Algorithmic Game Theory: College Admissions Problem

Michael Levet

January 10, 2016

1 Introduction

This tutorial introduces the College Admissions problem, as well as several algorithmic solutions. The College Admissions problem extends the Stable Marriage problem by allowing for a many-to-one matching. The Stable-Marriage problem is a one-to-one matching problem in which each proposer may be matched with at most single acceptor, and vice-versa: e.g., one man and one woman, one firm and one employee, or one student and one school. More realistically, a firm hires many desirable employees, and a college admits many students. To this end, we extend the notion of a one-to-one matching to allow for a many-to-one matching. The College Admissions problem provides a model allowing for a college to be matched with multiple students, but students may only be matched with one college. This is also more realistic of how firms hire workers. I assume familiarity with the Stable Marriage Problem and the Gale-Shapley algorithm.

2 Model

The College Admissions problem starts with two disjoint sets: a set S of students and a set C of colleges. Each student $i \in S$ has a strict, transitive preference relation $\succ^i$ over the set $C \cup \{\emptyset\}$. By convention, if an agent x prefers $\emptyset$ to another agent y, then x prefers being unmatched than to being matched with y. Each college $c \in C$ also has a strict, transitive preference relation $\succ^c$ over $S \cup \{\emptyset\}$. Additionally, each college $c \in C$ has a capacity $q_c \in \mathbb{Z}^+$ of students it can admit. The solution is a matching between colleges and students, which is formalized as follows:

Definition 2.1 (Many-to-One Matching)**.** Let S and C be sets. A many-to-one matching from C to S is a function $\phi : C \to 2^S$ such that for any distinct c_1 and c_2 in C, we have $\phi(c_1) \cap \phi(c_2) = \emptyset$. Furthermore, if $c_i \in C$ has capacity q_i, then $|\phi(c_i)| \leq q_i$.

Applying this definition to the College Admissions Problem, a Many-to-One matching function ϕ relates a college c to a subset of students it admits. The matching ϕ is further constrained to prohibit a student from being matched with multiple colleges. Lastly, ϕ does not allow college c to admit more than q_c students.

The definition of a Many-to-One Matching does not account for the actors' preferences. To this end, the notion of stability will be introduced. Stability in the College Admissions problem is analogous to that in the Stable Marriage problem. First, define the mate function to return a student's enrolled college: $\mu : S \to (C \cup \{\emptyset\})$ by:

$$\mu(s) = \begin{cases} c : & s \in \phi(c) \\ \emptyset : & \text{Otherwise} \end{cases} \tag{1}$$

Definition 2.2 (Stable Matching)**.** A Many-to-One Matching of colleges and students $\phi : C \to 2^S$ is stable if and only if the following conditions hold:

1. For any college c and student s such that $s \in \phi(c)$, s and c prefer being matched with each other to being unmatched.

2. There does not exist a student s and college c such that $s \notin \phi(c)$, but $c \succ^s \mu(s)$, $s \succ^c \emptyset$ if $|\phi(c)| < q_c$, and $s \succ^c s'$ for some $s' \in \phi(c)$ if $|\phi(c)| = q_c$.

Intuitively, a stable matching satisfies everyone. So no student-college pair should want to deviate and improve their outcomes. The definition of a stable matching prohibits any condition in which a student-college pair would want to match with each other over their current mates in a given matching. Consider a many-to-one matching ϕ. The first condition describes when it is preferable for a student and college not to be matched. The second condition describes then prohibits a student-college $s \in S$ and $c \in C$ pair not matched by ϕ to prefer matching with each other over their mates in ϕ. Clearly, s must prefer c over its mate in ϕ to deviate. The analysis of the college takes a little more work due to the fact that it can admit multiple students. If the college c has room for s (i.e., $|\phi(c)| \leq q_c$), it can and should admit s. If c has fulfilled its capacity, it must check if s is a better choice than one of its already admitted students. If so, c replaces its least preferred admitted student with s.

Let's consider an example.

Example 1: Suppose we have five student $S = \{s_1, ..., s_5\}$ and three colleges $C = \{c_1, c_2, c_3\}$. Each student i has the preference relation $\succ^i := (c_1, c_2, c_3)$ (that is, c_1 is the most preferred and c_3 is the least preferred). The colleges have the following preferences and capacities:

- College c_1 has the preference relation $\succ := (s_1, s_2, s_3, s_4, s_5)$ with capacity $q_{c_1} = 2$.
- College c_2 has the preference relation $\succ := (s_5, s_4, s_3, s_2, s_1)$ with capacity $q_{c_2} = 1$.
- College c_3 has the preference relation $\succ := (s_1, s_2)$ with capacity $q_{c_3} = 1$.

Consider the matching $\phi : C \to 2^S$ given by $\phi(c_1) = \{s_1, s_2\}$, $\phi(c_2) = \{s_5\}$, and $\phi(c_3) = \emptyset$. Observe that c_1 is matched with its top two choices, and s_1 and s_2 are matched with their top choices. Since $\phi(c_1)$ has reached its capacity of two students, it cannot deviate and improve its outcome.

Now consider c_2. Observe that ϕ gives c_2 its top choice, s_5. The only college s_5 prefers to c_2 is c_1. However, c_1 prefers both of its admitted students over s_5 and has no room for s_5. Thus, s_5 cannot match with another college c_1 or c_3 which will improve both s_5's outcome and the college's outcome.

Next, consider c_3. According to c_3's preference relation, it will only match with s_1 and s_2. Since s_1 and s_2 are already matched with c_1, which they prefer to c_3, it follows that c_3 will not admit any students in a stable matching.

Finally, consider s_3 and s_4. By the above analysis of c_3, s_4 and s_5 can only match with c_1 or c_2. However, both c_1 and c_2 have filled their admittance capacities. So s_4 and s_5 cannot be admitted to any college. So there exists no student-college pair which can mutually deviate and match with each other over their mates in ϕ. Additionally, any student-college pair that are matched under ϕ prefer each other to being unmatched. Thus, ϕ is a stable matching.

The definition of the core will be recalled, but I direct readers to my previous tutorial on the Stable Marriage Problem for further exposition on the core.

Definition 2.3 (Core)**.** Let $x = (x_1, ..., x_n)$ be an allocation. The allocation $y = (y_1, ..., y_n)$ is said to dominate or block x if there exists a coalition $S \subset N$ such that for every agent $i \in S$, $y_i \succeq^i x_i$; and for at least one $j \in S$, $y_j \succ^j x_j$. The *Core* contains the set of allocations x such that no other allocation y dominates x.

Recall from the Stable Marriage Problem that the unique stable one-to-one matching constitutes the core. The result extends to the College Assignment Problem in the case of many-to-one matchings, which is shown below.

Theorem 2.1. *Let S be the set of students, and let C be the set of colleges. Let $\mathcal{M} = \{\phi : C \to 2^S : \phi$ is a stable matching $\}$ and let $\mathcal{C}$ be the core of the College Admissions problem. We have $\mathcal{M} = \mathcal{C}$.*

Proof. Let $\phi : C \to 2^S$ be a stable matching. Suppose to the contrary ϕ is not in the core. Then there exists a blocking coalition. By the definition of stability, no agent prefers being unmatched to its mate in ϕ. So no individual will form a blocking coalition. Note that students only match with colleges, and colleges only match with students. It follows that no coalition consisting solely of students or solely of colleges will form a

blocking coalition. Let $\{x_1, ..., x_n\}$ be a coalition blocking ϕ consisting of both students and colleges. From the definition of a blocking coalition, at least one agent x_i in the coalition strictly improves its outcome. As each agent's preferences are strict, agent x_i's new mate in the blocking coalition must also strictly improve its outcome over ϕ, contradicting the definition of stability. It follows that $\mathcal{M} \subset \mathcal{C}$.

Now let x be a core allocation. As no coalition exists blocking x, no individual prefers being unmatched to its mate in x. So condition (1) of the stable matching is satisfied. Suppose x is not a stable matching. Then there exists a student-college pair that would strictly benefit from matching with each other over their mates in x. This student-college pair would form a coalition blocking x, contradicting the fact that x is a core allocation. So $\mathcal{C} \subset \mathcal{M}$. Thus, $\mathcal{M} = \mathcal{C}$. □

In order to find a stable matching in the College Admissions problem, the Gale-Shapley algorithm is adapted to derive two new algorithms: the *Student-Optimal Deferred Acceptance (SODA)* and *College-Optimal Deferred Acceptance (CODA)* algorithms. Recall the Gale-Shapley algorithm favors the proposers. The SODA algorithm is the analogue of the Gale-Shapley algorithm when the students propose, and the CODA algorithm is the analogue of the Gale-Shapley algorithm when the colleges propose. In fact, the CODA algorithm is how many colleges admit their students in the United States. We begin by introducing the SODA algorithm.

3 Student-Optimal Deferred Acceptance

The Student-Optimal Deferred Acceptance algorithm works quite similarly to the Gale-Shapley algorithm. Each student takes a turn proposing to a college. If the college has not fulfilled its quota, it admits the student only if it is preferable to being unmatched. If the college has fulfilled its quota and the student is more preferable than an already admitted candidate, the college admits the student applicant and revokes acceptance to its least preferred admittant. Let's work through an example using the SODA algorithm.

Example 2: Suppose we have six students and three colleges, with each college having a capacity of 2. The preferences are as given below:

- Students 1 and 4 with preferences: $\succ := (Y, X, Z)$.
- Students 2 and 5 with preferences: $\succ := (X, Z, Y)$.
- Students 3 and 6 with preferences $\succ := (X, Z, Y)$.
- Colleges X and Y with preferences $\succ := (2, 5, 3, 6, 1, 4)$.
- College Z with preferences $\succ := (1, 4, 2, 5, 3, 6)$.

The SODA algorithm proceeds as follows:

- Student 1 proposed to College Y. College Y accepted the proposal from Student 1.
- Student 2 proposed to College X. College X accepted the proposal from Student 2.
- Student 3 proposed to College X. College X accepted the proposal from Student 3.
- Student 4 proposed to College Y. College Y accepted the proposal from Student 4.
- Student 5 proposed to College X. College X accepted the proposal from Student 5 and unmatched from Student 3.
- Student 6 proposed to College X. College X rejected the proposal from Student 6.
- Student 6 proposed to College Z. College Z accepted the proposal from Student 6.
- Student 3 proposed to College Z. College Z accepted the proposal from Student 3.

The final matching is $\{(X, \{2, 5\}), (Y, \{1, 4\}), (Z, \{3, 6\})\}$.

Sample code is provided in the main tutorial. A proof of algorithm correctness will be provided.

Theorem 3.1. *The SODA algorithm terminates, resulting in a stable matching that is student-optimal.*

Claim 3.1.1. *The SODA algorithm terminates.*

Proof. Suppose that some student s is unmatched after iteration $k > |C|$. It follows that each college in C to which s has proposed has either rejected s outright, or accepted s's proposal and later unmatched from s. If a college c outright rejected s, then s need not propose to c again. If s accepted s's initial proposal and later unmatched from s, then s is matched with a second student t such that $t \succ^c s$. Furthermore, for any $x \notin \{s, t\}$ that c was matched with, it is necessary that $x \succ^c s$ (otherwise, as c is rational, it would not have unmatched from x). Agent s prefers to be unmatched than to match with any college to which it did not propose prior to iteration k (otherwise, s would have proposed to one of these colleges prior to iteration k). Thus, s need not be considered again by the algorithm. By consideration of all students, the algorithm must terminate. □

Claim 3.1.2. *The SODA algorithm produces a stable matching.*

Proof. Let ϕ be the matching returned by the SODA algorithm. Suppose to the contrary that it is not stable. By the algorithm, no student will propose to a college with which it would not match. Similarly, any college will reject the proposal from a student if it would prefer to remain unmatched over matching with the student. Thus, there exists a student s and college c such that $s \notin \phi(c)$, but $c \succ^s \mu(s)$, $s \succ^c \emptyset$ if $|\phi(c)| < q_c$, and $s \succ^c s'$ for some $s' \in \phi(c)$ if $|\phi(c)| = q_c$.

By the SODA algorithm, s would have proposed to c prior to its current mate in the matching. If $|\phi(c)| < q_c$ after the algorithm terminates, then c would would have accepted s's proposal, a contradiction. If $|\phi(c)| = q_c$ after the algorithm terminates; then by the algorithm, c would unmatch from its least preferred mate in ϕ and mate with s, a contradiction. It follows that the matching must be stable. □

Claim 3.1.3. *The matching produced from the SODA algorithm is student-optimal.*

Proof. Let ϕ be the matching returned by the SODA algorithm. Suppose to the contrary that ϕ is not student-optimal. Let ϕ' be a second stable matching which weakly improves the students' outcomes, and let s be a student such that $\mu'(s) \succ^s \mu(s)$. Then s would have proposed to $\mu'(s)$ prior to $\mu(s)$ under the SODA algorithm. Since $s \in \phi'(\mu'(s))$, $\mu'(s)$ prefers to match with s over being unmatched. However, $\mu'(s)$ unmatched from s or outright rejected s in the algorithm, implying that $\mu'(s)$ prefers $\phi(\mu'(s))$ to $\phi'(\mu'(s))$. Without loss of generality, suppose this was the first instance of a rejection or unmatching in the SODA algorithm. As this is the first instance of rejection or unmatching, there exists a student $t \in \phi(\mu'(s)) \setminus \phi'(\mu'(s))$. Furthermore, as this is the first instance of rejection or unmatching, each student in $\phi(\mu'(s))$ can have no stable partner better than $\mu'(s)$. It follows that t prefers $\mu'(s)$ to $\mu'(t)$, and so $\{t, \mu'(s)\}$ form a coalition blocking ϕ', contradicting the stability of ϕ'. It follows that ϕ is student-optimal. □

Claims 3.1.1, 3.1.2, and 3.1.3 together imply Theorem 1. Claim 3.1.3 also yields an important corollary.

Corollary 3.1.3. *The SODA algorithm returns the same matching regardless of the order in which the students propose to the colleges.*

Remark: When each college has capacity 1, this is exactly the Stable-Marriage problem. Recall the definition of a strategy proof mechanism is one in which truthfully revealing one's preferences is a weakly dominant strategy. The Stable-Marriage problem has no strategy-proof mechanism, which was shown in my previous tutorial. Therefore, the result extends to the many-to-one matching case. So while in the SODA algorithm, it is optimal for students to truthfully reveal their preferences, this is not the case for the colleges as demonstrated in the example from my previous tutorial. Furthermore, the colleges receive their worst core allocations under the SODA algorithm. This generalizes the result of the Gale-Shapely algorithm, which will be proven next.

Theorem 3.2. *Under the SODA algorithm, each college receives its least preferred core allocation.*

Proof. Let ϕ be the stable matching returned by the SODA algorithm. Suppose to the contrary that there exists a second stable matching ϕ' that grants each college its least preferred core allocation. Let c be a college such that c prefers strictly $\phi(c)$ to $\phi'(c)$. If $|\phi'(c)| > |\phi(c)|$, then any student in $\phi'(c) \setminus \phi(c)$ would have proposed to c under the SODA algorithm. As $|\phi(c)| < |\phi'(c)| \leq q_c$, c would have admitted at least one of these students. Thus, $|\phi(c)| \geq |\phi'(c)|$ necessarily. As c strictly prefers its allocation in ϕ over that in ϕ', there exists a student in $s \in \phi(c) \setminus \phi'(c)$. As ϕ is student-optimal, $\{\phi(c), c\}$ forms a coalition blocking ϕ', contradicting the stability of ϕ'. □

4 College-Optimal Deferred Acceptance

The College-Optimal Deferred Acceptance (CODA) algorithm is the analogue of the Gale-Shapley algorithm in which the colleges do the proposing rather than the students. The algorithm proceeds similarly as the SODA algorithm. It begins by selecting a college c with open slots. If there exist students with whom c will match, c proposes to them in order of preference. A student may accept a proposal or reject it. If a student is matched with a different college than the one proposing, the student must unmatch from its present mate before accepting a proposal from a new mate. Let's work through an example of the CODA algorithm.

Example 3: Recall the six students and three colleges from Example 2. The CODA algorithm proceeds as follows:

- College X proposed to Student 2. Student 2 accepted proposal from Student 2.
- College X proposed to Student 5. Student 5 accepted proposal from Student 5.
- College Y proposed to Student 2. Student 2 rejected proposal from Student 2.
- College Y proposed to Student 5. Student 5 rejected proposal from Student 5.
- College Y proposed to Student 3. Student 3 accepted proposal from Student 3.
- College Y proposed to Student 6. Student 6 accepted proposal from Student 6.
- College Z proposed to Student 1. Student 1 accepted proposal from Student 1.
- College Z proposed to Student 4. Student 4 accepted proposal from Student 4.

The final matching is $\{(X, \{2, 5\}), (Y, \{3, 6\}), (Z, \{1, 4\})\}$. Sample code is again provided in the main tutorial. We examine a proof of algorithm correctness for the CODA algorithm, which is analogous to that of the SODA algorithm.

Theorem 4.1. *The CODA algorithm terminates, resulting in a stable matching that is college-optimal.*

Claim 4.1.1: *The CODA algorithm terminates.*

Proof. Suppose there exists an unmatched college c at iteration $k > |S|$. By the CODA algorithm, each student to which c has proposed either outright rejected the proposal, or accepted the proposal then later unmatched from c. And so each student to which c has already proposed is matched with a college more preferable than c or unmatched. Therefore, c will be rejected if it proposes to these students again. Since $k > |S|$, c has proposed to all the students with whom it is willing to match. Therefore, c has no available options. By considering all such colleges, it follows that the CODA algorithm terminates. □

Claim 4.1.2: *The CODA algorithm produces a stable matching.*

Proof. Let ϕ be the matching returned by the CODA algorithm. Suppose to the contrary that ϕ is not stable. Colleges will only propose with whom they are willing to match, and students will only accept proposals from colleges with which they are willing to match. So no individual agent will form a blocking coalition by itself. So there must exist a student s and college c such that $s \notin \phi(c)$ but s and c prefer each other to their current

mates in ϕ. Under the CODA algorithm, c would have proposed to s. If $|\phi(c)| < q_c$, then s would have been added to $\phi(c)$ under the CODA algorithm. But since s was not added to $\phi(c)$, it follows that $|\phi(c)| = q_c$. Let $t \in \phi(c)$ be c's least preferred match. Since $\{c, \phi(c)\}$ blocks ϕ, $s \succ^c t$. Under the CODA algorithm, c would have proposed to s prior to t and s would have accepted, contradicting the fact that $t \in \phi(c)$. It follows that ϕ must be stable. □

Claim 4.1.3: *The matching produced by the CODA algorithm is college-optimal.*

Proof. Let ϕ be the matching returned by the CODA algorithm. Suppose to the contrary that there exists a second stable matching ϕ' which weakly improves upon the colleges' outcomes. Let $c \in C$ such that $\phi'(c) \succ^c \phi(c)$. Then there exists a student $s \in \phi'(c) \setminus \phi(c)$ such that c prefers s to its least preferred mate $t \in \phi(c)$. Then under the CODA algorithm, c would have proposed to s prior to t. However, s must have rejected c. Without loss of generality, suppose this is the first instance of rejection. As ϕ' is stable, s prefers being matched with c over being single. So s must be matched with another college x under ϕ. Since this is the first instance of rejection, it follows that x can have no better set of mates than $\phi(x)$. So $\{x, \phi(x)\}$ forms a blocking coalition of ϕ', contradicting the stability of ϕ'. □

Remark: Just as the SODA algorithm is not strategy proof, neither is the CODA algorithm. Furthermore, the CODA algorithm grants the students their worst stable matching, analogously to the SODA algorithm with respect to the colleges. This final result will now be proven.

Theorem 4.2. *Under the CODA algorithm, each student receives its least preferred core allocation.*

Proof. Let ϕ be the stable matching returned by the CODA algorithm. Suppose to the contrary that there exists a second stable matching ϕ' granting each student its least preferred core allocation. Let s be a student such that s strictly prefers its mate in ϕ over its mate in ϕ'. As ϕ is college-optimal, $\{\mu(s), \phi(\mu(s))\}$ blocks ϕ', contradicting the stability of ϕ'. □

5 Conclusion

In this tutorial, the notion of a one-to-one matching was extended to a many-to-one matching. Two extensions of the Gale-Shapley algorithm were explored, including the Student-Optimal Deferred Acceptance and College-Optimal Deferred Acceptance algorithms. We examined sample implementations, as well as proofs of correctness for these algorithms. The College-Admissions problem has an important extension, where Students signal their viabilities to colleges through the use of test scores. In certain approaches, Students may benefit by under-performing on certain tests. So the matching problem can be extended to design a mechanism rewarding students for doing their best on tests. Additionally, the natural question arises of how to extend the many-to-one matching problem to the case of many-to-many matchings.