Types

In general, the ``type'' of an agent encapsulates all the private information that the agent has (so all the information s/he has that is not common knowledge). The type of an agent is sometimes simply the agent's knowledge of her private valuation function, and so $\theta_i = v_i$. The type of an agent can also include beliefs about other agents' utilities, and about their beliefs about her own utilities, and any other higher-order beliefs, etc. In this module, however, we are not bothering about such beliefs. Type space is the space of all possible/potential types the agent can have.

In some other scenarios there could be a distinction between the types and values. E.g. in a voting scenario between three options A, B and C, we might think of the type of an agent as her preference ordering (e.g. θ_i = A > B > C where > is read as ``is preferred to''), and then the valuation $v_i(x), x \in \{A, B, C\}$ denotes how much she values option x being picked (and to be consistent with the preference ordering it must be $that\ v_i(A) > v_i(B) > v_i(C)$). You can see that in the voting scenario the notions of type and valuation are not the same, though are related to each other. Having said that, one can also imagine a different modelling of the voting scenario where we assume that $\theta_i = v_i$ and so are not distinguishing between the two.

Actions

Let's imagine that you are an agent in a game. Your actions are what you can do in the game according to the rules of the game. E.g. in a single item auction (with IPVs and quasilinear utilities) your type is how much you value the item on sale. Your type space is any number greater or equal to zero (assuming you don't negatively value the item).

An auction (such as first-price or Vickrey) might only allow you to submit a single bid. In this case your action space is equal to your type space ($A_i = \theta_i$) and you are allowed to declare any of your potential types as your type. The auction mechanism is a direct one. Another auction might only allow you to answer Yes or No to multiple queries; e.g it might ask you ``do you value it more than 10 pounds?'' and you can answer Yes or No, and then it will ask you another question of Yes/No-answer type. In this auction, your action space is not the same as your type space and the auction mechanism is not direct.

Mechanisms vs social choice

A social choice function is not bothered with agents' incentives. A mechanism is consciously aware of the fact that agents may not tell the truth.

Consider a setting where we want to decide to whom we should allocate an old painting. A social choice function f in this setting receives agents' valuations and decides who gets the painting, e.g., the agent who submitted the highest valuation. A direct mechanism (A, M) may tell us more: it receives agents' valuations and decides who gets the painting, and in addition it may decide how much every agent should pay to the mechanism.

In a quasilinear mechanism (A, M) we have that $M = (\chi, p)$ where χ is the choice function and p is the payment function. When we design a quasilinear mechanism to implement a social choice function f in equilibrium, we may set $A = \Theta$ (so our mechanism is direct), or we may not. And we may set $\chi = f$, or we may not. Consider the single-item auction setting where f returns the agent with the highest declared valuation. Both first-price and Vickrey auctions set $A = \Theta$ and $\chi = f$.