

---

### Greedy alg proofs

Where  $S$  is the ordered set of students,  $C_s$  is the preference list of  $s$ ,  $X$  is a set of unmatched students in the first round,  $C$  is the set of classes and “available” means it has open seats:

---

#### Algorithm 1 Greedy Algorithm ( $GA$ )

```

1: for each student  $s$  in  $S$  do                                ▷ start of round 1
2:   for each class  $c$  in  $C_s$  do
3:     if  $c$  is available then
4:        $\mu(s) \leftarrow c$ 
5:       break                                         ▷ move to next student
6:     end if
7:   end for
8:   append  $s$  to  $X$ 
9: end for
10: for each student  $s$  in  $X$  do                               ▷ start of round 2
11:   choose random available  $c$  from  $C$ 
12:    $\mu(s) \leftarrow c$ 
13: end for
14: return matching  $\mu$ 

```

---

**Theorem**  $GA$  outputs a stable matching in a vanilla environment.

**Proof** Let  $\mu$  denote the outputted matching over an ordered set of students,  $S$ . Let  $a, b \in S$ , where  $a$  comes before  $b$  in the order. Assume for the sake of a contradiction that  $a$  and  $b$  form a blocking pair, such that  $\mu(b) >_a \mu(a)$  and  $\mu(a) >_b \mu(b)$ .

By lines 3 and 11 of  $GA$ , we know  $\mu(a)$  was either  $a$ 's top available choice or a random assignment. So, if  $\mu(b) >_a \mu(a)$ , then  $\mu(b)$  must not be available when  $a$  is matched, which means  $b$  was matched first. However, this is a contradiction of our assumption that  $a$  comes before  $b$  in the input to the algorithm. Thus,  $\mu(b)$  is available when  $a$  is matched, so it must be the case that  $\mu(a) >_a \mu(b)$ . Therefore,  $a$  and  $b$  do not form a blocking pair, and we can conclude that  $\mu$  is stable.

**Theorem**  $GA$  does not always output a stable matching with quotas (caps).

**Proof** Let  $\mu$  denote  $GA$ 's outputted matching over a set of students,  $S$ . We will prove by counter example that  $\mu$  is not stable.

Let  $C = \{c_1, c_2, c_3\}$ , where there are two types of student  $\alpha$  and  $\beta$  such that each  $c \in C$  can have at most one student of each type. Let  $S = \{A1, B1, A2, B2\}$  where students  $A1, A2$  are of type  $\alpha$  and  $B1, B2$  are of type  $\beta$ . Let the preference lists of the students be as follows:

$$\begin{aligned} A1 : \quad & c_1 \succ c_3 \\ B1 : \quad & c_2 \succ c_3 \\ A2 : \quad & c_1 \succ c_2 \\ B2 : \quad & c_2 \succ c_1 \end{aligned}$$

Observe that in  $\mu$ ,  $\mu(A1) = c_1$  and  $\mu(B1) = c_2$ , because they are the first two students to be matched and their first choices are different, so they receive their top

choices. Next,  $\mu(A2) = c_2$  because  $c_1$  already has a student of type  $\alpha$  so  $c_1$  is no longer available when matching  $c_\alpha$ , and similarly  $\mu(B2) = c_1$  because  $c_2$  already has a student of type  $\beta$ . Notice that  $\mu(B2) >_A 2\mu(A2)$  and  $\mu(A2) >_B 2\mu(B2)$ , forming a blocking pair in which  $A2$  and  $B2$  are incentivized to switch classes with each other. So, it must be the case that  $\mu$  is not a stable matching and therefore  $GA$  cannot guarantee a stable matching under quota/cap constraints.

**Theorem** There is not always a stable matching with quotas (caps) and incomplete preferences.

**Proof** Take the above instance, and note that if we alter the environment slightly there is no stable matching. Remove  $c_3$  from  $C$  so that  $C = \{c_1, c_2\}$  and let the preference lists be as follows:

$$\begin{aligned} A1 : & c_1 \\ B1 : & c_2 \\ A2 : & c_1 \succ c_2 \\ B2 : & c_2 \succ c_1 \end{aligned}$$

In this case, regardless of the order of matching or algorithm, there is no stable matching. The  $A$  students will both wish to be matched to  $c_1$ , which only has one  $\alpha$  spot, and both  $B$  students wish to be matched to  $c_2$  which only has one  $\beta$  spot. So, when the first round of matching is done, that means that there is still an open  $\beta$  spot for  $c_1$  and an open  $\alpha$  spot for  $c_2$ , which the unhappy  $A$  and  $B$  students can then switch into on their own, respectively.

Clearly, our problem as it stands is too constrained. We want to work in an environment where a matching is always possible. We need to start from a more basic constraint environment, where instead of ranked preferences we have 0/1 preferences and therefore no notion of stability (for now). In this case, it becomes a weighted graph matching problem. Let  $C$  denote the set of seats, and  $S$  the set of students. We will put lower bounds on the seats, such that the total number of seats  $/|C|$  will be reserved for each type of student. Say that an edge  $e$  exists between a seat and a student if the student prefers that class, and each  $e_i$  has a weight  $w_i$ . Let  $w_i + 1$  if the seat is reserved for that student, and  $w_i + 1$  if the student prefers that class. So, every edge will have at least  $w = 1$  and at most  $w = 2$ .

**Theorem** If a matching exists, the weight of the max-weight graph will equal  $|C| + |S|$ .

**Proof** Let there be  $n$  types of students,