AgentNeko

Ryota Arakawa*

1 Negotiation Strategy

AgentNeko uses these parameters to make a decision. The target price p_a^{target} represents the unit price, which AgentNeko offers to the agent a at the final step of the negotiation. The reservation price factor r is used to calculate the reservation price $p^{reserve}$ as Eq.(1). At first, p_a^{target} and r are initialized by Eqs.(2) and (3).

α	the disposal cost
β	the shortfall penalty
q^{need}	the quantity of items needed to achieve the exogenous contract
p^{trade}	the trading price of the item which AgentNeko trades with the partner agents
p_a^{target}	the target price of the partner agent a
r	the reservation price factor

$$p^{reserve} = \begin{cases} rp^{trade} & \text{(if it is a seller)} \\ (2-r)p^{trade} & \text{(if it is a buyer)} \end{cases} \tag{1}$$

$$p_a^{target} = p^{trade} \tag{2}$$

$$r = \begin{cases} 0.95 - \alpha/5 & \text{(if it is a seller)} \\ 0.95 - \beta/5 & \text{(if it is a buyer)} \end{cases} \tag{3}$$

The followings show the summary of the AgentNeko's strategy.

- AgentNeko makes an offer mainly based on p_a^{target} and q^{need}.
- AgentNeko updates its target prices depending on the outcome of the negotiations to make more profit or to avoid the risk of making no contract.
- AgentNeko doesn't use a utility function, and the decision is based on the quantity and the unit price of the offer.

* Tokyo University of Agriculture and Technology, arakawa@katfuji.lab.tuat.ac.jp

1.1 Offering Strategy

At the negotiation step s, AgentNeko offers $p_{s,a}^{offer}$ and $q_{s,a}^{offer}$ to the partner agent a. Here, S is the limit of the negotiation step, p^{max} is the maximum unit price of the issue, and p^{min} is the minimum unit price of the issue. Note that s is 0-based.

$p_{s,a}^{offer}$ is calculated by Eqs.(4), (5) and (6). Since we can know which agent proposes first this year, We can know the last action our agent can take in the negotiation. If the last action of the agent is to respond, the offer it makes at the $(S-1)$th step will not reach the partner agent. So, we introduce s' to guarantee that AgentNeko can make an offer whose unit price is equal to p_a^{target}.

$$s' = \begin{cases} s & \text{(if it can make an offer as the last action)} \\ s+1 & \text{(otherwise)} \end{cases} \tag{4}$$

$$c_s = 1 - \left(\frac{S - s' - 1}{S - 1} \right)^{0.5} \tag{5}$$

$$p_{s,a}^{offer} = \begin{cases} p^{max} - (p^{max} - p_a^{target})c_s & \text{(if it is a seller)} \\ p^{min} + (p_a^{target} - p^{min})c_s & \text{(if it is a buyer)} \end{cases} \tag{6}$$

$q_{s,a}^{offer}$ is calculated by Eq.(7). Here, $q_a^{received}$ is the quantity of the last offer received from the agent a. AgentNeko considers $q_a^{received}$ to make an agreement easier.

$$q_{s,a}^{offer} = \min\left(q^{need}, q_a^{received}\right) \tag{7}$$

1.2 Responding Strategy

AgentNeko makes a response in the following procedure. Here, $p_a^{received}$ and $q_a^{received}$ represent the unit price and the quantity of the last offer received from the agent a.

1. If AgentNeko has already secured the quantity of items needed to achieve the exogenous contract, it responds with `END_NEGOTIATION`.
2. Otherwise, if $q_a^{received} > q^{need}$, it responds with `REJECT_OFFER`.
3. Otherwise, if $p^{received}$ is worse than $p_{s,a}^{offer}$ which it will offer at the next step, it responds with `REJECT_OFFER`.
4. It responds with `ACCEPT_OFFER`.

1.3 Updating of the Target Price

At the end of each day, AgentNeko updates its target prices depending on the outcome of the negotiations.

1. It introduces a boolean variable `no_contract_sensitive`. This variable is true if any of the following conditions are met.
   - AgentNeko is a seller and $\alpha > \beta$.
   - AgentNeko is a buyer and $\beta > \alpha$.
2. If any of the following conditions are met, AgentNeko updates p_a^{target} to make concessional offers (See section 1.3.1).
   - `no_contract_sensitive` is true and AgentNeko failed to make a contract on this day.
   - `no_contract_sensitive` is false and AgentNeko failed to make a contract for two consecutive days.
   - `no_contract_sensitive` is true and AgentNeko only made contracts whose sum of quantities is less than half the quantity of the exogenous contract for two consecutive days.
   - `no_contract_sensitive` is false and AgentNeko only made contracts whose sum of quantities is less than half the quantity of the exogenous contract for three consecutive days.
3. Otherwise, if AgentNeko made two or more contracts in this day, it updates p_a^{target} to make bullish offers (See section 1.3.2).
4. For each partner agent a, AgentNeko updates p_a^{target} with $p^{reserve}$ if needed (See section 1.3.3).

The variable `no_contract_sensitive` is introduced to change the behavior depending on the amount of the penalty when the agent makes no contract. If it is false, AgentNeko attempts to maintain the target prices to make a better contract.

1.3.1 Weakening the Target Prices

If AgentNeko failed to make a contract or only made contracts with low quantity, it will be too aggressive. Therefore, it weakens the target prices in the following procedure.

1. Let X be the list of the partner agents. X is sorted in descending order of the target price p_a^{target} for each agent a if AgentNeko is a seller, or in ascending order if it is a buyer.
2. Select each agent a from half of the top of the list X and do the following procedure.
   (a) If AgentNeko is a seller, update p_a^{target} with $0.95p_a^{target}$.
   (b) If AgentNeko is a buyer, update p_a^{target} with $1.05p_a^{target}$.
   (c) If $p^{reserve}$ is better than p_a^{target}, update r with $r - 0.05$.

1.3.2 Strengthening the Target Prices

If AgentNeko made n contracts, it can make bullish offers to $n - 1$ partners, since it can still make a contract with the remaining one agent if the other agents refuse to sign. Therefore, it strengthens the target prices in the following procedures.

1. Let X be the list of the partner agents which made contracts with AgentNeko.

2. It samples $n-1$ agents randomly from X, and works the following procedure on each agent a.
   (a) If AgentNeko is a seller, update p_a^{target} with $1.05p_a^{target}$.
   (b) If AgentNeko is a buyer, update p_a^{target} with $0.95p_a^{target}$.

1.3.3 Updating with the Reservation Price

p_a^{target} is updated with the formula (8). This procedure is introduced for AgentNeko to follow the market trend, and not to make a too much concessional offer.

$$p_a^{target} \leftarrow \begin{cases} \max\left(p_a^{target}, p^{reserve}\right) & \text{(if it is a seller)} \\ \min\left(p_a^{target}, p^{reserve}\right) & \text{(if it is a buyer)} \end{cases} \tag{8}$$

2 Evaluation

We tested AgentNeko against BetterAgent and AdaptiveAgent by running 100 simulations. Table 1 shows the result. The result shows that AgentNeko outperforms other agents.

Table 1: The test result

agent	mean	min	Q1	median	Q2	max
AgentNeko	1.050500	0.656962	0.995478	1.040101	1.095167	1.741313
BetterAgent	0.878098	0.479779	0.817417	0.889387	0.961870	1.114469
AdaptiveAgent	1.033641	0.806740	0.972998	1.024516	1.074866	1.703941