Draw Algorithms

by Sourav Mondal

May 29, 2020

Content

What is draw algorithm?
About

- The related branch of game theory is often referred to as incentives. So what do we mean by “incentives,” anyway? Let’s start with an example that should resonate with many of you — the Draw, meaning the process that assigns students to rooms in dorms and houses. Meaning draw is a process of allocating some items or activities or work to the the person with different preferences , allowing him to allocate his best choice available for him as a incentives.
- Incentives are given to students with their true choices as their best choice of room available at that point of time.

What is draw algorithm?

Here's how the Draw works:

The Draw

- 1. Each student submits a ranked list. Each item in the list is a type of room in a specific dorm or house (like a Toyon two-room double), and they are ordered from most preferred (at the top) to least preferred. (No ties are allowed.) This is how you express your preferences—that is, what you want—to the algorithm implementing the Draw
- 2.Each student is assigned a number in 1, 2, . . . , 3500.
- 3. For i = 1, 2, . . . , 3500: (a) Student i is assigned to her favorite choice among the options still available. In effect, the algorithm goes down student i's ranked list (from top to bottom), giving her the first option that is available

Why need of draw algorithm.

- Suppose we have a list of students and some rooms to allocate them. Now if we ask students for their preferred room number it might happen that some number of student have same preference of room.
- To deal with this situation we have come up with a solution known as draw which leads to the optimal solution for allocating rooms. According to the draw algorithm student will be asked to submit their list of preferred choices for rooms instead of preferred room only. Thus applying draw algorithm will give them their best choice of room available for them.
- This is how we can get the room allocated to the students easily and solve a game theory challenge optimally, which is not possible to get manually for large number of students.
- It also produces a pareto optimal and strategy proof solution.

Pareto Optimality

The outcome of the Draw is Pareto optimal

Pareto optimal (adj.): the property of an outcome that you can't make anyone better off without making someone else worse off.

- Pareto Optimality is a property of the draw system.
- Call the outcome of the Draw the "old" outcome. Consider some alternative "new" outcome. Assume that no student is worse off in the new assignment than in the old (otherwise the new assignment is no threat to the Draw's Pareto optimality). The plan is to show that the two assignments must be identical. Thus every assignment different from the outcome of the Draw makes someone worse off, and this establishes Pareto optimality.

Strategyproof Mechanisms

strategyproof

the property of a mechanism that honesty is always the best policy, meaning that lying about your preferences cannot make you better off.

- Draw numbers are generated independently of the ranked lists that students submit. So you can't affect the draw number i that you get. You also can't affect the choices made by the i1 students before you — their choices are independent of the ranked list that you submit. Thus, the available options at the time you are considered by the Draw does not depend on your submitted list. Since the Draw awards you your highest-ranked available option, it pays to be honest — any lie in your ranked list could only cause you to instead receive a remaining option that is not your favorite.
- The Draw is an example of a mechanism.

The Draw is strategyproof

mechanism

a procedure for making a decision or taking an action, as a function of what people want (i.e., of participants' preferences).

serial dictatorship

a mechanism that orders the participants, and in this order allows each player to dictate their favorite feasible option (given the choices made by previous players).

The Draw in the 1990s

The draw is being used since early 90's. Now someone can implement draw just adding a number called draw number. The rules of the Draw have changed many times over the years, most recently in 2009.

The Draw (Mid-90s Version)

1. Each student submits a ranked list of at most 8 options, out of the roughly 60 possibilities.
2. Same as before (modulo unimportant details): each student is assigned a number from some range, and these numbers are chosen independently of the submitted lists.
3. Same as before. But if none of student i's 8 options are still available, then the student is unassigned, or assigned by default to the least popular dorm.

Why Care About Strategyproofness?

- A mechanism needs to be strategyproof.Think of it this way: today, when you submit your housing preferences, you have to do a fair amount of soul-searching to figure out what you really want. But this is a noble type of hard thinking — any mechanism worth its salt needs to solicit your preferences. Back in the 1990s, you additionally had to think hard about what options were likely to be available at the time you were considered.
- Another problem with non-strategyproof mechanisms like the mid-90s Draw is regret, meaning that in hindsight a participant wishes they could go back and submit different preferences. (In today's Draw, you might wish you could have gotten a better draw number, but you'll never regret submitting your true ranked list.) Certainly your instructor's draw group regretted in hindsight not putting a safer safety in the eighth slot. Relatedly, the mid90s Draw could easily produce non-Pareto optimal outcomes .

Example of Draw Algorithm

Lets Consider a interview time slot allocation as an example to explain how draw algorithm works. Suppose we have a n number of interview time slot to be allotted for n number of students.Every candidate will be matched with the interviewer at their given time allotted. Now every candidate have some preference of interview slot time for the interviews. Like wise given below Four interview time slots (A,B,C,D) to be allotted to Four candidate.
Candidate 1 has preference : $D > C > B > A$
Candidate 2 has preference : $D > B > C > A$
Candidate 3 has preference : $B > D > C > A$
Candidate 4 has preference : $B > C > D > A$

Example of Draw Algorithm

Without using draw algorithm for allocation

- The algorithm used to allocate time slots pays a crucial role. If we consider the last choice of the available time slot to be given to the candidate as algorithm then candidate 1 will get A time slot,candidate 2 will get C, Candidate 3 will get D and candidate 4 will get B as time slot for the interview.
- In this case candidate can manipulate their time slot (Choosing the best choice at the last) choice to get their bast choice. In this case the algorithm is not strategyproof. For student without their true choice list will get the best choice just by putting it in last. Hence no optimal time slot as incentives.
- Again if we alter the choice list we might get a better allocation, Like taking choice list for candidate 2 first the for rest bottom to top choice to allocate then their allocation will be better. We will still get some better solution without making someone worse off.Thus the solution is not pareto optimal too.

Example of Draw Algorithm

Using draw algorithm for allocation

- If we use draw algorithm here then first of all we will add some randomly generated draw number to the end of the choice list. Then we will sort the preference list according to the ascending draw numbers.
- In this case draw numbers are generated according to the draw algorithm is - candidate 2 got draw number 1, Candidate 3 got draw number 2, candidate 3 draw number 3 and candidate 1 got draw number 4.
- Now we will be allocating time slot according to the draw numbers. So candidate 2 will get its best choice D, candidate will also get his best choice B , rest of candidate 4 and 1 will get C and A time slot for the interview.
- If we look at the allocation then then this the best allocation possible for interview timing, as if we alter any of the preference order then the allocation will change and the no more better solution can be found without changing others choice. Thus the solution is pareto optimal.

Example of Draw Algorithm

Using draw algorithm for allocation

- Now if we change our preference list or try to temper the time slot list by giving wrong choices then the solution for the draw algorithm will be different.Suppose candidate 3 and 4 got same choice list, then overall optimality of time slot allocation will be reduced.Thus candidate will get worse choice for tempering the choice list. If candidates submit only true choice list then only they will get incentives as a form of best available time slot for the interview.Thus this solution is strategyproof.
- As the solution is pareto optimal and strategyproof thus both the property of draw holds, and cnadidate will get the optimal solution for the time solution as incentives.

Implementation of draw algorithm.

Python

- A draw algorithm can be implemented by assigning a draw number to the student choice list.
- First, every student have to submit their list of preference for rooms.
- A algorithm will assign a random number(Draw Numbers) to each of the list and then the list will be sorted according to the ascending order of the draw numbers
- Student i is assigned to her favorite choice among the options still available. In effect, the algorithm goes down student i's ranked list (from top to bottom), giving her the first option that is available.

```
import random

#Used For generating random number list used as draw numbers
def randomNumbers(Numbers, size):
        Num=Numbers
        random.shuffle(Num)
        return Num

#Used for sorting choice list by draw number
def sortDrawList(choiceWithDrawNum):
        choiceWithDrawNum.sort(key=lambda num:num[1])



def generateStudentChoice(n):
        subList=[]
        for i in range(0,n):
                tempList=[i+1 for i in range(n)]
                randomNum=randomNumbers(tempList,n)
                subList.append(randomNum)
```

```
choiceList=[]
for i in range(n):
        string=""
        for j in range(n):
                if j==0:
                        string+=str(subList[i][j])
                else:
                        string+=" "+str(subList[i][j])
        choiceList.append(string)
print("Student Choice List:")
for i in range(n):
        print(choiceList[i])
print(" ")

f=open('input.txt','w')
for i in range(n):
        f.writelines(str(choiceList[i]))
        f.write("\n")
```

```
#The main draw algorithm function
def drawAlgoithm(studentChoice, finalAllotmentList,n):
        # #drawNumList will store the draw numbers
        generateStudentChoice(n)
        #choice=[]
        inputChoices=[]

        inputFromFile=open('input.txt','r')
        stringLine=inputFromFile.read()
        inputChoices.append(stringLine.splitlines())

        for i in range(0,n):
                tempVariable=
            [int(num) for num in
            inputChoices[0][i].split()]
                studentChoice.append(tempVariable)
        # drawNumList=[]
        drawNumbers=[i+1 for i in range(n)]
        drawNumbers=randomNumbers(drawNumbers,n)
```

```
print("Draw Numbers are:",drawNumbers[0:])
choiceWithDrawNum=[]
for i in range(0,n):
        pair=(i,drawNumbers[i])
        choiceWithDrawNum.append(pair)

sortDrawList(choiceWithDrawNum)
alloted=[0]*(n+1)

picked=0
for i in range(0,n):
        roomNow=choiceWithDrawNum[i][0]
        for j in range(0,n):
                picked=studentChoice[roomNow][j]
                if alloted[picked]==0:
                        finalAllotmentList[roomNow]
                        =picked
                        alloted[picked]=1
                        break
        print("Alloted Rooms to the student are:",
        finalAllotmentList)

def main():
        print("Implimentation of a Draw Algorithm
        for allotment of hostel room")
        studentChoice=[]
        n=int(input("Enter the number of student "))
        #Final alloted list
        finalAllotmentList=[0]*n

        drawAlgoithm(studentChoice,finalAllotmentList,n);
        print("------End-----")

if __name__=="__main__":
        main()
```

Application of draw algorithm

- College Admissions-College admissions is a two-sided market, with students on one side of the market and colleges on the other. In a two-sided market, it's not enough to choose—you must also be chosen.
- Hostel Allotments- Room assignment is a one-sided market, since only one side of the market (the students) has preferences.
- The National Resident Matching Program (NRMP)

References:

- CS269I: Incentives in Computer Science Lecture 1: The Draw and College Admissions Tim Roughgarden, September 26, 2016
- Game Theory- Alvin E. Roth and Lloyd S. Shapley for “the theory of stable allocations and the practice of market design”
- A. E. Roth. Who Gets What And Why. Mariner, 2015

The End