Fair Allocation

Fair Division

Fair Division considers the problem of splitting up goods among two or more people such that everyone gets a 'fair' share according to their own taste. An illustrative example considers an inhomogeneous cake, which is to be divided among two people: the first cuts it in two pieces and the second chooses a piece. Both receive a piece that is at least half in their valuation. The problem has been studied in economics, social sciences, and mathematics.

Fair Allocation

Here we consider the *Fair Allocation* of a set of m indivisible items among n parties (e.g., furniture pieces after a divorce, an inheritance following a death, or cities and territories after an armistice). Our goal is to compute a fair division securely, such that every party inputs a sealed bid that must stay secret, using a simple allocation scheme. In general, defining a criterion that makes an allocation fair can be difficult and computing such an allocation is a complex optimization problem.

More precisely, denote the items by $\mathcal{I} = 1, 2, \ldots, m$. Every party $P_1, P_2, \ldots, P_n$ inputs a list $V_i = (v_{i1}, \ldots, v_{im})$ containing its valuation, where $\sum_{j=1}^{m} v_{ij} = B$ and v_{ij} denotes the preference of P_i for item j. The number B is fixed. An allocation $(A_1, \ldots, A_n)$ consists of n sets with $A_i \subseteq \mathcal{I}$ and gives items worth $\sum_{j \in A_i} v_{ij}$ to Pi. A maximal allocation achieves the highest total worth, summed over all parties.

Exercise

Implement an algorithm for finding the maximal allocation using `MPyC` that keeps all valuations secret. It should use exhaustive search, enumerate all allocations, and return some maximal allocation.