Need-Aware College Admissions and the Stability of Marriage

Max Bender, Kirk Pruhs[1], and Alireza Samadian

Abstract

We consider generalizing the classic stable marriage problem to a setting where each university has a particular value for each student, as well as costs that depend upon the number of enrolled students, and the stability condition incorporates the revenue incentive of the universities. We show that the general problem is NP-hard. In contrast, we show that stable marriages can be found if the costs are convex. We also consider concave, and almost convex cost functions.

1 Introduction

In 1962, during that brief shining moment that was known as Camelot, Gale and Shapley published their seminal paper "University Admissions and the Stability of Marriage" in which they formulated the stable marriage problem, and gave their eponymous algorithm to find stable marriages[9]. In those halcyon days, public universities were well funded, and admissions criteria were universally need blind. Now the days of Camelot are but legend. Public support of public universities has cratered. For example, at the authors' public university, the University of Pittsburgh, the portion of the budget coming from state/public money has decreased from 35% two generations ago to approximately 7% today, and the university has started to make plans for going private in expectation for further future decreases in public support[5, 10, 17]. In response, public universities that historically prided themselves on being incubators of scholars and knowledge, from which ideas flow freely into the world, now commonly seek to monetize everything from sports slogans to scientific research[13]. Consistent with this trend, many universities are implementing more need aware admissions policies [18, 4]. Thus we reformulate the Gale and Shapley formulation of the university admissions problem to reflect the current profit motive of such universities, and study the stability properties of our reformulation.

In our Need-Aware Stable Assignment Problem (NASA), as in the Gale and Shapley formulation, we assume that there is a collection $S = \{s_1, \dots s_m\}$ of students that seek admission to a collection $U = \{U_1, \dots, U_n\}$ of universities, and that each student s_j has a strict preference order among the universities $U^j_1, ..., U^j_i, ..., U^j_n$. However, the formulation from the university side is more profit oriented. We assume that there is a value $v_i(s_j)$ of each student s_j to each university U_i. This value may reflect (1) the money that the university believes they can extract from the student and his/her family in tuition, gifts, bequeaths, etc., (2) whether the student is an athlete in a revenue generating sport, and (3) perhaps marginally the academic merit of the student. We assume that the universities preferences are also strict, that is no university values two students exactly equally. To take university costs into account, we assume that there is a cost function F_i for each university U_i, where $F_i(k)$ is the cost that the university incurs if it enrolls k students (note that we assume that costs depend only on the number of enrolled students, and not on the identity of these students). The revenue that a university receives from admitting a collection of students is the aggregate value of those students to that university minus the cost of that

[1]Supported in part by NSF grants CCF-1421508 and CCF-1535755, and an IBM Faculty Award.

many students. An assignment $A : S \to \{U_1, ..., U_n\}$ assigns each student to at most one university (note that not all students have to be assigned). As is usually the case in many-to-one or many-to-many matching problems [3], there is not a unique natural definition of a stable assignment. However, after some exploration of the alternatives, we came to the conclusion that the following definition of stability seems to be the leading candidate for the title of most natural definition of stability:

- An assignment A to be stable if there does not exist a university U_i and a set of students T that forms a blocking coalition, which occurs when all students in T are either assigned to U_i or prefer U_i over their current assignment, and U_i can increase its revenue by enrolling exactly the students in T (potentially disenrolling some currently enrolled students).

The original Gale and Shapley formulation was the special case of our formulation where each university cost function F_i was zero when the enrollment is not more than one students, and infinite for enrollments of two or more students. Gale and Shapely showed that if the number of students is equal to the number of universities, then one can efficiently compute a university optimal stable marriage (in which each university is at least as happy with its student as it would be in any other stable marriage), and one can efficiently compute a student optimal stable marriage (in which each student is at least as happy with its university as it would be in any other stable marriage).

1.1 Our Results

We first observe in Section 2 that if the university cost functions can be arbitrary, then the problem of determining whether a stable marriage exists is NP-complete. In particular, it is NP-hard even if the cost functions are convex everywhere but at zero. Thus we consider three special classes of cost functions: (1) convex, (2) concave, and (3) convex everywhere but at zero. For each of these settings, we want to determine if a stable assignment always exists, and if so, we want to find an algorithm to efficiently compute a stable assignment. And if a stable marriage need not always exist, we want to find an algorithm to efficiently determine whether stable assignment exists for a particular instance. We also consider the situation that the number of universities is small, as this seems like a natural type of instance. For example, such a situation arises when a university system has multiple campuses (which is common for large public universities) and seeks a stable assignment of students wishing to enroll in that university system.

In Section 3 we give efficient algorithms to compute a maximum revenue subcollection of students for a particular university and determine whether an assignment is stable.

In Section 4 we consider cost functions that are (everywhere) convex. Convex cost functions naturally arise in situations where there are scarce resources that must be procured, and as these scarce resources are depleted, the cost of these resources follows the law of supply and demand, and increases. A common example, that arises in university settings, occurs when the number of admitted students exceeds the capacity of the university's dormitories. As a result the university needs to rent additional housing on the free market, with the natural result being that the university will pay a higher cost per housing unit as it consumes the cheaper available rental options. We first give an efficient algorithm to compute a university optimal stable assignment. An assignment A is university optimal if the revenue that each university U_i achieves from A is at least as much as U_i would achieve in any stable assignment. We then give an efficient algorithm to compute a student optimal stable assignment. An assignment A is student optimal if for each student s_j there is not another stable assignment in which s_j is enrolled in a university which s_j prefers over the university s_j is assigned to in A. The results don't break any new ground in terms of algorithmic design and analysis, but we find the results appealing (if nothing, they would make a nice homework problem for students after they are taught the Gale-Shapely algorithm).

In Section 5 we consider cost functions that are concave. Concave cost functions naturally arise in settings where there is some economy of scale that can be exploited with more students. A common strategy for public universities to alleviate budget deficits is to admit more students, while keeping the number of classes and the number of faculty fixed; as a result, universities get more income, with minimal additional expenses. (Of course this increases class sizes, probably lowering the quality of education, and further burdens faculty. But these are secondary concerns.) We first show that there are instances with three universities where there is no stable assignment. We then give an efficient algorithm that always finds a stable assignment in instances with at most two universities.

Finally in Section 6 we consider cost functions that are convex everywhere but at zero. We first show that there are instances with only two universities for which there is no stable assignment. We then give an efficient algorithm that determines whether an instance with two universities admits a stable assignment.

1.2 Other Related Results

As far as we are aware, all previous papers that consider stable marriage in two sided matching assume each side specified a preference order (sometimes with ties and/or incomplete). So as far as we are aware, considering stability based on revenue is novel. But to quote from [3]: "Since 1962, the study of matching problems involving preferences has grown into a large and active research area, and numerous contributions have been made by computer scientists, economists, and mathematicians, among others. Several monographs exclusively dealing with this class of problems have been published." So it would not at all be surprising if there was some paper in the literature, that we are not aware of, that considers such a revenue based stability condition. Similarly it is infeasible to fully cover even a decent fraction of the related results in this area, and we confine ourselves to the results that we are aware of that seem most closely related, which primarily follows from many-to-many matchings.

The results in [1] consider the case where each man and woman wishes to be matched to as many acceptable partners as possible, up to his or her specified quota, and gave a polynomial time algorithm for finding a stable matching that minimizes the sum of partner ranks across all men and women. [2] showed that when universities have both upper and lower quotas on their enrollment, it is NP-hard to determine if a stable marriage exists. As in our formulation of NASA, the assumption in [2] assumes a university would prefer to close than have an enrollment below its lower quota. Several other NP-hard variants can be found in [14].

Literature also often studies many to many matchings in the context of firms and markets. In the setting where every firm has a ranking over the workers and the workers have a strict ranking over a set of acceptable firms, then [7] and [8] give algorithms under different optimality objectives for finding stable assignments. In the setting where workers have a strict ranking function over an acceptable set of subsets of the firms (and vice-versa for the firms over the workers) [16] showed that pairwise stable solutions always exist if the ranking functions satisfy substitutability and [15] gives an algorithm for finding all such pairwise stable solutions. Setwise stable solutions, which more closely resemble the definition for stability within this work, for the second setting have also been considered in [19], [6], [12], and [11].

2 NP-hardness

In this section we prove that the problem of determining whether a NASA instance admits a stable assignment is NP-Complete.

Theorem 2.1. *If the universities' cost functions can be arbitrary, then determining if there exists a stable matching, is NP-Complete.*

Proof. First, we show that this problem is in NP. Let M be a matching, we can verify if M is stable by the following steps. For each i, let Q_i be the set of students that are either matched to U_i or prefer U_i to their current position under M. We can sort this set in decreasing order of v_i, and if there is a k such that U_i makes more profit by having the k most valuable students of Q_i, then M is not stable. This can be done in polynomial time with respect to the number of students and universities.

Furthermore, we need to reduce an NP-hard problem to our problem. The College Admission problem with lower quotas (CA-LQ) is an NP-complete variation of College Admission problem [2]. We reduce CA-LQ to the problem of deciding if an instance of the Need-Aware Stable Assignment problem (NASA) with arbitrary cost function has stable matching.

Based on [2], CA-LQ is a College Admission problem in which in addition to the upper quota, each college also has a lower quota and if the number of students matched to a college is less than the lower quota, then the college automatically closes and disenroll all of its students. So the input for an instance of CA-LQ is a set of colleges $C = \{U_1, U_2, U_3, \ldots, U_n\}$ and a list of students $A = \{s_1, s_2, \ldots, s_m\}$. Each of the students has a strict order of preferences over the colleges that he applies to and each college has a strict order of preferences over its acceptable students. Furthermore, each college also has an upper bound $up(U_i)$ on the number of students that it can accept. In addition to the upper bound, there is a lower bound quota $low(U_i)$ that shows the minimum number of students needed to keep a college open. If the number of students in a college is less than its lower bound, it will automatically close. An assignment is stable if and only if the following two conditions hold:

1. There is no student who prefers an open university to his current assignment and that university also prefers to enroll him either by adding him or replacing another less preferred student.
2. There is no closed university U_i which can get open by enrolling a set of students that prefer U_i over their current assignment.

Let I be an instance of CA-LQ, in which $C = \{U_1, U_2, \ldots, U_n\}$ is the set of colleges, $A = \{s_1, s_2, \ldots, s_m\}$ is the set of students, $low(U_i)$ and $up(U_i)$ are lower quota and upper quota of U_i. We construct I', an instance of NASA as follows. The set of universities and students are the same so we have $U = C$ and $S = A$. The preference of each student over the universities remains unchanged. We set the valuation functions of U_i for a student that is not acceptable by U_i in I to be zero and be $v_i(s_j) = 1 + K_i^j \epsilon$ otherwise, in which s_j is the $(K_i^j)^{th}$ least valuable student in U_i's preference list in I, and ϵ is an arbitrary positive constant less than $\frac{1}{n^2}$. By using such a valuation function, if U_i prefers student s over s' in I, then there is the same preference in I'. Furthermore, we set the cost function of U_i to

$$F_i(k) = \begin{cases} 0 & k = 0 \\ low(U_i) + k\epsilon & 0 < k \le up(U_i) \\ 100k & up(U_i) < k \end{cases}$$

We claim that I has a stable matching if and only if I' has a stable matching.

A matching in I' is stable if and only if the following conditions hold:

1. There is no student matched to U_i that is unacceptable for U_i in I. Because in that case the value of that student is zero and since the cost functions are strictly increasing, U_i can make positive profit by disenrolling that student.

2. The number of matched students to each university is between its lower quota and upper quota in I. Otherwise the university will make negative revenue and prefers to disenroll everyone and close.

3. There is no subset of students who prefer U_i over their current assignment and U_i prefers to enroll them or replace some of its students with them.

Based on the definition of stability in I the above conditions are equivalent to the definition of stability in I'. So we can conclude the assignment M is stable for I if and only if M is stable for I'.

□

3 General Cost Functions

We first give an efficient algorithm A_0 for selecting a subset of students to admit, from a collection of students seeking admission to a particular university, that maximizes the revenue for that university (the algorithm can be proved correct using a simple exchange argument). We then give an efficient algorithm A_1 to determine whether a particular assignment A is stable (the algorithm's correctness follows more or less from the definition of stability). Finally we give an alternative characterization of stability that will be useful in some of our proofs.

Algorithm A_0: The algorithm takes as input a particular university U_i and a collection S of students. Let S_k be the collection of the k students in S that U_i most values. The maximum revenue subset of S is then the set S_k with maximum revenue, namely the S_k where $k = \arg\max_k \sum_{s_j \in S_k} v_i(s_j) - F_i(k)$.

Algorithm A_1: The algorithm takes as input an assignment A. For each university U_i, let C_i be the students either assigned to U_i in A, or that prefer U_i to their assignment in A (which can be because the student was assigned to no university in A or prefers U_i to the university the student was assigned in A). The assignment is then stable if there is no university U_i where U_i's maximum revenue subset of C_i is different than the students assigned to U_i in A. The maximum revenue subset can be computed using algorithm A_0.

An assignment A is stable if and only if all of the following three conditions hold:

1. For each university U_i, there is no subset S of students that prefer U_i over their current assignment in A, and such U_i can increase its revenue by additionally enrolling the students in S.

2. No university can increase its revenue by disenrolling some of its enrolled students.

3. There is no single student s_j that prefers a university U_i over the university it is assigned in A and U_i would increases its revenue if it replaces one of its current students with s_j.

This is true because if any of the above conditions do not hold, then there is a subset of students S' willing to join U_i that U_i can make more profit by taking S' rather than its current assigned students. Furthermore, if the students assigned to U_i are not the most valuable students willing to join U_i then U_i can replace some of them or disenroll or add some other good students to its current assigned students.

4 Convex Cost Functions

We consider instances where all university costs functions are convex. For such instances, we give an algorithm A_2 that produces a student optimal stable assignment, and then give an algorithm A_3 that produces a university optimal stable assignment.

Algorithm A_2: The algorithm consists of rounds. The algorithm maintains a set Q_i^k containing the students tentatively enrolled in university U_i after $k-1$ rounds. Initially each Q_i^1 is empty. In round k, each student s_j that is not tentatively enrolled in some university, proposes to their top ranked university among those universities to which they have not proposed to in a previous round. Let P_i^k be the students that propose to university U_i in round k. The set of students Q_i^{k+1} tentatively enrolled at the end of round k is then set to be the subset of students that either were previously enrolled or are seeking enrollment, the subset of $Q_i^k \cup P_i^k$, that maximizes U_i's revenue. This can be computed using algorithm A_0. The other students, namely the students in $Q_i^k \cup P_i^k \setminus Q_i^{k+1}$ are permanently rejected. The algorithm terminates when all students have either been assigned to some university or rejected by every university.

Lemma 4.1. *When the A_2 algorithm terminates, there exists no pair (s_j, U_i) such that s_j prefers U_i to its assignment and U_i could increase its revenue by enrolling s_j.*

Proof. Suppose for the sake of contradiction that there exists some student s_j who prefers university U_i to their current assignment (which may be no university) where U_i would be willing to enroll s_j by letting go of some collection T of students. Because the cost function is assumed convex, it follows that T is either empty or at most one student s'. Since $|T| \leq 1$ it follows that the number of students assigned to U_i is never decreasing. By construction of the algorithm, s_j must have applied to U_i at some point before termination of the algorithm. Student s_j was then either rejected at this moment or enrolled only to be replaced later by some student s. In either case, at that moment every student assigned to U_i had higher value than $v_i(s_j)$ and $v_i(s_j)$ was lower than the marginal cost of adding a new student. Since the total valuation of U_i over all of its students is monotonically increasing, it follows that $v_i(s_j)$ was still lower than the marginal cost of adding a new student at the termination of the algorithm. Thus T must be some student s' such that $v_i(s') < v_i(s_j)$. Thus s' could not have been in U_i's assignment at the time of s_j's rejection, since s' would have been rejected instead. Thus s' must have been added after s_j was rejected. But this is also a contradiction, as $v_i(s')$ must also have been lower than the marginal cost at the time it was added, so it must have replaced another student whom s_j could also have replaced instead. Thus s_j and U_i cannot exist. □

Theorem 4.2. *The A_2 algorithm gives a stable assignment which is student optimal.*

Proof. By Lemma 4.1 the solution returned by the SP algorithm must be a stable assignment, since every university would drop students if it could make revenue by doing so, and would have enrolled any students it could make revenue from who would prefer to change universities. Further, this solution is student optimal - that is, each student is assigned to their maximally ranked school over all possible stable assignments. To see this, suppose there is an instance where a student does not get their best possible university. Let s be the first student to get rejected by their best possible university U and let T be the set of students assigned to U at the time s is rejected. Since U is a possible university for s, then there is an assignment A' where s is assigned to U. Note that not every student in T can be assigned to U in A', because otherwise U could remove s to make more profit. Thus there exists some $s' \in T$ such that s' is not assigned to U in A' and is instead assigned to U'. Since s was the first student to be rejected by their best possible university, it follows that s' prefers U to U'. Since U preferred all of the students in T to s, it also follows that U prefers s' to s. Thus A' is not a stable assignment, since U would make more profit by swapping s' with s. □

Algorithm A_3: The algorithm consists of rounds. The algorithm maintains a set Q_i^k containing the students tentatively enrolled in university U_i after $k-1$ rounds. Initially each Q_i^1 is empty. In round k, let C_i^k be the collection of candidate students for university U_i, which are those students that have not rejected university U_i in a previous round. Let I_i^k be the students in C_i^k that would maximize university U_i's revenue, which can be computed by algorithm A_0. Each university U_i then sends invitations to each student in I_i^k. Each student then permanently rejects every university that is not the student's top choice from among the universities from which the student received an invitation. Then Q_i^{k+1} is the students in I_i^k that did not reject U_i's invitation. The algorithm terminates in the first round k for which it is the case that no university had an invitation rejected.

Lemma 4.3. *When A_3 terminates no university would prefer to drop students to increase revenue.*

Proof. For the sake of contradiction suppose there exists some university U_i and collection of students T such that the revenue U_i makes with its assignment at the termination of the algorithm is greater than the assignment where it drops all students T from its assignment. First, note that by construction of the assignment no students are ever being disenrolled and hence their options are only getting better and better. Because of this, if a student chooses not to go to U_i the moment they are accepted, they will never choose to go to U_i later. If such a group of students T exists, then each of these students has a lower value than their marginal utility. However, consider that at each moment the university sent invitations to these students, their value was higher than the marginal cost of adding them. Thus the only way their value could become lower than the marginal cost of adding them is if higher-valued students came to the university after them. However, this is a contradiction, since the university must have invited higher valued students to the university no later than the students in T, and as noted students who do not come at the time of invitation to a university will never join later. □

Theorem 4.4. *The algorithm A_3 gives a stable assignment which is university optimal.*

Proof. The stability of the assignment follows from lemma 4.3 and the observation that each student has been given an invitation to attend any university that could make profit from them - therefore there cannot be any pair (U_i, s_j) such that s_j prefers U_i to their current assignment and U_i can make more revenue by accepting s_j. The university-optimality of the solutions follows similarly to the student-optimality of A_2. That is, suppose that it is not university optimal. Then there is an instance where some university doesn't get its maximal possible profit from students by A_3. Let U be the first university that gets rejected by a student $s \in T \subset S$ where T would give U its maximal profit over all stable assignments and let A' be the assignment where U gets assigned exactly the students T. Since s rejects U in A_3, it follows that s is assigned to some university U' in A_3 that it ranks higher than U. Since s is assigned to U' and there have been no rejections before this moment, it follows that U' will always prefer to have s in its assignment. Thus in A' we have that s is assigned to U, s prefers U to U, and U would enroll s if given the opportunity. Thus A' is not stable. □

5 Concave Cost Functions

In this section we assume that the university cost functions are concave. We first show in Theorem 5.1 that there is an instance with three universities in which there is no stable assignment. We also provide an Algorithm A_4 which can always finds a stable assignment in instances with at most 2 universities.

Theorem 5.1. *For all $n > 2$ there exists an instance with n universities with concave cost functions that does not have a stable assignment.*

Proof. First we show there is an instance I with 3 universities that does not have any stable assignment. Assume the cost function for all the universities is the following

$$F(k) = \begin{cases} 0 & k = 0 \\ 100 & k = 1 \\ 150 & k = 2 \\ 200 & k > 2 \end{cases}$$

The valuation function of the universities are $v_1 = \{(s_1, 101), (s_2, 51), (s_3, 1)\}$, $v_2 = \{(s_2, 101), (s_3, 51), (s_1, 1)\}$, and $v_3 = \{(s_3, 101), (s_1, 51), (s_2, 1)\}$. Furthermore, let the S_1's ordered preference lists be (U_2, U_3, U_1), S_2's list be (U_3, U_1, U_2), and S_3's be (U_1, U_2, U_3). We can verify there is no stable assignment by checking the following categories of assignments: (1) An assignment with one unassigned student. (2) An assignment with each university having one student. (3) An assignment with two students assigned to one university and one student assigned to some other university. (4) Assignment in which all of them are assigned to one university.

In all of the above categories of assignment there is a university that is making negative revenue or there is a student s_j who is willing to join another university U_i and U_i makes more profit by enrolling s_j.

For any instance with $n > 3$ universities, we can just add $n - 3$ universities that are not willing to accept any student to instance I. □

Algorithm A_4: The algorithm consists of rounds. The algorithm maintains a set Q_i^k containing the students tentatively enrolled in university U_i after $k - 1$ rounds. Initially Q_1^1 consists of the unique stable assignment if university U_2 did not exist, which can be computed by algorithm A_0, and Q_2^1 is empty. In round k let the candidate students C_2^k for university U_2 be the collection of students that either prefer university U_2 to university U_1, or who are not tentatively enrolled at university U_1 (that is, they aren't in Q_1^k). Q_2^{k+1} is set to be the subset of C_2^k that maximizes university U_2's revenue, which can be computed by algorithm A_0. Q_1^{k+1} is set to be the subset of students remaining at U_1, namely, $Q_1^k - Q_2^{k+1}$, that maximizes U_1's revenue, which can be computed by algorithm A_0. The algorithm terminates in the first round when there is no change in enrollment; that is, when $Q_1^k = Q_1^{k+1}$ and $Q_2^k = Q_2^{k+1}$.

Lemma 5.2. *Let T be a subset of students that U_1 disenrolls in some iteration of algorithm A_4. Disenrolling any subset others than T will make less profit for U_1.*

Proof. Any subset that makes U_1 the most profit by disenrolling should contain the least valuable students of U_1. If that is not the case then we can replace some of the students in that subset with students that have less value. The algorithm disenrolls the k least valuable students and selects k to be a number that maximizes the profit. This means that U_1 cannot make more profit by disenrolling another subset. □

Lemma 5.3. *Let T be a subset of students that U_1 disenrolls in some iteration of algorithm A_4. U_1 cannot enroll any subset of T if the number of students assigned to U_1 does not increase.*

Proof. If U_1 could make positive profit by adding a subset of T then T would not be the best subset for U_1 to disenroll and U_1 could make more profit by disenrolling a subset of T.

Furthermore, since the cost function is concave, for all positive integers a and b and x such that $a < b$ we know

$$F(a + x) - F(a) > F(b + x) - F(b). \tag{1}$$

So we can conclude that if the number of students assigned to U_1 does not increase, then U_1 cannot make profit later by enrolling any subset of T. □

Lemma 5.4. *Let T be the set of students U_1 disenrolls during the execution of A_4. In the assignment A_4 produces, U_1 cannot make more profit by enrolling any subset of T.*

Proof. Let Q be the best set of students that U_1 can enroll and make the most possible profit. Let T_i be the set of students who are disenrolled in the i^{th} iteration of A_4. We can show $Q \cap T_1 = \emptyset$. Because, if the intersection is not empty, we can add $Q - T_1$ to the students of U_1 and U_1 should prefer after that to enroll $Q \cap T_1$ to make more profit. However, even if U_1 enrolls all the students who have been disenrolled before T_1, it cannot have more students than after disenrolling in the first iteration; therefore, as a result of Lemma 5.3 it follows that U_1 cannot enroll any subset of T_1 after enrolling $Q - T_1$ to make more profit, so we can conclude $Q \cap T_1 = \emptyset$.

Similarly we can conclude that for all i it follows that $Q \cap T_i = \emptyset$. That means $Q = \emptyset$. □

Lemma 5.5. *During the iterations of algorithm A_4, U_2 cannot make any profit by disenrolling a subset of its students.*

Proof. Let Q be a set that U_2 can disenroll and make the most profit. Let T_1 be the set of students that U_2 enrolled in the first iteration. Note that even if U_2 disenrolls all of its students it cannot have fewer students than what it had in the first iteration. So based on equation 1, if $Q \cap T_1 \neq \emptyset$, then U_2 can enroll $Q \cap T_1$ again and make profit. So we can conclude $Q \cap T_1 = \emptyset$.

Similarly, we can show that for all i it follows that $Q \cap T_i = \emptyset$. Therefore, we can conclude $Q = \emptyset$. □

Theorem 5.6. *Algorithm A_4 always produces a stable assignment.*

Proof. The only set of students that might be willing to join U_1 are those who have been disenrolled. The other students either left U_1 to join U_2 or they are assigned to U_1. So based on Lemma 5.4, it follows that U_1 cannot make any profit by enrolling any subset of students. We also know U_1 does not want to disenroll anyone because otherwise the algorithm A_4 would not stop.

Furthermore, U_2 does not want to enroll any subset of students because if it could enroll more, the algorithm A_4 would not finish. We also know, based on Lemma 5.5, U_2 cannot disenroll any subset of its students to make more profit.

So we can reach the conclusion that the resulting assignment from A_4 is a stable assignment. □

6 Convex Everywhere But 0

In this variation of the problem the cost function is convex everywhere except for 0. We first show in Theorem 6.1 that there are instances with only two universities where there is no stable assignment. We then give an efficient algorithm A_5 that can determine whether an instance with two universities has a stable assignment.

Theorem 6.1. *There are instances with two universities with cost functions that are convex everywhere but 0 in which there is no stable assignment for them.*

Proof. We can make the instance I with two universities and to make any example with more than two we can just add some universities with extremely high initial cost to our instance I.

Assume we have two universities and three students. Both of the universities have the following cost function:

$$F(k) = \begin{cases} 0 & k = 0 \\ 100 & k = 1 \\ 110 & k = 2 \\ 200k & k > 2 \end{cases}$$

The valuation functions of the universities are $v_1 = (s_1, 101), (s_2, 15), (s_3, 11)$ and $v_2 = (s_3, 101), (s_2, 15), (s_1, 11)$. Furthermore S_1 and S_2 prefer U_2 over U_1 and S_3 prefers U_1 over U_2. We can verify there is no stable assignment by checking the following categories of possible assignments: (1) any assignment with one unassigned student, (2) all the students assigned to one university, and (3) two of the students are assigned to one university and one student to another university.

In all of the above categories either a university is making negative revenue or a university can enroll a student who is willing to join. $\square$

Algorithm A_5: The algorithm first checks whether both universities being closed is a stable assignment using algorithm A_1. The algorithm then checks whether there is a stable assignment where only university U_1 is open (which means it has a positive number of students assigned to it). If there is such a stable assignment then U_1 is assigned its maximum revenue subset of all the students, which can be computed using algorithm A_0. Whether this assignment is stable can be computed using algorithm A_1. Similarly the algorithm can check whether there is a stable assignment where only university U_2 is open. Then the algorithm checks if there is a stable assignment where both universities are open. To accomplish this, the algorithm changes the cost functions of the universities to be $F'(0) = 0$ and for $k \geq 1$, $F'_i(k) = F_i(k) - F_i(1)$. Note that the new cost function is everywhere convex. The algorithm then finds university optimal stable assignment for the new cost function using algorithm A_3. If this assignment is stable for the original cost function, then the algorithm outputs this assignment, otherwise the algorithm declares that no stable assignment exists.

Lemma 6.2. *There is only one possible stable assignment with U_i being open and U_j being closed and that assignment is the one returned by algorithm A_0 for U_i.*

Proof. If we do not choose the best subset for U_i then U_i will prefer to make more profit by adding, replacing, or disenrolling a subset of students. So there is no other possible stable assignment with U_i open and U_j closed. $\square$

Lemma 6.3. *Let I be an instance of the problem that has 2 universities with cost functions that are convex everywhere except for 0. Let I' be an instance resulting from changing the cost functions to be $F_i'(k) = F_i(k) - F_i(1)$. Then I has stable assignment if and only if the university optimal stable assignment of I' is stable for I.*

Proof. The proof has two sections: first, we prove that any stable assignment in I is also stable in I'. Then, we prove that if M is a stable assignment in I and I' then any stable assignment in I' that has more profit for both universities is a stable assignment in I. Note that in I', since the cost functions are convex everywhere except for 0, if a university prefers to disenroll a subset of students that are not all of its students, then it also prefers to disenroll any individual student from that subset.

Let M be a stable assignment in I and let Q_i be the list of students enrolled to U_i in M. We can show that if we use the same assignment for I', then no university wants to disenroll any subset of their students and no university can make more profit by adding a student or replacing one of its students. Then we know U_i does not want to disenroll any of the students and does not want to shut down the university, which means $\forall s \in Q_i : v_i(s) > F_i(|Q_i|) - F_i(|Q_i - 1|)$ and $\sum_{s \in Q_i} v_i(s) > F_i(|Q_i|)$. We can show the same inequalities are correct for I. Note that

$$\begin{aligned}
&F_i'(|Q_i|) - F_i'(|Q_i - 1|) \\
=&F_i(|Q_i|) - F_i(1) - F_i(|Q_i - 1|) + F_i(1) \\
=&F_i(|Q_i|) - F_i(|Q_i - 1|).
\end{aligned} \tag{2}$$

Therefore, $\forall s \in Q_i : v_i(s) > F_i'(|Q_i|) - F_i'(|Q_i - 1|)$ and $\sum_{s \in Q_i} v_i(s) > F_i(|Q_i|) > F_i'(|Q_i|)$. We also know that no university can add a student to make profit. Based on Equation 2, we know if none of the students who are willing to join U_i have value more than $F_i(|Q_i|+1) - F_i(|Q_i|)$, then none of them can have more value than $F_i'(|Q_i|+1) - F_i'(|Q_i|)$. Furthermore, if U_i is not willing to replace one of its students in I, then it is not willing to replace any of its students in I' because U_i has the same set of students. Therefore, we can conclude that M is stable on I'.

Now we need to show that any other stable assignment M' on I' that makes more profit than M for all the universities is also stable on I. We know in M' on I' no universities prefer to get a new student or disenroll a single student. Because of Equation 2 we can conclude the assignment M' on I guarantees that no university prefers to get a new student or disenroll a single student. The only condition that can possibly make instability for M' on I is the case that a university prefers to disenroll all of its students. This is impossible because M was stable on I and in M' all the universities are making more profit.

Note that the cost functions in I' are convex everywhere which means we can find a university optimal assignment for it. If there is any stable assignment on I with both the universities open, then the university optimal assignment which results from running the proposed algorithm in the second section on I' will be a stable assignment. □

Theorem 6.4. *For instances with two universities with cost functions convex everywhere but 0, A_5 returns a stable assignment if and only if there exists one.*

Proof. The algorithm A_5 first tries to find a stable assignment with zero universities open. Based on Lemma 6.2 A_5 will find a stable assignment with one university being open if and only if there exists one. By Lemma 6.3 A_5 will find a stable assignment with both the universities open if and only if there exists one. □

7 Conclusion

There are many natural follow-up questions in this for-profit admissions model. Perhaps the most prominent algorithmic ones are:

- Determine if there is a polynomial time algorithm to determine whether an instance admits a stable assignment if the cost functions are concave.
- Determine if there is a fixed parameter tractable algorithm for cost functions that are convex everywhere but zero, where the parameter is the number of universities.

Acknowledgments: Our interest in this problem arose out of discussions with Claire Mathieu about the revision of the French university assignment mechanism, and the differences in university motives between France and the United States. We also thank Amos Fiat, Adi Rosén, and Vincent Cohen-Addad for helpful discussions.

References

[1] V. Bansal, A. Agrawal, and V. S. Malhotra. Polynomial time algorithm for an optimal stable assignment with multiple partners. *Theor. Comput. Sci.*, 379(3):317–328, July 2007.

[2] P. Biró, T. Fleiner, R. W. Irving, and D. F. Manlove. The college admissions problem with lower and common quotas. *Theor. Comput. Sci.*, 411(34-36):3136–3153, July 2010.

[3] F. Brandt, V. Conitzer, U. Endriss, J. Lang, and A. D. Procaccia. *Handbook of Computational Social Choice.* Cambridge University Press, New York, NY, USA, 1st edition, 2016.

[4] S. Brint. Amid recession, some college admissions policies look at students' wealth. *Washington Post*, January 10, 2010.

[5] B. Bumsted. Pennsylvania's state-related colleges prepare for worst as budget impasse drags. *TribLive*, March 2, 2016.

[6] F. Echenique and J. Oviedo. A theory of stability in many-to-many matching markets. *Theoretical Economics*, 1(2):233–273, 2006.

[7] P. Eirinakis, D. Magos, I. Mourtos, and P. Miliotis. Finding all stable pairs and solutions to the many-to-many stable matching problem. *INFORMS Journal on Computing*, 24(2):245–259, 2012.

[8] P. Eirinakis, D. Magos, I. Mourtos, and P. Miliotis. Finding a minimum-regret many-to-many stable matching. *Optimization*, 62(8):1007–1018, 2013.

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

[10] P. Gallager. *University of Pittsburgh Budget Presentation to the General Assembly of the Commonwealth of Pennsylvania*, 2016.

[11] B. Klaus and M. Walzl. Stable many-to-many matchings with contracts. *Journal of Mathematical Economics*, 45(7):422 – 434, 2009.

[12] H. Konishi and M. U. nver. Credible group stability in many-to-many matching problems. *Journal of Economic Theory*, 129(1):57 – 80, 2006.

[13] R. Lord. The branding of the american mind': How colleges and universities turn student researchers into profit. *Pittsburgh Post-Gazette*, March 5, 2017.

[14] D. F. Manlove, R. W. Irving, K. Iwama, S. Miyazaki, and Y. Morita. Hard variants of stable marriage. *Theor. Comput. Sci.*, 276(1-2):261–279, Apr. 2002.

[15] R. Martinez, J. Masso, A. Neme, and J. Oviedo. An algorithm to compute the full set of many-to-many stable matchings. *Mathematical Social Sciences*, 47(2):187–210, 2004.

[16] A. E. Roth. Stability and polarization of interests in job matching. *Econometrica*, 52(1):47–57, 1984.

[17] B. Schackner. What if pitt were to go private? some ponder the unthinkable as dwindling state aid becomes less certain. *Pittsburgh Post-Gazette*, February 23, 2018.

[18] A. Sostek. Need-blind admissions may be reconsidered by colleges. *Pittsburgh Post-Gazette*, February 14, 2013.

[19] M. Sotomayor. Three remarks on the many-to-many stable matching problem. *Mathematical Social Sciences*, 38(1):55 – 70, 1999.

Max Bender
University of Pittsburgh
Pittsburgh, PA
Email: `mcb121@pitt.edu`

Kirk Pruhs
University of Pittsburgh
Pittsburgh, PA
Email: `kirk@cs.pitt.edu`

Alireza Samadian
University of Pittsburgh
Pittsburgh, PA
Email: `samadian@cs.pitt.edu`