Algorithmic Game Theory

Giada Gabriele

July 29, 2022

1 Assignment

The system must select the user who will perform the tour, must define the places that the tour will visit and must define the payments charged to the users. The only constraint we have is the maximum payout. Costs proportional to kilometers should be equally distributed and fixed costs should be based on declared utilities, so we must implement mechanisms that lead to truthful declarations of utilities.

2 Hypothetical mechanism choice

In this section we will show how we arrived at the choice of the mechanism. Studying game theory and related theorems this is the theoretical process that led to the final choice.
We start from the definition of **what** is a Mechanism:

> A *Mechanism* (for a Bayesian game setting (N, O, Θ, p, u)) is a pair (A,M), where
>
> - A $= A_1 \times \ldots A_n$, where A_i is the set of actions available to agent $i \in N$; and

- M : $A \mapsto \Pi(O)$ maps each action profile to a distribution outcomes.

We want to pay attention on the truthfulness of the game and we know that a *direct mechanism* is one in which the only action to each agent is to announce his private information. Since in a Bayesian game an agent's private information is his type, direct mechanism have $A_i = \Theta_i$, so a direct mechanism is said to be truthful (or incentive compatible) if, for any type vector θ, in equilibrium of the game defined by the mechanism every agent i's strategy is to announce his true type, so that $\hat{\theta}_i = \theta_i$. We can restrict our attention to truthful mechanism, in this case the *Quasilinear preferences*:

> Agents have quasilinear preferences in a n-player Bayesian game when the set of outcomes is
>
> $$O = X \times \mathbb{R}^n$$
>
> for a finite set X, and the utility of an agent i given joint type θ is given by
>
> $$u_i(\text{o},\theta) = u_i(\text{x},\theta) \text{ - } f_i(p_i),$$
>
> where $o = (x, p)$ is an element of O,
> $u_i : X \times \Theta \mapsto \mathbb{R}$ is an arbitrary function and $f_i : \mathbb{R} \mapsto \mathbb{R}$ is a strictly monotonically increasing function.

Then we split the mechanism into two pieces that are linearly related:

$$u_i(o, \theta) = u_i(x, \theta) - f_i(p_i)$$

- $x \in X$ is a discrete non-monetary outcome
- $p_i \in R$ is a monetary payment that agent i must make to the mechanism

In this way agents care only about the choice selected and about their own payments, not about the monetary payments made by other agents. At this point we can define a **Quasilinear mechanism**:

> A mechanism in the quasilinear setting (for a Bayesian game setting$(N, O = X \times \mathbb{R}^n, \Theta, p, u)$) is a triple $(A, \chi, \wp)$, where
>
> - $A = A_1 \times \ldots A_n$, where A_i is the set of actions available to agent $i \in N$
> - $\chi : A \mapsto \Pi(X)$ maps each action profile to a distribution over choices, and
> - $\wp : A \mapsto \mathbb{R}^n$ maps each action profile to a payment for each agent.

We have split the function M in two functions, χ is a **choice rule** and $\wp$ is a **payment rule**. Now we exploit the **direct quasilinear mechanism** which is the only setting where each agent is asked to state his type.

> A direct quasilinear mechanism (for a Bayesian game setting$(N, O = X \times \mathbb{R}^n, \Theta, p, u)$) is a pair $(\chi, \wp)$; it defines a standard mechanism in the quasilinear setting, where each i, $A_i = \Theta_i$.

At this point we make again assumption that agents' utilities depend only on their own types exploiting the property of **conditional utility independence**:

> A Bayesian game exhibits conditional utility independence if for all agent $i \in N$, for all outcomes $o \in O$ and all pairs of joint types θ and $\theta' \in \Theta$ for which $\theta_i = \theta'_i$ it holds that
> $u_i(o, \theta) = u_i(o, \theta')$.

And again we go back at the definition of **truthfulness**, in this case for a quasilinear mechanism:

> A quasilinear mechanism is truthful if it is direct and $\forall_i \forall v_i$, agent i's equilibrium strategy is to adopt the strategy $\hat{v}_i = v_i$.

Established what is truthful for game theory, we can analyze the part relating to costs, *cost sharing*, in particular the most coherent choice for this type of problem is to look at **coalitional games** and **Shapley value**; let's go in order.

What is a coalitional game:

> In coalitional game theory our focus is on what groups of agents, rather than individual agents, can achieve. Given a set of agents, a coalitional game defines how well each group (or coalition) of agents can do for itself. We are not concerned with how the agents make individual choices within a coalition, how they coordinate, or any other such detail; we simply take the payoff (or costs, this definition is valuable also for costs - that is our case) to a coalition as given.

Now we can define some aspects related to the division of costs, starting from the definition of **feasible payoff**:

> Given a coalitional game (N,v), the feasible payoff set is defined as
>
> $$\{x \in \mathbb{R}^N | \sum_{i \in N} x_i \leq v(N)\}$$

this set contains all payoff vectors that do not distribute more than the worth of the grand coalition. We can view this as requiring the payoffs to be **weakly budget balanced**. This balance aspect responds to the goal we set ourselves at the beginning.
Therefore for this kind of issue we need to think about group of people who wants to maximize utilities and obtain as outcome the most convenient tour, in a *fair* way. Now we can exploit the definition of **Shapley Value**:

> Given a coalitional game (N,v), the Shapley Value of player i is given by:
>
> $$\phi_i(N,v) = \frac{1}{|N|!} \sum_{S \subseteq |N| \setminus \{i\}} |S|!(|N| - |S| - 1)![v(S \cup \{i\}) - v(S)].$$
>
> This expression can be viewed as capturing the “average marginal contribution” of agent i, where we average over all the different sequences according to which the grand coalition could be built up from the empty coalition.

3 Application theory

We must establish some variables of the problem, for example **how** to evaluate user preferences. Each user has a list of cities, with fixed costs necessary to visit them (the same for all) on which to express their preferences, **in what way?** Of course from a mathematical point of view this won't be exactly right, we could be more precise with a larger scale but for this example and to not complicate too much the explanation I prefer to stay on a classical ratings-style evaluation, choose a number from 1 to 5. **What's our best outcome?** First of all, a city can be visited by a particular user if that user can afford to visit that city or that tour of cities, it cannot happen that a user declares a preference and then does not have the possibility to do so, we will demonstrate in the next paragraph how to calculate it. Each rating is equivalent to the utilities of a given city, so the tour that has the highest utility (the one that the majority prefers) wins. For the same utility (if this happens) the one with the lowest cost wins, as often happens in real life, the one on which you save the most wins. At the same cost, the tour that has more cities on the list wins, I would prefer to visit as many cities as possible, as well as have as many people as possible in the vehicle.

At this point we have to clarify some variables in order to be able to respond to the problem requests in the application example part.

The assignment says that we have to create a game in which a group of people, using a vehicle with **k** sits, do a tour in some cities. We have the cities but we didn't shaped the problems based on the k constraint. So, exploiting the Shapley value formula we have to change the N variable, in our case it will be:

$$\sum_{p=1}^{k} \binom{|N|}{p}$$

So in this way we will calculate all the possible games with a group of people $\leq$ k, we remain in the coalition game setting and after calculate all the possibilities we will find our best outcome. We have also to define our **characteristic function**, v: it's **the cost** of the winning tour. What does Shapley value allow us to do indeed? We will have as a result the cost of every game and understand if that is a **valid** game, people can afford it, or not. In this way we have also a **fair** game. We don't want any participant to be unhappy, so even if the sum of declared utilities is enough to go on a certain tour is *unfair* if anyone pay, for example, 200 and an other one pay 10 just because the value of the tour is 210, everyone must pay in an "equal" way. Exploiting the Shapley value we start from the bottom, from an empty group, then we calculate the cost of singleton, then couple, etc, in this way passing through singleton we know that everyone in the game can afford it. As the cost of singleton we can pick the cost of the tour with the maximum utility. The Shapley value defined a fair way of dividing the grand coalition's payment among its members. Also it is not convienent to lie on preferences because if you can afford to go to both Milan and Rome and give 5 to Milan and 4 to Rome and you really want to go to Rome, if Milan wins with maximum utility you go to Milan, it makes no sense, given the statements made, don't tell the truth. However, this analysis ignored questions of stability so we have to define what is the **core**:

> A payoff vector x is in the core of a coalitional game (N,v) if and only if
>
> $$\forall S \subseteq N, \sum_{i \in S} x_i \geq v(S)$$
>
> Thus, a payoff is in the core if and only if no subcoalition has an incentive to break away from the grand coalition and share the payoff it is able to obtain independently.

At this point we have to clarify some issues.

We know that Shapley value can be used only in a **convex** game but we didn't defined the game as convex. There are two theorems about this concept:

> (1) Every convex game has a nonempty core.
>
> We know that the core may be empty, but if it is not, is the Shapley value guaranteed to lie in the core? The answer in general is no, but the following theorem gives us a sufficient condition for this property to hold. We already know from this theorem that the core of convex games is nonempty. The following theorem further tells us that for such games the Shapley value belongs to that set.
>
> (2) In every convex game, the Shapley value is in the core.

So we are not able to define and be sure that this game is convex, we assume that and we declare that the Shapley value is in the core based on these theorems.

4 Application example

Let's suppose we have 2 people - Alice and Bob, and 3 cities - Milan(M), Rome(R) and Florence(F). Alice and Bob will declare a utility related to each city, like:

	M	R	F
Alice	5	4	3
Bob	5	4	2

Alice has 250 as maximum payment, Bob has 350. The costs related to cities are:

M	400
R	300
F	200
MR	500
MF	450
RF	370
MRF	700

Now we have to calculate v.
The empty group has v = 0, then
v({A}) = 200,
we look at utilities and we know where she prefers to go, then we look at the "wallet" and we know if she can afford it, in this case Florence it's the "least desirable" but is the only one; same reasoning for Bob.
v({B}) = 300 (he can afford to go to Rome, and he

prefers Rome to Florence)
v({A} ∪ {B}) = **?**

We calculate the value of the group using *Shapley value.* First, the most valid (with highest utility) tour is MR, because MRF = 700 $\geq$ 250+350, now we have to verify if there will be a fair division of costs.

Considering $\sum_{p=1}^{k} \binom{|N|}{p}$, every single valid coalition will be represented by a variable **G** so in this case we have:

$$\phi_i(G, v) = \frac{1}{G!} \sum_{S \subseteq G \setminus \{i\}} |S|!(G - |S| - 1)![v(S \cup \{i\}) - v(S)].$$

S({A}) = $\frac{500-300}{2} + \frac{200-0}{2} = 200$

we know that the other S should be 300 because budget balance, let's check:

S({B})= $\frac{500-200}{2} + \frac{300-0}{2} = 300$ Both are valid because S({A}) = 200 < 250 (Alice's "wallet") and
S({B}) = 300 < 350 (Bob's "wallet").

If these conditions were not met we had to calculate the next tour with the major utility. Of course from a computer science point of view this is not an ideal condition, being recursive, having to calculate everything every time and being able to fall into a very large case is expensive (computationally speaking) but it is still a concrete solution to solve this type of problem.