# MatchingAgent

Chris Mascioli and Amy Greenwald
Brown University

May 3, 2023

# 1  Introduction

`MatchingAgent` is an agent designed to take full advantage of the concurrency in the SCML OneShot world. The rules changes made from last year's competition made the aspirational strategies of the old agents (including `PatientAgent`) obsolete. It also uses some basic opponent detection to allow it to better respond to the agents the environment provides (`SingleAgreementAgent`, `GreedyOneShotSyncAgent`, and `GreedyOneShotAgent`).

## 1.1  World Information

`MatchingAgent` focuses on getting as close to matching its exogenous contracts as it can on each day, because of how utility is calculated in the SCML world. The day's utility $u$ is a function of an ordered pair $(q, p)$, where $q$ is the quantity from a contract and $p$ is the price. In general, $u(0,0) \ll u(q_{\text{exog}}, p_{\text{worst}}) < u(q_{\text{exog}}, p_{\text{best}})$, so the main thing `MatchingAgent` does each day is try its best to quantity match the exogenous contracts it receives in order to minimize the loss of utility that comes along with each unit it fails to secure. In the SCML world, the agents vastly prefer to trade at the worst price than to fail to come to any agreement.

# 2  MatchingAgent

`MatchingAgent` has two separate strategies: one for when it is on the side producing more items and one when it is on the side where demand is higher than supply.

**Proposal Strategy**  `MatchingAgents`'s primary objective is to match the exogenous quantity. `MatchingAgent` has a pretty simple proposal strategy

1. It splits how much it still needs equally among all it's trading partners. If there's extra, it assigns the extra to the agent's with the lowest balances

2. If the agent is on the side with more supply than demand it offers the best price for itself. If it's on the other side it offers the worst price for itself.

**Response Strategy**  `MatchingAgent` has a simple response strategy. It waits to receive an offer from all opponent's and then finds the bunde that best matches its quantity. If it's on the side where supply is more than demand it accepts this bundle no matter what. If it's on the other side, it accepts the bundle only if all the trades in the bubdle give it the best price.