Delft University of Technology

Artificial Intelligence Techniques

IN4010-12

Assignment: Negotiation

Authors:
Yiting Li(5281873)
Jiaxuan Zhang(5258162)
Yi Yao(5250161)
Yu shen(5249260)

1 BackGround

1.1 Background of Report

Our report and implementation focus on a very popular shopping method, the group buying buyer coalition. In China, one of the most important internet group buying platform *Pinduoduo* ranks among the top Internet companies. Different buyers have different preference and a commodity category may have different styles. For seller, they will offer a discount when the sales volume of certain goods reaches a certain amount, regardless of the number of buyers.

1.1.1 Negotiation Domain

In this part, the representation of issues and values in negotiation domain are given. The meaning of weights of issues, utility of values and buyer's power are explained.

An agent represents a buyer. The representation of issues and values is as follows

1. Issues: One issue represents a category of goods, e.g. issue 1 for lamb, issue 2 for camera, etc.
2. Values of issues: Each value of a issue represents a buying choice among this issue. More specifically, values of a issue is a set of different styles of this products category, they vary in buyers' behaviour and goods' prices, e.g. for category "camera" (issue 2 in the formal example), there are 4 values: {"0"(means not buying),"1"(portable camera),"2"(means normal SLR camera),"3"(means professional SLR camera)}. We set "0", which is changeable, as buying nothing.

Here, the meaning of weights of issues and the meaning of utility of values are clarified:

1. Weights of Issues: Weights of issues, which vary due to different demand of different buyers, represents the importance of a single product category for a buyer. For example, buyer 1 prefer camera than lamb while buyer 2 is badly in need of a lamb, as a result, the weights of camera is higher than lamb for buyer1, for buyer 2, the weight is reversed.
2. Utility of Values: Utility of values for one issue represents buyers' preference among different items in one category, e.g. buyer who wants to decorate home prefers a elaborated lamp while buyer who wants home brighter will only consider lamp's brightness;

By using above explanations, our model can describe reality buyer demand, for example:

1. example 1: Buyer 1 doesn't need normal sunglasses but this one is a limited edition, so issue sunglasses has low weights but this style has a high utility still yields a fair utility contribution.
2. example 2: Buyer 1 starve for a lamb but the prices of different styles of lambs is too high, so issue lambs with high weights but each value has low utility, it still yields a fair contribution.
3. example 3: Buyer 1 really need product A but the current style are all too expensive. So, this issue will have a high weight but only the not buying choice will have a acceptable utility.

The meaning of power of different is clarified below. For customer needing badly for some goods and having a sufficient budgets, they're possible to pay extra money to buy a product above a discounted price. For example, buyer 1 may pay 100$ for a camera whose discounted price is 80(120$ as original). 20$ subsides buyer 2 and 3 who can pay 70$. Buyer 1 spends extra money but saves money. Customer like buyer 1 in this situation is set to have high power.

Based on the Domain Model, the main purpose for each agent is to find a optimal consensus solution which means buyer agree on this solution can get what they want at a discounted price.

1.1.2 Problem Translation

Compared to other protocols, MoPAC protocol helps the problem finds a partial optimal solution and by Opt-In procedure, a buyer can better reject other's bid if the buyer finds other agent may have similar preference.

1. Bids phase: Each agent give their choices in each products category (not buying or select a style).
2. Voting Phase: Each agent calculates the expected utility he(she) will get following another agents' buying plan. Compared to the reservation utility, each agent will vote to others' bids: reject, accept unconditionally or conditionally accept (popularity).

3. Opt-In Phase: For an agent who infer that other buyers have similar preference according to the result of Voting Phase, the agent may reduce the possibility to arrive a partial consensus result by high up his(her) C_{min} and vice verse.

4. Continuation or Termination: This process will continue and come to an end when a partial consensus result is achieved or time is out. Notice that after generates a partial consensus result, another consensus result will not be generated. Because most times, seller needs to guarantee the income, and the number of remaining buyers is not enough for the merchant to continue to offer discounts.

2 Agent Generation

2.1 Party Architecture

The Party architecture of Main Java document is as follows:

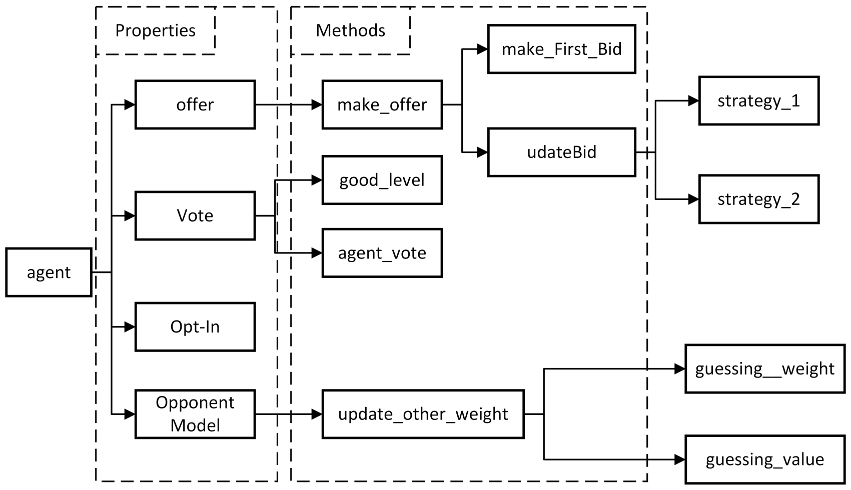

Figure 1: Party Architecture

2.2 Offer Strategy

Our agent have three kinds of Offer Strategy, they are used in different period of the negotiation. So we classified them int three catalog according to the situation to use them: First offer, Early offer and Later Offer.

2.2.1 First Offer

For the first time of making offer, there is totally no other's preference can be taken into consideration. So for the first offer, our agent use the most greedy one. Because the utility space is linear additive, for each issue, our agent review all values and choose the best one greedily.

2.2.2 Early Offer

In the early stage of the negotiation, it is hard to learn other agents preference. So we try to randomly change issue that are not important to our agent to try to get the consent of others (as shows in algorithm 1. At the same time, our agents collect other agents' behaviour to analysis their preference.

2.2.3 Later Offer

In the later stage, we have got some information about the preferences of different agents. So we can use heuristic methods to make our bid be accepted easier. We use the guessed value table we have established before to calculate the preference distribution of different agents. Then we choose the value with the maximum weighted expectation for each issue(as shows in algorithm 2).

2.3 Voting Strategy

We develop a dynamic voting strategy, it can adjust the C_{min} and C_{max} based on other agents' power and their bids. We set 5 satisfy level and judge each bid by the *goodlevel* function. for each level, we set C_{min} and C_{max}

Algorithm 1 Early Offer Strategy

```
Input: The weight list of issues of agents: $Weight\_List$; the list of issues $Issue\_List$
Output: a new bids: Bid
1: Select one of the issues with the smallest weight of the first fifty percent. $issue\_name$
2: for $issue$ in $Issue\_List$ do
3:     if $issue \neq issue\_name$ then
4:         find the value of highest utility $value$
5:     else
6:         random change the value of $issue$
7:     end if
8:     add [$issues$,$value$] into bid
9: end for
```

Algorithm 2 later Offer Strategy

```
Input: The weight list of issues of agents: $Weight\_List$; the list of issues $Issue\_List$; the list of agents
    $Agent\_List$;the list of values $Value\_List$; the list of guessed utility for different values $Guessed\_Value$
Output: a new bids: Bid
1: for $issue$ in $Issue\_List$ do
2:     max ← 0
3:     for $value$ in $Value\_List$ do
4:         sum ← 0
5:         for $agent$ in $Agent\_List$ do
6:             sum += $Guessed\_Value[agent,issue,value] \times Weight\_List[agent,issue]$
7:         end for
8:         if $sum > max$ then
9:             max ← sum ; $currentValue$ ← value
10:        end if
11:    end for
12:    add [$issues$,$currentValue$] into bid
13: end for
```

based on other's power. A better level means the bid more acceptable, so the difference between the C_{min} and C_{max} is larger. While for a bid with lower level, the difference will be less.

2.4 Opt-In Strategy

In the Opt-In phase, out agent use strategy to avoid making too much sacrifice for other agents. If the information of the previous voting stage indicates that bids that are only slightly better than the reservation utility are likely to reach an agreement, the possibility of accepting these bids will be reduced by increasing c_{min}.

2.5 Opponent Model

2.5.1 Issue Weights of Opponents

In order to infer issue weights of other agents, we seek idea from practical negotiation process. In a daily negotiation or discussion process. If a person insists on a certain proposition, then this proposition may be very important to him. Therefore, we estimate the importance of different issues to an agent by counting its number of changes on each issue and transfer it to a normalize weight as shown in algorithm 3. More changes means less important(weights).

Algorithm 3 Update Weight

```
Input: the recording of changes $change\_record$
Output: a guessing weight list of another agent $weight\_list$
1: for i=0; i ≤ length of $change\_record$; i++ do
2:     $weight\_list$[i]=1/($change\_record$[i]+1);
3: end for
4: $weight\_list$ ←Normalize($weight\_list$)
```

2.5.2 Issue Values of Opponents

In a daily negotiation or discussion process. If a value is important for an agent, the agent may always try to propose this value for this issue. To obtain the utility of different values for a specific agent, we set up a table to count the occurrence of different values in all bids raised by this agent. A value is regarded as a better choice if it appears more times in this agent's previous bids. As the values in count table will grow larger and larger, we use the count value divided by the total round number as the guessed utility for these values.

2.6 Concession Strategy

Initially, we set the reservation utility as 0.7, and after each round this value is multiplied by 0.95 until it is close to 0.3. It is suitable to have a low reservation utility as the round growing, and it will not lead to a apparently decrease of agent's utility. Because the voting result of our agent is based on the good level of a bid, not simply depends on the relationship between the revenue and the numerical value of the reservation utility.

3 Result and Improvements

3.1 Result

Figure 2 shows the result our party. We design a profile which always has a low utility to simulate the real situation. This guy just want to look at some goods and don't want to buy anything. So, you can see the grey line is never higher than 0.2. For other agents, they all can reach a partial agreement at some points. However, we find that the bids in later rounds are fixed which may cause a deadlock in negotiation. This is caused by our heuristic method. Our guessed value will approximate to a constant value as we have enough data. Then, our algorithm will always generate the same bid in every round.

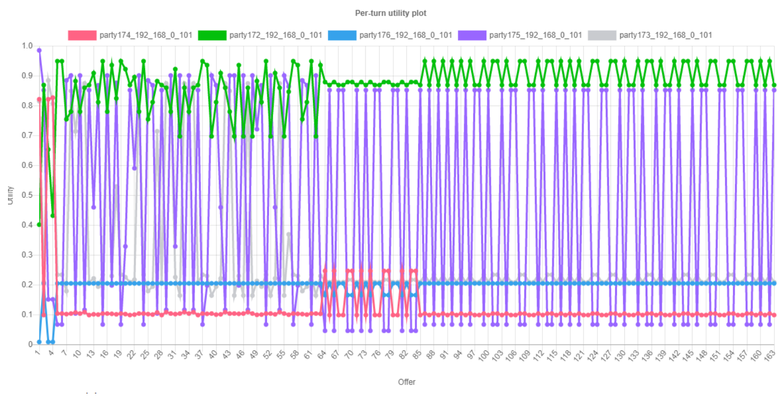

Figure 2: Result of our party with 5 agents

To solve this problem, we got some inspiration from the process of simulated annealing algorithm, which try to jump out the local optimal solution by use time-variant threshold and random next point. We introduce some random variables into our algorithm to get rid of the locally optimal solution. We set a threshold to ignore the weight of current agent. If a random value from 0 to 1 is larger than the weight of an issue, than the weight of current agent in this issue will be ignored. Figure 3 is the result of our improved algorithm. Now, the behaviors in later stages are more flexible and agents can reach some points which may have greater utility than before.

Figure 3 shows how our parties negotiate with two random parties in a different domain. As we choose a dynamic voting strategy, our party can ensure a higher utility as the random party can only ensure a 0.6 utility. As shown in figure 3, the green line and pink line are random parties, they reach a consensus at the start point because their utilities are bigger than 0.6. Our parties are still trying to find a better bid although their are some points that their utilities are also bigger than 0.6. At last, they reach a consensus and utility 0.77 and 0.64. So our parties have a better strategy to balance the utility of different agents.

Then we test the performance of our party by running it in a domain for multiple times and calculate the weighted average utility of every agent. Because the weighted average utility is a good measure to test the value achieved by this negotiation. More specifically, we set the maximum round to 100 and choose 4 profiles with "power:1". We compare our result with the result of random party to show the performance increase achieved by our strategies. As shown in table 1, in the "buyer" domain, the agreed party number and average utility of

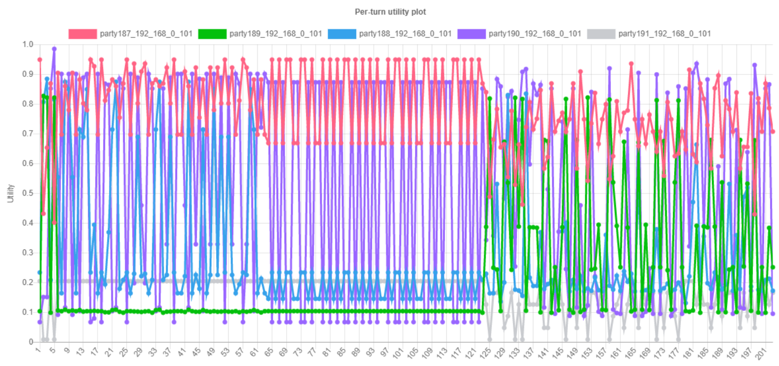

Figure 3: Result of improved party with 5 agents

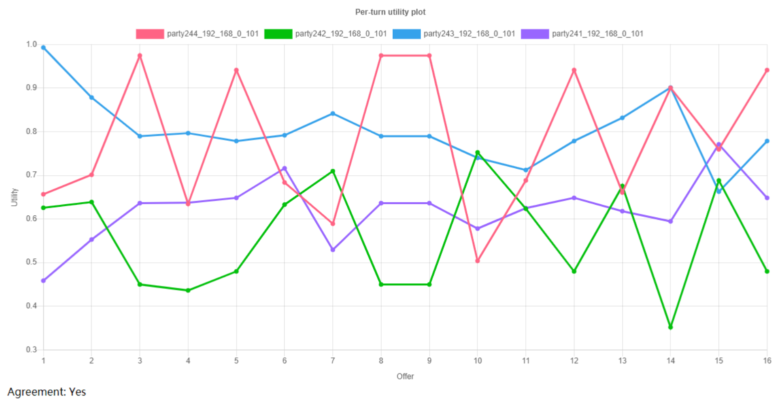

Figure 4: Negotiation of different parties

our party are higher than the random party which means our party can discover the potential utility in a better way. In the "party" domain which is not the domain we design our party for, our performance also is better which shows the flexibility of our party.

Party name	agree number in "party"	average utility in "party"	agree number in "buyer"	average utility in "buyer"
Our party	4	0.85	4	0.83
Random	3.6	0.61	3	0.56

Table 1: performance of our party and random party

3.2 Pros & Cons

3.2.1 Pros

1. Inspired by real life and human behavior, our organization not only considers its own interests, but also guesses the situation of its opponents, and seeks alliances to ensure its own interests.

2. Inspired by real life and human behavior, our organization not only considers its own interests, but also guesses the situation of its opponents, and seeks alliances to ensure its own interests.

3.2.2 Cons

1. The method of modeling opponents is relatively basic, which may have errors and may be misled by the intentional behavior of opponents.

2. When catering to the opponent, only the most influential opponent is considered, and no attempt is made to cater to everyone else