \qquad P_{s^j_1} = \{p^j_1, p^j_3\}$$
$$P_{s^j_2} = \{p^j_2\} \qquad P_{s^j_3} = \{p^j_4\}$$
$$q_{s^j_1} = q_{s^j_2} = q_{s^j_3} = 1 \qquad c_{p^j_1} = c_{p^j_2} = c_{p^j_4} = c_{p^j_4} = 1$$

Furthermore, if $w_j \in W^t$ had preference list $[u_{i_1}, u_{i_2}]$ then we append p_{i_1} to the end of the preference list of a^j_1 and we append p_{i_2} to the end of the preference list of a^j_3. Likewise, we adjust the preference list of $p_i \in P^u$ as follows: for every $w_j \in W^s$, we replace w_j with a_j, whilst if $w_j \in W^t$ then we replace w_j with either a^j_1 or a^j_3 according to whether u_i was the first or the second man in the tie of w_j. Finally, we add a gadget G^*, which is a copy of the unsolvable instance in Example 4.2, together with an additional applicant a^* who accepts one of the projects, say p^*, in G^*. Let a^* be the most preferred applicant by the projects in G^*, so including this applicant in G^* will turn the instance solvable by assigning a^* to p^* and leaving the other applicants in G^* unmatched. To link $G^* \cup a^*$ with the rest of I' we put all the projects in P^u ahead of p^* in the preference list of a^* and we also append a^* to the end of the preference list of each $p_j \in P^u$.

We will show that I has a complete weakly stable matching if and only if I' has a strongly stable matching. Let us suppose first that M is a complete stable matching of I, we construct a strongly stable matching M' of I' as follows. For every $(u_i, w_j) \in M$, where $w_j \in W^s$ we simply add the corresponding pair (p_i, a_j) to M'. Suppose now that $(u_i, w_j) \in M$, where $w_j \in W^t$ and u_i is the first man in the single tie of w_j, so p_i is linked with a^j_1 in I'. Let then include (p_i, a^j_1) in M', together with the corresponding strongly stable matching M_2 of G^j that leaves a^j_1 unmatched, namely $M^j_2 = \{(a^j_2, p^j_2), (a^j_3, p^j_3), (a^j_4, p^j_4)\}$. Similarly, if u_i is the second man in the tie of w_j, then we include (p_i, a^j_3) in M', together with the corresponding strongly stable matching M_1 of G^j that leaves a^j_3 unmatched, namely $M^j_1 = \{(a^j_1, p^j_1), (a^j_2, p^j_2), (a^j_4, p^j_4)\}$. Finally, we add (a^*, p^*) to M'. The strong stability of M' is implied by the following reasons. Observe first that a^* cannot block with any project in P^u, since all of these projects are assigned with better applicants. The matching in gadget G^* is also stable internally, as well as in each gadget G^j. Finally, the weak stability of M implies the lack of the blocking pairs of form (a_j, p_i) for M', where $a_j \in A^s$ and $p_i \in P^u$.

In the other direction, let us assume that M' is a strongly stable matching for I' and we construct a complete weakly stable matching M for I as follows. First we note that a^* must be matched to p^* in M', as otherwise the separated gadget G^* would cause instability. But if a^* is matched to p^* then each project $p_i \in P^u$ must be assigned to an applicant better than a^*. Now, if $(p_i, a_j) \in M'$ for $a_j \in A^s$ then we add (u_i, w_j) to M (where $w_j \in W^s$), and if (p_i, a^j_1) or (p_i, a^j_3) belongs to M' then we add (u_i, w_j) to M (where $w_j \in W^t$). It is obvious that M is a complete matching in I, its weak stability is implied by the strong stability of M' as follows. Suppose for a contradiction that a pair (u_i, w_j) would be blocking for M (where $w_j \in W^s$), then the corresponding pair (p_i, a_j) would also block M', a contradiction. This completes the proof. □

Our hardness result is strong because it holds for a very restricted setting and also implies NP-completeness for the setting of Kamada and Kojima [13] that concerns disjoint regions, even if at most two hospitals belong to each region. The complexity of existence of strongly stable matching was also an open problem for the model of Kamada and Kojima [13].[2] Furthermore, this case can also occur when one hospital has a common upper quota for two different types of jobs, e.g., daytime and night shifts, or surgical and medical internship positions. Another motivating example is the Hungarian college admission scheme, where students can be admitted to a programme under two contracts, state-funded and privately-funded, and there is a common upper bound on them [5].

5 WEAK STABILITY

In view of the non-existence and NP-completeness of checking the existence of strongly stable matchings, one can consider a weaker stability criterion. Kamada and Kojima [15] proposed a weak stability concept for a setting that does not concern budgets but which has an abstract feasibility indicator function for any given matching. We present the definition in our terminology of applicants and projects.

Definition 5.1 (Weak Stability). We call a matching M weakly stable if for any blocking pair (a, p) for matching M, the following two conditions are satisfied.

- $a' \succ_p a$ for all applicants $a' \in M(p)$
- $M \cup \{(a, p)\}$ is not feasible.

Note the similarity in the definition of strong stability and weak stability. The only difference is that in the second condition, applicant a can have two contracts: one with project $M(a)$ and another with the project p she is blocking with. One way to see this is that in order for applicant a to block with p, it must sign the contract with p before it opts to annul its match with project $M(a)$. We call any blocking pair that satisfies these condition **permitted under weak stability**. Note that due to the second condition, any empty matching need not be weakly stable.

Note that strong stability implies weak stability, and in a setting without distributional constraints both definitions are equivalent

[2]Most of the computational hardness results concern overlapping regions (see e.g. Goto et al. [9]).

to classical stability as formulated by Gale and Shapley [8]. We present a polynomial-time algorithm that always returns a weakly stable matching. Kamada and Kojima [15] mentioned an algorithm (Appendix B.3 [15]) that produces in finite time a weakly stable matching under any distributional constraints that can be represented by a **feasibility function** $f : \mathcal{M} \rightarrow \{0, 1\}$ that satisfies the following condition:

- $f(\emptyset) = 1$ and, for any two matchings M and M', if $|M'(p)| \leq |M(p)|\ \forall p \in P$ then $f(M) = 1 \Rightarrow f(M') = 1$.

A matching M is feasible if and only if $f(M) = 1$. Note that the feasibility constraints are 'anonymous' in the sense that they do not depend on the names of the applicants matched but only on the number.

Our first insight is that our concept of supervisor-feasibility satisfies this condition. Therefore, we note that the algorithm of Kamada and Kojima can be applied to our setting if we treat the budget constraints as abstract feasibility constraints and if we ignore how exactly the research funding is allocated to the projects. In particular, the algorithm returns a weakly stable matching in finite time as noted by Kamada and Kojima.

We show that for our problem, a weakly stable matching can be computed in strongly polynomial-time. To prove the result, we first revisit the algorithm of Kamada and Kojima and present it in pseudocode. We carefully study the running time and understand the conditions under which the algorithm is polynomial-time computable. We prove that the algorithm returns a weakly stable matching. We then show how a weakly stable matching can be computed in polynomial time for our problem.

The idea of the algorithm is as follows. The algorithm maintains an ordering of the projects. The matching is initialized to the empty matching. With respect to the current matching, the algorithm maintains a set B_p for each project which contains the set of applicants that can block with p with respect to the current matching. If no $a^* \in B_p$ can simply be added to p's match without violating a feasibility constraint, we then return the current matching. Otherwise, we go through the list of projects and find the first project that has a non-empty B_p and for which some $a^* \in B_p$ can simply be added to p's match without violating a feasibility constraint. The algorithm then identifies $a^* \in B_p$ with such a property and which the highest priority for p among all such applicants. Applicant a^* is allowed to block with p to get a new matching. Such an applicant a^* is simply rematched to p to an updated matching. After getting the new matching, we repeat the process by again going through the list of projects starting from the first one. If there is a known priority ranking of the central administrator over the projects that ranking can be used as the order employed by the algorithm.

We first establish the following properties of the algorithm.

Lemma 5.2. *At every step of Algorithm 1, for the current matching M, there exist no pairs $(a, p) \in A \times P$ such that $p \succ_a M(a)$ and there exists $a' \in M(p)$ such that $a \succ_p a'$.*

Proof. We say that the algorithm begins with matching $M_0 = \emptyset$, takes a step every time M is updated and produces matching M_k after the kth step.

Suppose that M_k, $k > 0$, is the first matching in which we encounter a blocking pair (a, p) such that there exists $a' \in M_k(p)$

```
Input: lists $\succ_p$ for all $p \in P$ and $\succ_a$ for all $a \in A$; feasibility
    function $f$; project order $P^* = (p_1, ..., p_k)$
Output: Matching $M$
1  Initialize $M$ to empty.
2  $C_p \longleftarrow \{a \in B_p \mid f(M \cup (a, p)) = 1\}$ for each $p \in P$ % $B_p$ is the
   set of applicants who form blocking pairs with $p$ in matching
   $M$
3  if $C_p = \emptyset$ for all $p \in P$ then
4      return $M$
5  else
6      while $C_p \neq \emptyset$ for some $p \in P$ do
7          Locate the first $p_j$ in the list $P^*$ for which $C_{p_j} \neq \emptyset$.
8          Consider $a^*$ to be $p_j$'s most preferred applicant from
           $C_{p_j}$
9          Update $M$ as follows: $M \longleftarrow (M \setminus \{a^*, M(a^*)\}) \cup$
           $(a^*, p_j)$
10         Update $C_p$ for each $p$.
```

Algorithm 1

such that $a \succ_p a'$. Call this an envious blocking pair for matching M. We now note that Algorithm 1 is Pareto improving for the set of applicants, since applicants are never displaced, and only move up in their project rankings. Thus $p \succ_a M_k(a) \Rightarrow p \succ_a M_{k-1}(a)$. We consider two cases:

- Suppose $(a', p) \in M_{k-1}$. Then (a, p) is an envious blocking pair for M_{k-1}, violating our assumption that M_k was the first matching in which we encountered such a pair.
- Suppose $(a', p) \notin M_{k-1}$. Thus, (a', p) is the blocking pair satisfied in step k by Algorithm 1. Since our feasibility function is anonymous, (a, p) could have been satisfied in step k. And, since $a \succ_p a'$, a' was not p_j's most preferred applicant from C_{p_j} at that point, which results in a contradiction.

Therefore, it must be the case that there are no envious blocking pairs created during Algorithm 1 □

Using the lemma above, we prove the following.

Theorem 5.3. *A weakly stable matching always exists for a matching problem under any set of distributional constraints that can be represented by a feasibility function. Algorithm 1 produces one such matching.*

Proof. From Lemma 5.2 we know that Algorithm 1 does not introduce any envious blocking pairs as it is running. Thus, it correctly resolves a blocking pair that is not permitted under weak stability during every step, and terminates when there are no such pairs remaining, so the output M is weakly stable. Further, since the set of matchings is finite, and Algorithm 1 is applicant-improving, it must terminate in finite time. □

The next theorem shows that as long as the feasibility of a matching can be tested in polynomial time, Algorithm 1 runs in polynomial time.

Theorem 5.4. *Suppose checking $f(w)$ takes t time. Then, the running time of Algorithm 1 is $O(|A|^2|P|^2t)$.*

Proof. C_p can be computed in time $O(|A|t)$ so it takes time $O(|A||P|t)$ to compute all C_ps. The while loop iterates at most $|A|\cdot|P|$ times because each new match is a Pareto improvement for the applicants. In the while loop, we need to update C_ps which takes time $O(|A||P|t)$. Hence the overall time is $O(|A|^2|P|^2t)$. □

We note that Algorithm 1 can be applied to the summer internship problem where the feasibility function is based on the supervisor budgets. Therefore for the summer internship problem a weakly stable matching exists. Next we show that for our problem, a weakly stable matching can be computed in polynomial time. In view of Theorem 5.4, it is sufficient to show that for our problem, it can be checked in polynomial time whether a matching is feasible or not.

Checking Feasibility of a Matching. We can formulate the concept of feasibility in terms of network flows.[3] Define the **funding flow graph** G_M associated with a matching M as follows:

- $V(G_M) = \{s^*\} \cup S \cup P \cup \{t^*\}$, where s^* is the source, and t^* is the sink
- Arcs (s, p), for all supervisor-project pairs where $p \in P_s$ with capacity ∞
- Arcs (s^*, s), for all $s \in S$, each with capacity q_s
- Arcs (p, t^*), for all $p \in P$, each with capacity $|M(p)|$

Theorem 5.5. *The feasibility of a matching can be checked in polynomial time $O((\max\{|S|, |P|\}^3)$ for the summer-internship problem.*

Proof. Our first claim is that a matching M is feasible if and only if G_M admits a feasible s^*-t^* flow of size $|M|$.

Suppose M is feasible. Define a flow f on G_M as follows:

- $f(s,p) = x_{s,p}, \forall s \in S, p \in P_s$
- $f(s^*, s) = \sum_{p \in P_s} x_{s,p}, \forall s \in S$
- $f(p, t^*) = \sum_{s \in S_p} x_{s,p}, \forall p \in P$

It is easy to see that this is a feasible flow of size $|M|$. Now suppose that G_M admits a feasible flow f of size $|M|$. Set $x_{s,p} = f(s,p)$, $\forall s \in S, p \in P_s$. We can then show that this $\{x_{s,p}\}$ satisfies the conditions of feasibility.

Now that we have established the claim, we use the fact that the maximum flow problem can be solved in $O(V^3)$ time, using for instance, the algorithm proposed by Malhotra et al. [19]. $V(G_M) = |S| + |P| + 2$ and, given a matching M, G_M can be constructed in $O((|S| + |P|)^2)$ time. We can check whether M is feasible in $O(\max\{|S|, |P|\}^3)$ time by computing a maximum flow and verifying whether it equals $|M|$. □

Note that by the integer property of the network flow problem, if all the capacities of the supervisors are integer and the flow is feasible then an integer funding allocation exists.

Although Algorithm 1 satisfies weak stability, it also has some drawbacks. Next we establish some properties of the algorithm.

Theorem 5.6. *The following properties hold for Algorithm 1.*

1. *Algorithm 1 is not strategyproof for the applicants.*
2. *Algorithm 1 does not always find a strongly stable matching whenever one exists.*
3. *Changing the order of projects ordered after p by P^* can change p's allocation.*
4. *There exist weakly stable matchings that cannot be produced as a result of Algorithm 1 by changing the project order P^*.*

The proofs of the statements are based on examples.

Proof. We prove each of the statements separately.

(1) Consider the following instance.

$$A = \{a_1\} \qquad \succ_{a_1}: p_2, p_1$$
$$P = \{p_1, p_2\} \qquad \succ_{p_1}: a_1 \qquad \succ_{p_2}: a_1$$
$$S = \{s\} \qquad P_s = \{p_1, p_2\} \qquad q_s = c_{p_1} = c_{p_2} = 1$$

If we set $P^* = (p_1, p_2)$ then the algorithm outputs $M = \{(a_1, p_1)\}$. However, if a_1 were to lie about their preferences, setting $\succ_{a_1}: p_2$, then the algorithm will output $M = \{(a_1, p_2)\}$, which is preferred by a_1. Thus, the algorithm is not strategy-proof for applicants.

(2) For the example above, since (a_1, p_2) is the only strongly stable matching, the algorithm does not find the strongly stable matching when one exists.

(3) Consider the following instance.

$$A = \{a_1, a_2, a_3\}$$
$$\succ_{a_1}: p_2, p_1 \qquad \succ_{a_2}: p_1 \qquad \succ_{a_3}: p_3$$
$$P = \{p_1, p_2, p_3\}$$
$$\succ_{p_1}: a_1, a_2 \qquad \succ_{p_2}: a_1 \qquad \succ_{p_3}: a_3$$
$$S = \{s\} \qquad P_s = P$$
$$c_{p_1} = c_{p_2} = c_{p_3} = 1 \qquad q_s = 2$$

Setting $P^* = (p_1, p_2, p_3)$, the algorithm proceeds as follows: a_1 gets matched to p_1, then a_1 gets matched to p_2, and then a_2 gets matched to p_1. The algorithm terminates with matching $(a_1, p_2), (a_2, p_1)$ However, if we set $P^* = \{p_1, p_3, p_2\}$, a_1 gets matched to p_1 and then a_3 gets matched to p_3. The algorithm terminates with matching $(a_1, p_1), (a_3, p_3)$. This shows that: changing the order of projects ordered after p by P^* can change p's allocation.

(4) Consider the following instance.

$$A = \{a_1, a_2\} \qquad \succ_{a_1}: p_1, p_2 \qquad \succ_{a_2}: p_1, p_2$$
$$P = \{p_1, p_2\} \qquad \succ_{p_1}: a_1, a_2 \qquad \succ_{p_2}: a_1, a_2$$
$$S = \{s\} \qquad P_s = P$$
$$c_{p_1} = c_{p_2} = q_s = 2$$

$P^* = (p_1, p_2)$ produces $M = \{(a_1, p_1), (a_2, p_1)\}$
$P^* = (p_2, p_1)$ produces $M = \{(a_1, p_2), (a_2, p_2)\}$
There is no project order P^* that produces $M = \{(a_1, p_1), (a_2, p_2)\}$. Thus, there exist weakly stable matchings that cannot be produced as a result of the algorithm.

This complete the proof. □

It is natural to consider applicants with varying costs in our summer internship setting. For instance, rural applicants can require an accommodation allowance, and international students may receive less governmental support. However, this would lead to the existence of some blocking pairs that are not permitted under weak stability, but can no longer be satisfied without violating feasibility.

[3]For an overview of network flows, see [2].

To address this, we can modify our definition of weak stability to the following:

Definition 5.7 (Weak Stability with varying applicant costs). We call a matching M weakly stable if for any blocking pair (a, p) for matching M, the following conditions are satisfied.

- For each applicant $a' \in M(p)$, either:
  - $a' >_p a$, or
  - $M \cup \{(a, p)\}/\{afi, p\}$ is not feasible
- $M \cup \{(a, p)\}$ is not feasible.

This new definition simply prevents less costly applicants from being replaced with more costly ones if that would violate the feasibility constraint.

Example 5.8. Denote the wage, in units, of applicant a by w_a. The following instance shows a situation with varying applicant costs that does not admit a weakly stable matching.

$$A = \{a_1, a_2, a_3\}$$

$>_{a_1}: p_2, p_1$	$>_{a_2}: p_1, p_2$	$>_{a_3}: p_1$
$w_{a_1} = 1$	$w_{a_2} = 1$	$w_{a_3} = 2$
$P = \{p_1, p_2\}$		
$>_{p_1}: a_1, a_3, a_2$	$>_{p_2}: a_2, a_1$	
$S = \{s_1, s_2\}$	$P_{s_1} = p_1$	$P_{s_2} = p_2$
$c_{p_1} = c_{p_2} = 2$	$q_{s_1} = 2$	$q_{s_2} = 1$

The following matchings are not weakly stable:

- $\{(a_1, p_1), (a_2, p_1)\}$ - blocked by (a_1, p_2)
- $\{(a_1, p_2), (a_2, p_1)\}$ - blocked by (a_3, p_1)
- $\{(a_1, p_2), (a_3, p_1)\}$ - blocked by (a_2, p_2)
- $\{(a_2, p_2), (a_3, p_1)\}$ - blocked by (a_1, p_1)
- $\{(a_1, p_1), (a_2, p_2)\}$ - blocked by (a_2, p_1)

All other feasible matchings are of size one or zero, and are therefore not weakly stable. Thus, no weakly stable matching exists.

This result also applies for the setting of Kamada and Kojima [13] with disjoint regions, and no supervisors, if, instead of having varying costs, applicants take up varying amounts of slots in a project.

6 FAIR BUDGET ALLOCATIONS

Given a feasible matching M, there can exist multiple ways to allocate supervisor budgets among projects to fund all applicants matched to them. Our goal is to find a method for the fairest such allocation. We have chosen to deal with fairness post-match, in order to find a solution that does not constrain the set of feasible matchings.

For each supervisor $s \in S$, and each project they supervise $p \in P_s$ set $0 < t_{s,p} \leq 1$ such that, for each $p \in P$, $\sum_{s \in S_p} t_{s,p} = 1$. Call this the normative 'target' - how much we would want s to contribute to the funding of any applicant matched to p. For instance, if we would ideally want the supervisors of any given project to contribute equally to its funding, we would set: $t_{s,p} = \frac{|M(p)|}{|S_p|}$ $\forall p \in P, s \in S_p$. However, given a feasible matching M, some supervisors may lack sufficient funding to reach these targets. Therefore, we seek to find a funding allocation that is closest to the target allocations. In order to define closest, we consider specific lexicographic comparisons. Let $X = \{x_{s,p}\}_{s \in S, p \in P_s}$ be a feasible funding allocation for matching M. Denote by ϕ_X the vector corresponding to the weakly decreasing ordering of the set: $\left\{\frac{x_{s,p}}{t_{s,p}}\right\}_{s \in S, p \in P_s}$. Denote by Φ_M the set of such vectors corresponding to all feasible funding allocations for matching M.

The **fairest feasible funding allocation** for matching M is the funding allocation corresponding to $\phi^* \in \Phi_M$ such that for all $\phi \in \Phi_M$, $\phi^* \prec_{lex} \phi$, where $\prec_{lex}$ refers to the well-known lexicographic order. This is exactly equivalent to finding the leximin

```
Input: $T = \{(s,p) | s \in S, p \in P_s\}$, $q_s \ \forall s \in S$ and $\{t_{s,p} | (s,p) \in T\}$.
Output: Funding Allocation $x$
1  $T_{tight} \leftarrow \emptyset$
2  while $T_{tight} \neq T$ do
3      Minimise the maximal component of $\left\{\frac{x_{s,p}}{t_{s,p}} | (s,p) \in T\right\}$
       that is not yet tight by solving the following LP:
       $\lambda^* \leftarrow$ Min $\lambda$ s.t.
           $x_{s,p} \geq 0 \quad \forall (s,p) \in T$
           $\sum_{s \in S_p} x_{s,p} = |T(p)| \quad \forall p \in P$
           $\sum_{p \in P_s} x_{s,p} \leq q_s \quad \forall s \in S$
           $\frac{x_{s,p}}{t_{s.p}} \leq \lambda \quad \forall (s,p) \in T \backslash T_{tight}$
           $\frac{x_{s,p}}{t_{s.p}} = \lambda_{s,p} \quad \forall (s,p) \in T_{tight}$
4      for $(s,p)^* \in T \backslash T_{tight}$ do
5          Determine whether the constraint corresponding to
           $(s,p)^*$ is tight by solving the following Auxiliary LP:
           $\epsilon^* \leftarrow$ Max $\epsilon$ s.t.
               $x_{s,p} \geq 0 \quad \forall (s,p) \in T$
               $\sum_{s \in S_p} x_{s,p} = |T(p)| \quad \forall p \in P$
               $\sum_{p \in P_s} x_{s,p} \leq q_s \quad \forall s \in S$
               $\frac{x_{s,p}}{t_{s.p}} \leq \lambda^* \quad \forall (s,p) \in T \backslash T_{tight} \backslash \{(s,p)^*\}$
               $\frac{x_{s,p}}{t_{s.p}} = \lambda_{s,p} \quad \forall (s,p) \in T_{tight}$
               $\frac{x_{s,p}}{t_{s.p}} + \epsilon \leq \lambda^* \quad$ for $(s,p) = (s,p)^*$
6          if $\epsilon^* = 0$ then
7              Add $(s,p)^*$ to $T_{tight}$
8              $\lambda_{s,p} \leftarrow \lambda^*$
9  return $\{\lambda_{s,p} | (s,p) \in T\}$
```

Algorithm 2: Computing the fairest funding allocation for a given feasible matching

optimum of the following set: $\left\{\frac{-x_{s,p}}{t_{s,p}}\right\}_{s\in S, p\in P_s}$. The fairest feasible funding allocation can be achieved via Algorithm 2, which runs a series of linear programs.

Theorem 6.1. *Given a feasible matching, the fairest funding allocation can be computed in time $O(|S|^2|P|^2)LP(O(|S|\cdot|P|))$, where LP refers to the running time of the linear programming algorithm used.*

Proof. The algorithm solves a series of linear programs with $O(|S|\cdot|P|)$ constraints. The while loop iterates at most $|T|\le|S||P|$ times, as the algorithm adds at least one element to the set T_{tight} in every iteration. The for loop also iterates at most $|T|$ times. Thus, the overall complexity is $O(|S|^2|P|^2)LP(O(|S|\cdot|P|))$, where LP refers to the running time of the linear programming algorithm used. Since linear programs can be solved in polynomial time, the fairest funding allocation can also be computed in polynomial time. □

7 CONCLUSION

We presented a novel matching model that captures many real-world scenarios. For the model, we presented a compelling solution that is polynomial-time and satisfies stability and fairness properties. Several directions and problems arise as a result of our study. Our approach to finding a fair budget allocation was to first compute a weakly stable matching and then find the fairest possible budget allocation. It will be interesting to explore a fair outcome that is fairest in some global sense across all weakly stable matchings. We showed that the algorithm we consider is not strategyproof for applicants. It is open whether there exists an algorithm that is strategyproof and satisfies weak stability. The problem has been open even for the abstract setting of Kamada and Kojima [15].

ACKNOWLEDGMENTS

Aziz gratefully acknowledges the UNSW Scientia Fellowship and Defence Science and Technology (DST). Baychkov's work was supported by the CSIRO undergraduate vacation scholarship program, and he extends many thanks to Haris Aziz and Gavin Walker for their invaluable mentorship. Biró is supported by the Hungarian Academy of Sciences under its Momentum Programme (LP2016-3/2019) and Cooperation of Excellences Grant (KEP-6/2019), and the Hungarian Scientific Research Fund, OTKA, Grant No. K128611.

REFERENCES

[1] D. J. Abraham, R. W. Irving, and D. Manlove. 2007. Two algorithms for the Student-Project Allocation problem. *J. Discrete Algorithms* 5, 1 (2007), 73–90.

[2] R. K. Ahuja, T. L. Magnanti, and J. B.Orlin. 1993. *Network Flows: Theory, Algorithms, and Applications.* Prentice Hall.

[3] H. Aziz, J. Chen, S. Gaspers, and Z. Sun. 2018. Stability and Pareto Optimality in Refugee Allocation Matchings. In *Proceedings of the 17th International Conference on Autonomous Agents and Multiagent Systems (AAMAS).* 964–972.

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

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

[6] D. Fragiadakis, A. Iwasaki, P. Troyan, S. Ueda, and M. Yokoo. 2016. Strategyproof matching with minimum quotas. *ACM Transactions on Economics and Computation* 4, 1 (2016), 1–40.

[7] D. Fragiadakis and P. Troyan. 2017. Improving matching under hard distributional constraints. *Theoretical Economics* 12, 2 (2017), 863–908.

[8] D. Gale and L. S. Shapley. 1962. College admissions and the stability of marriage. *The American Mathematical Monthly* 69, 1 (1962), 9–15.

[9] M. Goto, A. Iwasaki, Y. Kawasaki, R. Kurata, Y. Yasuda, and M. Yokoo. 2016. Strategyproof matching with regional minimum and maximum quotas. *Artificial intelligence* 235 (2016), 40–57.

[10] M. Goto, F. Kojima, R. Kurata, A. Tamura, and M. Yokoo. 2017. Designing matching mechanisms under general distributional constraints. *American Economic Journal: Microeconomics* 9, 2 (2017), 226–262.

[11] A. Ismaili, N. Hamada, Y. Zhang, T. Suzuki, and M. Yokoo. 2019. Weighted Matching Markets with Budget Constraints. *J. Artif. Intell. Res.* 65 (2019), 393–421.

[12] A. Ismaili, T. Yamaguchi, and M. Yakoo. 2018. Student-Project-Resource Allocation: Complexity of the Symmetric Case. In *PRIMA 2018: Principles and Practice of Multi-Agent Systems.* Springer Verlag, 226–241.

[13] Y. Kamada and F. Kojima. 2015. Efficient Matching under Distributional Constraints: Theory and Applications. *American Economic Review* 105, 1 (2015), 67–99.

[14] Y. Kamada and F. Kojima. 2017. Recent Developments in Matching with Constraints. *The American Economic Review* 107, 5 (2017), 200–204.

[15] Y. Kamada and F. Kojima. 2017. Stability concepts in matching under distributional constraints. *Journal of Economic Theory* 168, C (2017), 107–142.

[16] Y. Kawase and A. Iwasaki. 2017. Near-Feasible Stable Matchings with Budget Constraints. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI-17.* 242–248. https://doi.org/10.24963/ijcai.2017/35

[17] Y. Kawase and A. Iwasaki. 2018. Approximately Stable Matchings With Budget Constraints. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18).* 1113–1120.

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

[19] V. Malhotra, M. P. Kumar, and S. N. Maheshwari. 1962. An $O(|V|^3)$ algorithm for finding maximum flows in networks. *Inform. Process. Lett.* 7, 6 (1962), 277–278.

[20] D. F. Manlove. 2013. *Algorithmics of Matching Under Preferences.* World Scientific Publishing Company.

[21] D. F. Manlove, R. W. Irving, K. Iwama, S. Miyazaki, and Y. Morita. 2002. Hard variants of stable marriage. *Theoretical Computer Science* 276, 1–2 (2002), 261–279.

[22] A. E. Roth. 2008. Deferred acceptance algorithms: history, theory, practice, and open questions. *International Journal of Game Theory* 36 (2008), 537—569.