Introduction

This project studies a one-on-one bartering problem without payments using automated mechanism design (AMD). Agent 1 acts as the *mechanism designer* and commits in advance to a deterministic mechanism, while agent 2 strategically reports his preferences over goods. The objective is to compute a mechanism that is direct, truthful and individually rational for agent 2, while maximizing agent 1's expected utility. The implemented solution follows the branch-and-bound depth-first search algorithm proposed by Conitzer and Sandholm[1].

Model

Each agent initially owns N goods, for a total of $2N$ goods. An outcome is a partition of the goods between the agents. Thus, the outcome space is

$$O = \{0,1\}^{2N}, \quad |O| = 2^{2N}, \tag{1}$$

where a 1 indicates ownership by agent 1 and a 0 ownership by agent 2. The first N positions represent goods initially owned by agent 1, and the remaining N those initially owned by agent 2. For example, with $N = 2$, outcome $[1,1,0,0]$ corresponds to no trade, while $[0,0,1,0]$ indicates that agent 1 traded both of her goods for the first good of agent 2.

Agent 2 has a private type $\theta \in \Theta$, assigning a value in $\{1,2,3,4,5\}$ to each good. His utility function is defined as the value of the goods received in the outcome minus the value of his initial endowment. Agent 1's valuation function is defined analogously, based on her fixed preferences.

Constraints and Objective

The mechanism is *direct*: agent 2 reports a type θ, and the mechanism selects an outcome $M(\theta) \in O$.

It must satisfy:

- **Truthfulness**: reporting the true type is a dominant strategy for agent 2,

$$u(\theta, M(\theta)) \geq u(\theta, M(\hat{\theta})) \quad \forall \theta, \hat{\theta} \in \Theta. \tag{2}$$

- **Individual Rationality**: agent 2 never incurs a loss by participating,

$$u(\theta, M(\theta)) \geq 0 \quad \forall \theta \in \Theta. \tag{3}$$

Subject to these constraints, agent 1 maximizes her expected utility over the type distribution.

Outcome-Set-Based Mechanism in Branch-and-Bound DFS

Following Conitzer and Sandholm, mechanisms are represented by subsets of outcomes $X \subseteq O$, each inducing a truthful mechanism by assigning to each reported type the utility-maximizing outcome in X, with ties broken in favor of agent 1. Since agent 2 always receives their maximal-utility outcome in X, truthfulness holds for every subset.

The *branch-and-bound depth-first search* explores assignments of outcomes to included (X) or excluded (Y) sets, using an admissible upper bound function $v(X,Y)$ to prune suboptimal branches. Individual rationality is ensured by requiring that, for every type, at least one included outcome yields non-negative utility.

Implementation[2]

Several optimizations were applied in the implementation:

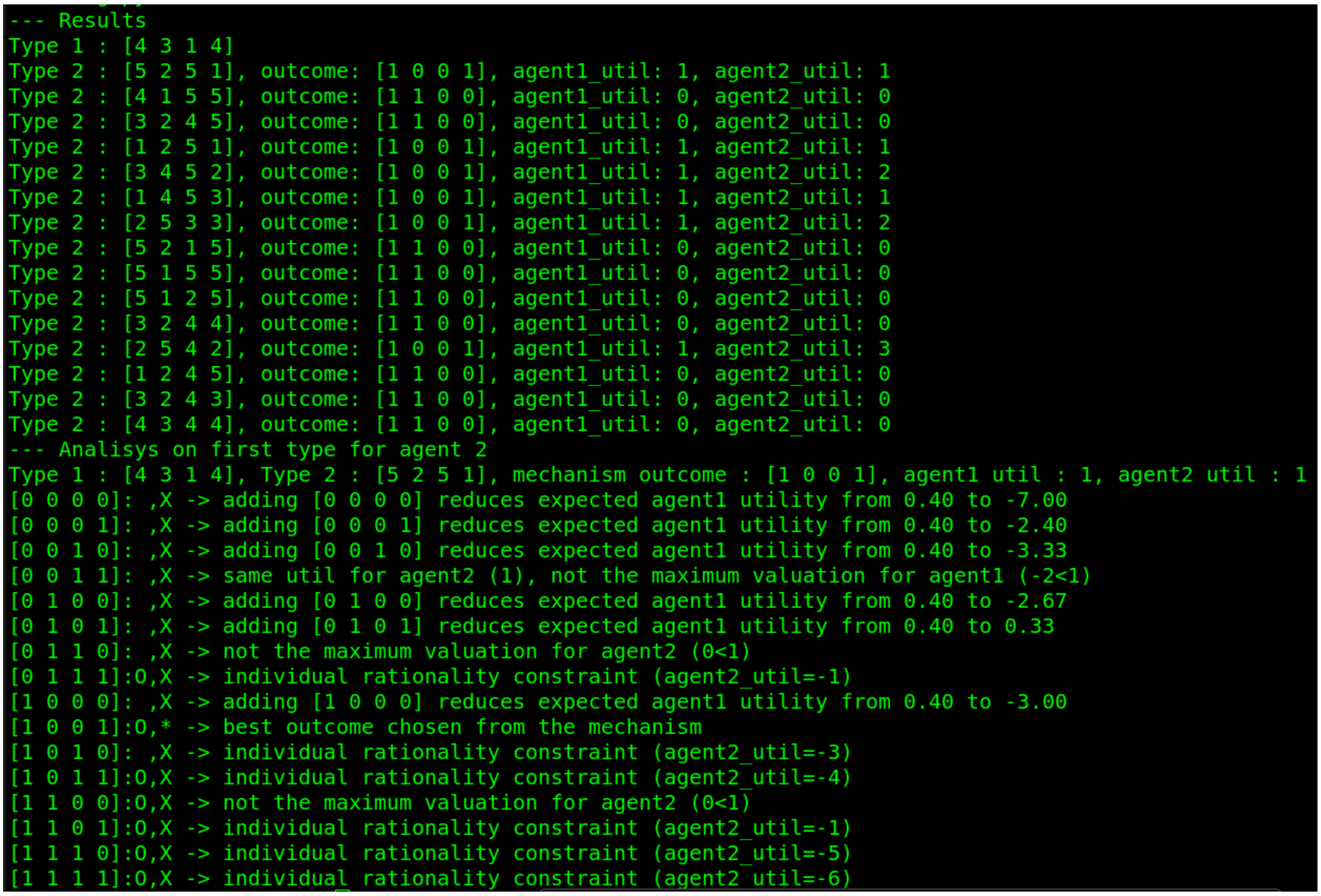
- Utilities for both agents were cached to avoid recomputation.
- The evaluation of $v(X,Y)$ compares candidate outcomes only against the best utility per type.
- Outcome exclusion was implemented using boolean masking.
- A finite subset of types randomly sampled with $|\Theta| = K$ was precomputed.

Due to the time complexity arising from enumerating all O subsets, experiments were limited to $N = 2$ goods per agent with $K = 15$ sampled types. In the worst cases, the algorithm considers up to 2^{2^4} subsets, whereas increasing to $N = 3$ requires exploring up to 2^{2^6} subsets, over 10^{14} times larger. Consequently, the algorithm did not terminate in reasonable time for $N = 3$.

Verification

For a fixed preference profile of agent 1 and a selected type of agent 2, all outcomes in O were analyzed[Fig 1]. Each outcome was reported in the form "outcome:status → reason", where the status indicates whether the outcome belongs to the set computed by the mechanism, is excluded (**X**), is the outcome selected by the mechanism (*****), or is an alternative yielding identical utilities for both agents ($\sim$).

For every excluded outcome, a specific explanation is provided, identifying violations of individual rationality, truthfulness, or dominance by another outcome that yields higher utility to either agents. This complete enumeration confirms that the chosen outcome is optimal and that all exclusions are consistent with the constraints.



```
--- Results
Type 1 : [4 3 1 4]
Type 2 : [5 2 5 1], outcome: [1 0 0 1], agent1_util: 1, agent2_util: 1
Type 2 : [4 1 5 5], outcome: [1 1 0 0], agent1_util: 0, agent2_util: 0
Type 2 : [3 2 4 5], outcome: [1 1 0 0], agent1_util: 0, agent2_util: 0
Type 2 : [1 2 5 1], outcome: [1 0 0 1], agent1_util: 1, agent2_util: 1
Type 2 : [3 4 5 2], outcome: [1 0 0 1], agent1_util: 1, agent2_util: 2
Type 2 : [1 4 5 3], outcome: [1 0 0 1], agent1_util: 1, agent2_util: 1
Type 2 : [2 5 3 3], outcome: [1 0 0 1], agent1_util: 1, agent2_util: 2
Type 2 : [5 2 1 5], outcome: [1 1 0 0], agent1_util: 0, agent2_util: 0
Type 2 : [5 1 5 5], outcome: [1 1 0 0], agent1_util: 0, agent2_util: 0
Type 2 : [5 1 2 5], outcome: [1 1 0 0], agent1_util: 0, agent2_util: 0
Type 2 : [3 2 4 4], outcome: [1 1 0 0], agent1_util: 0, agent2_util: 0
Type 2 : [2 5 4 2], outcome: [1 0 0 1], agent1_util: 1, agent2_util: 3
Type 2 : [1 2 4 5], outcome: [1 1 0 0], agent1_util: 0, agent2_util: 0
Type 2 : [3 2 4 3], outcome: [1 1 0 0], agent1_util: 0, agent2_util: 0
Type 2 : [4 3 4 4], outcome: [1 1 0 0], agent1_util: 0, agent2_util: 0
--- Analisys on first type for agent 2
Type 1 : [4 3 1 4], Type 2 : [5 2 5 1], mechanism outcome : [1 0 0 1], agent1 util : 1, agent2 util : 1
[0 0 0 0]: ,X -> adding [0 0 0 0] reduces expected agent1 utility from 0.40 to -7.00
[0 0 0 1]: ,X -> adding [0 0 0 1] reduces expected agent1 utility from 0.40 to -2.40
[0 0 1 0]: ,X -> adding [0 0 1 0] reduces expected agent1 utility from 0.40 to -3.33
[0 0 1 1]: ,X -> same util for agent2 (1), not the maximum valuation for agent1 (-2<1)
[0 1 0 0]: ,X -> adding [0 1 0 0] reduces expected agent1 utility from 0.40 to -2.67
[0 1 0 1]: ,X -> adding [0 1 0 1] reduces expected agent1 utility from 0.40 to 0.33
[0 1 1 0]: ,X -> not the maximum valuation for agent2 (0<1)
[0 1 1 1]:O,X -> individual rationality constraint (agent2_util=-1)
[1 0 0 0]: ,X -> adding [1 0 0 0] reduces expected agent1 utility from 0.40 to -3.00
[1 0 0 1]:O,* -> best outcome chosen from the mechanism
[1 0 1 0]: ,X -> individual rationality constraint (agent2_util=-3)
[1 0 1 1]:O,X -> individual rationality constraint (agent2_util=-4)
[1 1 0 0]:O,X -> not the maximum valuation for agent2 (0<1)
[1 1 0 1]:O,X -> individual rationality constraint (agent2_util=-1)
[1 1 1 0]:O,X -> individual rationality constraint (agent2_util=-5)
[1 1 1 1]:O,X -> individual rationality constraint (agent2_util=-6)
```

Figure 1: Example execution report

References

1. An Algorithm for Automatically Designing Deterministic Mechanisms without Payments
2. GitHub project repository