Argumentation-based Negotiation for choosing a car engine

Charles Boy de la Tour - Zakariae El Asri

I. Introduction

A company has a number of engine options (Diesel Engine, Electric Engine, Steam Engine, Natural Gas Engine) from which to choose for its new car. Each option has different benefits and drawbacks in terms of consumption, environmental impact, cost, durability, noise. To decide on the best engine for their new car, the company wants to simulate a negotiation process among agents with different opinions and preferences. This will allow them to make an informed decision based on all the criteria that are important to them. We decided to implement and test this environment based on two scenarii. In both of them, 2 agents with different order of criterion importance (preferences criterion) but the same value for each criterion for each object are negotiating together. In the first scenario, the preferences criteria are defined randomly; in the second scenario 2 agents with defined preferences are negotiating together.

II. Negotiation Model

In order to achieve the tasks described in the introduction, we decided to start from the algorithm proposed in the exercise, making some changes and adding more rules to ensure a robust process.

The idea of our negotiation process consists in the fact that at each step we will have an agent that attacks to eliminate an item, and the other agent that defends the same item. Thus, we can distinguish two major strategies:

A defense strategy: consists in evoking the criteria on which the item is well evaluated (average, good, very good), following the order of the agent's preferences.

An attack strategy: consists in evoking the criteria on which the object is less good (bad, very bad), always adopting the order of the agent's preferences.

1. Description

Our negotiation protocol is described in the pseudocode below (3 parts) . At the beginning (Appendix: Algorithm1 - part1), the starting agent proposes his most preferred item, the adversaire accepts it if it is among its top 10% of items, else he will ask for arguments motivating this proposal.

The attack strategy (described in Appendix: Algorithm1 - part 2 of the model) is a bit complicated. We have configured an agent with an offensive tendency. When it receives a proposal, he first tries to eliminate it by proposing another object, this is possible if the agent has an item in his database that is better rated than the item proposed on the criterion put forward by the opponent.

Then, if he is unable to propose a better alternative, he will check if the object is badly rated for him on the announced criterion, in this case he simply refuses the object without proposing an

alternative. (this option will be avoided in our experiment since items have the same values for all agents)

If neither of the two previous conditions is verified, the agent will search among the low-rated criteria if there is a more important criterion than the one announced by the opponent. In this case, the agent attacks the product by mentioning this more important criterion.

If the agent does not manage to attack the object, it will accept it

The defense strategy (described in the pseudocode - part3) is simpler, the defending-agent takes the criterion challenged by the attacker. He simply looks at the evaluation table of the object to find a criterion with a good value and which is more important than the attacked criterion. If he does not find a criterion verifying this condition, he will try to propose another object if he has any left, otherwise he will accept one of the objects previously proposed by the opponent.

2. Our contribution:

We first changed the case 1 in the attacking strategy, instead of Argue with a new item, we thought it's better to decompose it on Two messages: Propose followed by argue (like if the other was asked Why).

If we don't send the argue and we wait for the adversary to ask why, we run the risk of answering with a criterion other than the one on which we based our proposal, in which case we lose the logical flow of the argumentation

Then, we have put some locks to avoid loops. Thus, the same argument is not reproduced for a given item. Also the agent cannot propose an item already proposed by himself or by the opponent.

At the end, when an agent is blocked and can't neither argue nor propose, he is forced to accept an item. Instead of taking the last proposed item, we decided to iterate over the items already proposed by the opponent and take the best one. The other agent obviously agrees, since he is the one who proposed the item, and we judged that this is more appropriate than taking the last proposed item

3. Check of correctness

To check the correctness of our implementation before running the experiment, we set several manual exemples tuned to perform each specific transition.

The example below with 2 agents and 3 items regroup the majority of transitions:
Agent_0 Preference = Consumption, Cost, Durability, noise, Environment
Agent_1 Preference = noise, Environment, Durability, Consumption, Cost
The table below summarize the scores for each agent (Score_ag0, Score_ag1)

	Cost	Consumption	Durability	Environment	Noise
Diesel	(3,2)	(1,2)	(4,2)	(4,1)	(4,2)
Electrical	(4,4)	(1,2)	(1,3)	(2,2)	(0,3)
Steam	(1,1)	(1,1)	(1,1)	(4,3)	(1,1)

```
================================================================================
From agent_0 to agent_1 (PROPOSE)       Diesel Engine (A super cool diesel engine)
From agent_1 to agent_0 (ASK_WHY)       WHY Diesel Engine  ?
From agent_0 to agent_1 (ARGUE)         Diesel Engine           <=      PRODUCTION_COST = GOOD,
From agent_1 to agent_0 (PROPOSE)       Electric Engine (A very quiet engine)
From agent_1 to agent_0 (ARGUE)         Electric Engine         <=      PRODUCTION_COST = VERY_GOOD,
From agent_0 to agent_1 (ARGUE)         not Electric Engine     <=      CONSUMPTION = BAD,  CONSUMPTION > PRODUCTION_COST,
From agent_1 to agent_0 (ARGUE)         Electric Engine         <=      NOISE = GOOD,  NOISE > CONSUMPTION,
From agent_0 to agent_1 (ARGUE)         not Electric Engine     <=      DURABILITY = BAD,  DURABILITY > NOISE,
From agent_1 to agent_0 (ARGUE)         Electric Engine         <=      ENVIRONMENT_IMPACT = AVERAGE,  ENVIRONMENT_IMPACT > DURABILITY,
From agent_0 to agent_1 (PROPOSE)       Steam Engine (A steam-punk engine)
From agent_0 to agent_1 (ARGUE)         Steam Engine            <=      ENVIRONMENT_IMPACT = VERY_GOOD,
From agent_1 to agent_0 (ARGUE)         not Steam Engine        <=      NOISE = BAD,  NOISE > ENVIRONMENT_IMPACT,
From agent_0 to agent_1 (ACCEPT)        Electric Engine
From agent_1 to agent_0 (COMMIT)        Electric Engine
From agent_0 to agent_1 (COMMIT)        Electric Engine
```

III. Results and Analysis

For the state of the experiment, we have decided to create 4 items: Diesel Engine, Electric Engine, Steam Engine, Natural Gas Engine. Their scores for each of the criteria are presented in the table below. The higher the score the better the object for the criteria considered. The sum of scores for each engine is equal to 12.

	Cost of production	Consumption	Durability	Environment impact	Noise
Diesel Engine	3	3	4	1	1
Electric Engine	0	4	2	2	4
Steam Engine	1	3	3	3	2
Natural Gas Engine	1	2	4	3	2
Total	5	12	13	9	9

A. Random criterion Preferences: Which item is the best ?

Following the simulation discussed in part II, the model for 1000 negotiations with random criterion preference order for the agents gives the results presented in Figure 1.These results show that the Electric Engine and the Diesel Engine are more often chosen at the end of a negotiation, while the Steam Engine and Natural Gas engine appear to be less chosen. If we try to understand why those engines were chosen, we can explore the different arguments that were used by the winners during each negotiation. The results are presented in Figure 2, the distribution is more evenly distributed compared to the chosen engine distribution but we can see that production cost is the argument which is the least used. The results seem to be linked to the total of scores for each of the criteria which is highlighted in the table of scores.

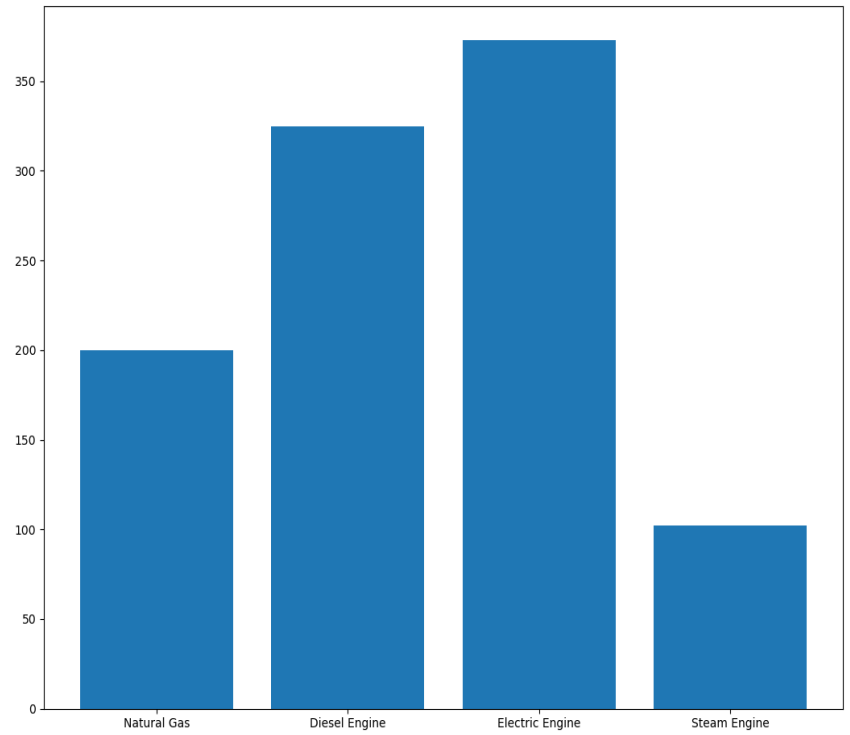

Figure 1: Frequency of chosen engine for 1000 negotiations between random agents

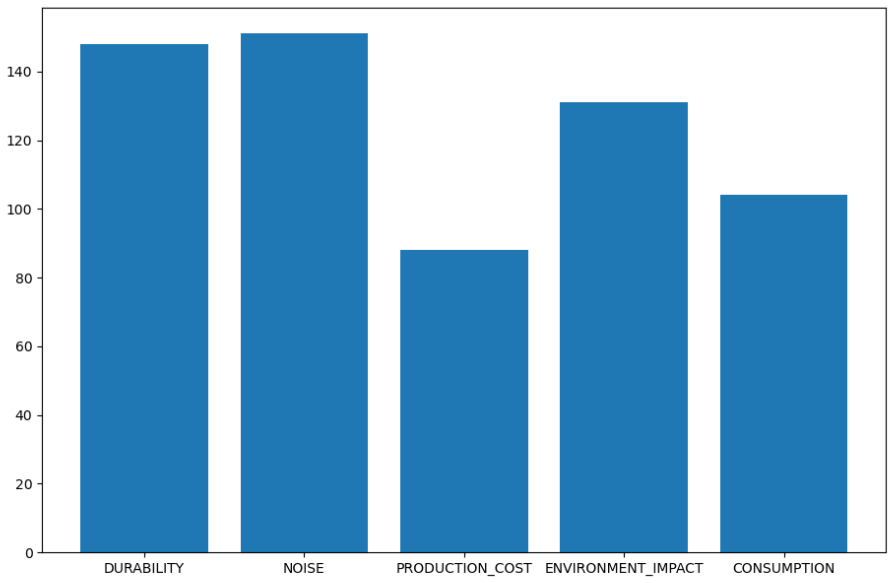

Figure 2: Frequency of used criterion in argument by the winner for 1000 negotiations between random agents

Hence, the best item seems to be the Electric Engine due to its high score for its noise compared to the other engines.

B. 2 agents with defined preferences

For the second approach, we set the preferences of 2 agents according to the following table, ranked from 0 (more important) to 4 (least important):

	Agent 1	Agent 2
Cost of production	0	3
Consumption	3	4
Durability	1	0
Environment impact	2	1
Noise	4	2

2 situations are then possible, either the Agent 1 or the Agent 2 start the negotiation.

The 2 processes can be found in Figure 3 and Figure 4.

```
From agent_0 to agent_1 (PROPOSE)       Diesel Engine (A super cool diesel engine)
From agent_1 to agent_0 (ASK_WHY)       WHY Diesel Engine  ?
From agent_0 to agent_1 (ARGUE)         Diesel Engine           <=      PRODUCTION_COST = GOOD,
From agent_1 to agent_0 (ARGUE)         not Diesel Engine       <=      ENVIRONMENT_IMPACT = BAD,  ENVIRONMENT_IMPACT > PRODUCTION_COST,
From agent_0 to agent_1 (PROPOSE)       Steam Engine (A steam-punk engine)
From agent_1 to agent_0 (ACCEPT)        Steam Engine
From agent_0 to agent_1 (COMMIT)        Steam Engine
From agent_1 to agent_0 (COMMIT)        Steam Engine
```

Figure 3: Negotiation starting with agent 1

```
From agent_0 to agent_1 (PROPOSE)       Steam Engine (A steam-punk engine)
From agent_1 to agent_0 (ASK_WHY)       WHY Steam Engine  ?
From agent_0 to agent_1 (ARGUE)         Steam Engine            <=      ENVIRONMENT_IMPACT = GOOD,
From agent_1 to agent_0 (ARGUE)         not Steam Engine        <=      PRODUCTION_COST = BAD,  PRODUCTION_COST > ENVIRONMENT_IMPACT,
From agent_0 to agent_1 (PROPOSE)       Natural Gas (A compressed natural gas engine)
From agent_1 to agent_0 (ASK_WHY)       WHY Natural Gas  ?
From agent_0 to agent_1 (ARGUE)         Natural Gas             <=      ENVIRONMENT_IMPACT = GOOD,
From agent_1 to agent_0 (ARGUE)         not Natural Gas <=      PRODUCTION_COST = BAD,  PRODUCTION_COST > ENVIRONMENT_IMPACT,
From agent_0 to agent_1 (PROPOSE)       Electric Engine (A very quiet engine)
From agent_1 to agent_0 (ASK_WHY)       WHY Electric Engine  ?
From agent_0 to agent_1 (ARGUE)         Electric Engine         <=      NOISE = VERY_GOOD,
From agent_1 to agent_0 (ARGUE)         not Electric Engine     <=      PRODUCTION_COST = VERY_BAD,  PRODUCTION_COST > NOISE,
From agent_0 to agent_1 (PROPOSE)       Diesel Engine (A super cool diesel engine)
From agent_1 to agent_0 (ACCEPT)        Diesel Engine
From agent_0 to agent_1 (COMMIT)        Diesel Engine
From agent_1 to agent_0 (COMMIT)        Diesel Engine
```

Figure 4: Negotiation starting with agent 2

In these experiments, we can notice that the winning agent, i.e, the agent with the highest engine score, is the agent who is not starting the negotiation. Indeed, the responding agent manages to return the arguments of the starting agent ending in choosing the best engines in its opinion.

IV. Appendix

Algorithm 1 Negociation Protocol / part 1

Initialisation
  $-$ *create a set of n Agents*
  $-$ *setup a list of m items* $\{O_1, ..., O_m\}$
  $-$ *setup a list of k criteria* $\{C_1, ..., C_k\}$
for *each* (A, B) *in Agents* ... **do**
    $O_i = best\_object(A)$
    A *to* B, *propose* (O_i)
    B : $receives - message(propose(O_i))$
    if $O_i \in$ *top* 10% *favorite items of* B **then**
        B *to* A, $Accept(O_i)$
    else
        B *to* A, $Ask\_Why(O_i)$
    end if
    $A : receives - message(Ask\_why(O_i))$
    if A *has a supporting argument for* O_i **then**
        A *to* B, $Argue(O_i,\ Arguement)$
    else if A *has not proposed items* **then**
        A *to* B, *Propose* (O_j), $O_j \neq O_i$
    else
        A *accept* O_k *from already recieved items*
    end if

Algorithm 1 Negociation Protocol / part 2 - Attacking

$B : receives - message(Argue(O_i,\ Arguement))$
if *B has an attacking argument on* O_i **then**
    $Switch\ (ifArgument\ (C_i = x)\ do$
        $Case\ (B\ has\ better\ O_j\ on\ C_i)\ do$
            $B\ to\ A,\ Propose\ (O_j)$
            $B\ to\ A,\ Argue\ (O_j\ ,\ C_i = y)$
        end
        $Case\ (B\ has\ bad\ evaluation\ on\ C_i\ for\ O_i)\ do$
            $B\ to\ A,\ Argue(notO_i, C_i = y)$
        end
        $Case\ (B\ has\ bad\ evaluation\ on\ C_j\ for\ O_i\ and\ C_j \geq C_i)\ do$
            $B\ to\ A,\ argue(notO_i, C_j = y, C_j \geq C_i))$
        end
    end
    $Switch\ (ifArgument\ (C_i = x\ ,\ C_i \geq C_k)\ do$
        $Case\ (B\ has\ better\ O_j\ on\ C_i)\ do$
            $B\ to\ A,\ propose\ (O_j)$
            $B\ to\ A,\ argue\ (O_j\ ,\ C_i = y)$
        end
        $Case\ (B\ has\ bad\ evaluation\ on\ C_j\ for\ O_i\ \&\ C_j\ \geq C_j)\ do$
            $B\ to\ A,\ argue(notO_i, C_j = y, C_j \geq C_i))$
        end
    end
else
    $B\ to\ A,\ Accept\ (O_i)$
    $A\ to\ B,\ Commit\ (O_i)$
    $B\ to\ A,\ Commit\ (O_i)$
end if

Algorithm 1 Negociation Protocol / part 3 - Defending

$A : receives - message(Argue(not\ O_i,\ Arguement))$
if *A has a supporting argument on* O_i **then**
        $A\ to\ B,\ argue(O_i, C_j = y, C_j \geq C_i))$
else if *A has not proposed items* **then**
    $A\ to\ B,\ Propose\ (O_j),\ O_j \neq O_i$
else
    $A\ accept\ O_k\ from\ already\ recieved\ items$
    $A\ to\ B,\ Accept\ (O_k)$
    $B\ to\ A,\ Commit\ (O_k)$
    $A\ to\ B,\ Commit\ (O_k)$
end if