DoSomething: Try and Failure in Making Dynamic Utility Function

Masanori HIRANO, The University of Tokyo
hirano@g.ecc.u-tokyo.ac.jp

July 14, 2020

Abstract

As my agent's strategy, I employed a new dynamic utility function. But, it was too challenging to perform well in the league. I also found some issues in my agent and confirm my agent doesn't performed well.

1 Introduction

In my agent, I employed a new method to deal with negotiations using dynamic utility function based on the opposite negotiation of my factory. However, unfortunately, this method is not working well due to algorithmic failure and incompatible with negotiation flow. I think my agent cannot win this league, but, in this report, I will explain what I intended as the agent's strategy and what was the problem of my agent.

2 The Design of MyAgent

2.1 Negotiation Choices & Utility Function & Concurrent Negotiation

In my agent, I employed "dynamic utility function" based on opposite negotiations. Each negotiation offer is evaluated by what percentage of opposite ongoing negotiation can be covered.

In the following, I refer "negotiations" as the sequence of negotiation and "offer" as each shot of offering to negotiation partners. Moreover, "offer" is noted by o and its quantity, delivery time and unit price p_o is noted by (q_o, t_o, p_o). On the other hand, "negotiation" has a possible range of negotiation which is consist of offers. This is noted by $n_i = ([q_{min_{n_i}}, q_{max_{n_i}}], [t_{min_{n_i}}, t_{max_{n_i}}], [p_{min_{n_i}}, p_{max_{n_i}}])$. For example, $(5, 4, 10)$ means an offer whose quantity, delivery time, and unit price are 5, 4, 10 respectively. And, $([1, 5], [4, 6], [10, 12])$ means a negotiation whose possible quantity, delivery time, and unit price are from 1 to 5, from 4 to 6, from 10 to 12, respectively. Although this range was usually defined by

CFP before starting negotiations, which is published by some factories, here, I employed an estimated possible range which is estimated by negotiation history. This estimation is held based on rejection for my past proposing offer. For example, when the current possible negotiation range is $([1,5],[20,25],[10,20])$ and my last proposing sell offer was $(4,24,19)$ and rejected by opponents, the possible range will be changed to $([1,4],[20,24],[10,19])$. This estimation is based on two assumptions. One is the opponent's utility for my proposing offer will be decreasing according to negotiation steps because the earlier proposing of my factory is usually good only for my factory. The other assumption is the opponents' utility function is monotonicity.

Moreover, in the following, "the opposite ongoing negotiations" means ongoing buying negotiations if the offer which is begin evaluated is sell offer, or ongoing selling negotiations if the offer is buy offer.

My dynamic utility function for offers is calculate by the current contracts/agreements and the ongoing opposite negotiations. At first, comparing selling and buying contracts/agreements, my factory calculate these differences on each step. Then, my factory manager calculate how percentages of the difference and the ongoing opposite negotiations will be covered by the offer. If the offer will cover the differential contracts/agreements, the number of these covered contracts/agreements are counted. Then, the percentages of coverage of each opposite ongoing negotiation will be also added to the number, and the number will be divided by the total number of contracts/agreements and the ongoing opposite negotiations and turned into the final evaluation percentage. The coverage will be calculated by what percentage of the possible range of one opposite ongoing negotiation is compatible with the offer. Here, the possible range employed only the delivery time and price, and compatibility is defined by whether the delivery time of selling contracts/agreements will be possible and the factory doesn't lose money (the profits is not less than 0). If the quantity of an offer is more than 1, the percentage will be calculated for possible negotiations. That is, if the quantity is N, the contracts/agreements and opposite ongoing negotiations whose quantity is equal or more than N will be targetted and the final evaluation percentage is calculated by the sum of the final evaluation percentage for $1, 2, \cdots, N$ th quantities. This value is the utility for the offer.

Then, as the next step, my factory manager calculate the possible best utility among all possible offer.

Using the best utility, the relative utility scaled to $[0,1]$ is calculated. In addition, the threshold of relative utility is also calculated by $(1-t)*(0.9-0.3)+0.3$ where t means the relative negotiation step. Then, using this value, my factory manager decide offering and responding. It offers the worst offer whose utility is better than the threshold, and accepts opponents' offer if the utility is better than the threshold.

3 Evaluation

I haven't evaluated enough but the results were almost the same or less than the built-in decentralized agent. But sometimes our agents win the agent.

4 Lessons and Suggestions

I think there are there reason of my agents didn't work well. One is an algorithmic failure. For the processing for multiple numbers of quantity is inappropriate. Usually, the more quantity is better. However, sometimes this algorithm is not. If the utility for the second material of buy offers is not good, my agents decide not to buy.

The second reason is compatibility for NegMas Negotiation flow. The architecture of my dynamic utility function is made for concurrent negotiation. Of course, the negotiation in negmas is partially concurrent. However, the concurrent negotiation is limited to the negotiations whose CFP is published by my agents. My dynamic utility function is assumed to work well under the condition that all negotiations proceed on the same negotiation time step. This problem was found after my coding. So, I couldn't avoid and fix this problem.

The third problem is the calculation limit. In my dynamic utility function, a big amount of calculation is necessary for calculate for all possible range of offers and its opposite ongoing negotiations. In my submitted code, I employed sampling for the range, but it also affects the performance of searching the best offering and strategy.

Conclusions

As my agent's strategy, I employed a new dynamic utility function. But, it was too challenging to perform well in the league. I also found some issues in my agent and confirm my agent doesn't performed well. I thank the organizers for their efforts and would like to work better next year.