AI: Negotiation Homework

Michael Paris (mlp5ab) and Vince Ning (vcn8fn)

Our Agents

We created a total of 7 negotiating agents thoughout this assignment. Each negotiator has its own negotiation strategy and level of aggression that dictate how willing one is to accept an offer and how aggressive their counter offer will be. We created an external framework file called driver.py that automatically runs a particular agent against all other agents 10 times in a row for each test scenario. Using this driver.py framework we were able to make more data-driven decisions as to which strategies work in a particular scenario. The 7 negotiation agents we created are called the **LinearNegotiator, AsymptoticNegotiator, LinearThenAsymptotic, FlexibleNegotiator, MeanNegotiator, PseudoRandomNegotiator, and BANegotiator**.

Asymptotic Negotiator

We think this is our best negotiator and is the one we would like to be submitted for grading. It is located in testers.py.

This negotiator is a combination of aggressive and forgiving. Its acceptance policy is driven by the function 1/x where x is the number of past iterations. The policy results in a relaxed acceptance plan where the agent is more willing to accept worse offers later in the negotiation. The policy begins with an upper threshold of 90% of our total possible utility we could possibly receive and asymptotically converges towards 0. However, we do not simply let our negotiator to accept anything that gives us any inkling of utility at the end since other negotiators might exploit that by sending their preferences during the last iteration. Thus, we set the lower bound to be 10% of our total possible utility that we could possibly receive.

The negotiator also has a safety mechanism for dealing with particularly aggressive opponents. If an opponent is being overly-aggressive, which we define as sending us a 75% majority of offers resulting in a negative utility to our bot, we attempt to throw them a bone because our bot is not interested in those negative offers. In this case we do a few things to try to find an offer that is beneficial to both parties. 1) We look back at their past offers and pinpoint the one which gave them the most utility. We treat this offer as an approximation for their preference list giving our bot the ability to check what utility they might get from our offer. 2) We then begin shuffling offers and calculating both the utility we get from that offer and the utility we expect them to get from that offer. 3) If the difference between those utilities is within 30% of one another we send that fair offer and hope the other agent will accept it.

This ability to estimate how much utility our opponent would receive from offers really enabled a lot of behavior defining this bot. For example, in situations where the opponent is being aggressive and offering us negative utility items, we are able to use this estimation to create

counter offers that put us on the same playing ground. We are also able to use this mechanism on the other side and are able to accept offers where we expect their utility to be less than our own.

The bot has another safety mechanism aimed at exploiting weak agents on the last turn of a negotiation. If over multiple rounds of negotiations between two bots we find that the other bot accepts on the last turn every time, we will send our preferences on the last turn. We do this because if there is a bot that accepts by default on the last turn to avoid getting negative points on a failed negotiation, we will win maximum points.

The last thing to note about this negotiator is that in the default case it also has a relaxing policy for sending new offers. We realized that preserving the early elements from our preference list earlier in offers results in more points than the later elements of the preference list and we created a policy to take advantage of this. Essentially as the bot relaxes it will begin to shuffle more and more elements over time from right to left, where the initial offer begins transparently as just our true preference list. Over time you might see a series of offers from our default policy that looks like this:

```
    ['snowshoe', 'ski', 'snowboard', 'trolley', 'wheelchair', 'plane',
'motorbike', 'computer', 'car', 'tv', 'bike']
    ['snowshoe', 'ski', 'snowboard', 'trolley', 'wheelchair', 'plane',
'motorbike', 'computer', 'car', 'bike', 'tv']
    ['snowshoe', 'ski', 'snowboard', 'trolley', 'wheelchair', 'plane',
'motorbike', 'computer', 'car', 'tv', 'bike']
    ['snowshoe', 'ski', 'snowboard', 'trolley', 'wheelchair', 'plane',
'motorbike', 'car', 'tv', 'computer', 'bike']
    ['snowshoe', 'ski', 'snowboard', 'trolley', 'wheelchair', 'plane', 'tv',
'car', 'computer', 'bike', 'motorbike']
    ['snowshoe', 'ski', 'snowboard', 'trolley', 'wheelchair', 'bike',
'motorbike', 'plane', 'car', 'computer', 'tv']
    ['snowshoe', 'ski', 'snowboard', 'trolley', 'plane', 'wheelchair',
'computer', 'car', 'motorbike', 'bike', 'tv']
    ['snowshoe', 'ski', 'snowboard', 'motorbike', 'tv', 'computer', 'bike',
'plane', 'trolley', 'wheelchair', 'car']
    ['snowshoe', 'ski', 'car', 'bike', 'plane', 'snowboard', 'wheelchair',
'trolley', 'computer', 'motorbike', 'tv']
    ['snowshoe', 'ski', 'snowboard', 'tv', 'plane', 'bike', 'motorbike', 'car',
'trolley', 'wheelchair', 'computer']
```

As you can see, the policy relaxes from right to left and results in a steady decrease in the utility we gain from an offer over time. This policy is aggressive but fair as we are becoming more forgiving with how much utility we are willing to receive over time.

Linear Negotiator

A negotiator with a linearly decreasing threshold over time when it comes to accepting offers. This bot was created mainly to test the sensitivity of aggression. In other words, was this decreasing linear acceptance policy from start to finish considered aggressive or not? After testing it against the other bots that we made, we found that many times, due to aggression from other bots, we were not accepting as many offers. The counter-offer mechanism in place was fairly aggressive, mirroring the default relaxing counter-offer policy that our Asymptotic Negotiator had. This created a vicious cycle since the counter-offer mechanism in place was fairly aggressive as well, leading to a highly-negative aggregate number of penalty points from frequent failed negotiations.

LinearThenAsymptoticNegotiator

This negotiator is a middle man between the Linear and Asymptotic negotiator. In the first half of the total iterations in a negotiation it relaxes its acceptence criteria at a linear rate, and during the second half it relaxes with the function 1/x. However, when we tested this bot against the others that we made, we found that it was still had too aggressive of an acceptance policy, and didn't lead to many acceptances. This strongly led to us believe that we needed to follow an even more lenient offer-acceptance policy, which is why Asymptotic Negotiator worked best since it favors earlier acceptances at lower standards of utility.

FlexibleNegotiator

For this bot, we tried to make its acceptance criteria flexible, as the name provides. This means that it basically alternates its strictness for accepting an offer depending on how fast or slowly the opponent is giving in. The Flexible Negotiator takes its opponent's most recent offers and tracks the trend of the offers by how much perceived utility the opponent is receiving. Then, the Flexible Negotiator mimics its opponent's aggression by applying the same increase or decrease in the opponent's offers and treats that as its own criteria to accept or decline an offer. Thus, if the opponent is playing nice, we will play nice, and if the opponent is being mean, we will return the aggression as well. However, the problem with this bot mainly is that it does not ever make the first move, and merely follows the opponent's actions and trends in offers, making it hard to ever win handily on our terms.

MeanNegotiator

The mean negotiator is a real jerk and wont budge on his preferences. There's really not much more to say about this bot except that it does not consider the opponent's historical or current offers. It merely returns its own preferences as proposed offers until the opponent either accepts or continues declining until both bots receive penalties. This approach is essentially a toss-up. If the general pool of opponents concede to us, we receive boat loads of points; however, in the case that our opponents are even slightly aggressive and do not give us exactly what we want, then the Mean Negotiator will accumulative an exorbitant amount of penalty points.

PseudoRandomNegotiator

This negotiator creates new offers by swapping two elements at random each time. This is why it's called the Psuedo-random Negotiator since it does not completely shuffle our preferences to make an offer. This provides a level of conservativeness in making offers since we randomly adjust just two items each iteration, potentially creating some level lenient trend for the opponent to detect and potentially accept. Randomly adjusting just two items in each iteration prevents our bot from giving up too much utility each successive iteration as well. In the end, we never expected this bot to be our best bot, and we mainly used it for testing purposes.

BANegotiator

This negotiator was our first attempt to conquer the world. We had big dreams here that ended up not really working like we had hoped. Essentially we were trying to make some sort of sense out of the other bots' trends of past offers. To do this we calculated a best fit line and an R^2 value to judge the quality of this fit. If the fit was strong enough to show some sort of correlation to a trend we would act accordingly. If we saw a negative trend we took an aggressive acceptance policy because we figured it would be better to let the trend take its course so the other bot would have to concede more points before we accepted. If we saw a flat or positive trend, we knew the other bot was being aggressive towards us and thus we would need to either try to make a fair offer or be aggressive in return. After running this bot through some tests we created we found that the correlation data from the trend line was often not reliable with low R^2 values and the insights we wanted to see were not as visible as we hoped.

Some Data

tc5.csv

This is an example showing my bot performing against itself in a scenario with perfectly opposite preference lists. To accomodate for this scenario, the bot has learned to accept offers where both parties involved are expected to be getting the same or relatively similar amount of utility from the offer. This was a change that allowed parties with very different preferences to get along. At first we had a tough time handling this scenario as our bots were too aggressive. This is an interesting data set because it shows how many of the offers were accepted on the first turn in a situation with perfectly opposite preferences. This can happen when the list that our bot is using as its best approximation for the other users preferences is incorrect.

```
['plane', 'motorbike', 'computer', 'car', 'tv', 'bike'] None
['plane', 'motorbike', 'computer', 'car', 'tv', 'bike'] ['bike', 'tv', 'car',
'computer', 'motorbike', 'plane']
['plane', 'motorbike', 'computer', 'car', 'tv', 'bike'] ['bike', 'tv', 'car',
'computer', 'motorbike', 'plane']
```

```
['plane', 'motorbike', 'computer', 'car', 'bike', 'tv'] ['bike', 'tv', 'car',
'computer', 'plane', 'motorbike']
['plane', 'motorbike', 'computer', 'car', 'bike', 'tv'] ['bike', 'tv', 'car',
'motorbike', 'computer', 'plane']
Successful negotiation:
    Negotiator A: 12.7
    Negotiator B: 2.7
['plane', 'motorbike', 'computer', 'car', 'tv', 'bike'] None
Successful negotiation:
    Negotiator A: 14.7
    Negotiator B: 2.7
['plane', 'motorbike', 'computer', 'car', 'tv', 'bike'] None
Successful negotiation:
    Negotiator A: 14.7
    Negotiator B: 2.7
['plane', 'motorbike', 'computer', 'car', 'tv', 'bike'] None
Successful negotiation:
    Negotiator A: 14.7
    Negotiator B: 2.7
['plane', 'motorbike', 'computer', 'car', 'tv', 'bike'] None
['plane', 'motorbike', 'computer', 'car', 'tv', 'bike'] ['bike', 'motorbike',
'computer', 'car', 'tv', 'plane']
Successful negotiation:
    Negotiator A: 14.7
    Negotiator B: 4.7
['plane', 'motorbike', 'computer', 'car', 'tv', 'bike'] None
Successful negotiation:
    Negotiator A: 14.7
    Negotiator B: 2.7
['plane', 'motorbike', 'computer', 'car', 'tv', 'bike'] None
Successful negotiation:
    Negotiator A: 14.7
    Negotiator B: 2.7
['plane', 'motorbike', 'computer', 'car', 'tv', 'bike'] None
Successful negotiation:
    Negotiator A: 14.7
    Negotiator B: 2.7
['plane', 'motorbike', 'computer', 'car', 'tv', 'bike'] None
Successful negotiation:
    Negotiator A: 14.7
    Negotiator B: 2.7
['plane', 'motorbike', 'computer', 'car', 'tv', 'bike'] None
Successful negotiation:
    Negotiator A: 14.7
    Negotiator B: 4.7
Final result:
    Negotiator A: 145.0
```

```
    Negotiator B: 30.999999999999996
```

tc3.csv

A run of our negotiator against all other negotiators in the pool. Our aggressive but fair strategy leads us to victory but, in a scenario with overlap in preferences, rewards the other player with a few points as well. Our agent beats all our test agents except for the MeanNegotiator. This happens because the Mean Negotiator refuses to accept offers and only offers their preference list thus forcing our bot to accept their preference list when it also satisfies our point threshold. In a situation where our preferences may be somewhat closely aligned with the other agent, the results will be more collaborative than violent. Our bot focuses on scoring points more than hurting the other players and thus works well with other bots.

```
Summary for AsymptoticNegotiator vs. LinearThenAsymptoticNegotiator
    Scenario: tc3.csv
        A
            Wins: 8
            Score: 270.18650793650795
        B
            Wins: 0
            Score: 226.18650793650795

Summary for AsymptoticNegotiator vs. BANegotiator
    Scenario: tc3.csv
        A
            Wins: 10
            Score: 236.18650793650795
        B
            Wins: 0
            Score: 158.18650793650795

Summary for AsymptoticNegotiator vs. MeanNegotiator
    Scenario: tc3.csv
        A
            Wins: 0
            Score: 288.18650793650795
        B
            Wins: 10
            Score: 332.18650793650795

Summary for AsymptoticNegotiator vs. AsymptoticNegotiator
    Scenario: tc3.csv
        A
            Wins: 10
            Score: 256.18650793650795
```

```
        B
            Wins: 0
            Score: 222.18650793650795

Summary for AsymptoticNegotiator vs. LinearNegotiator
    Scenario: tc3.csv
        A
            Wins: 8
            Score: 256.18650793650795
        B
            Wins: 1
            Score: 236.18650793650795

Summary for AsymptoticNegotiator vs. PseudoRandomNegotiator
    Scenario: tc3.csv
        A
            Wins: 7
            Score: 10.87460317460318
        B
            Wins: 0
            Score: -57.12539682539682

Summary for AsymptoticNegotiator vs. Negotiator
    Scenario: tc3.csv
        A
            Wins: 10
            Score: 75.53055555555557
        B
            Wins: 0
            Score: -62.46944444444444

Summary for AsymptoticNegotiator vs. FlexibleNegotiator
    Scenario: tc3.csv
        A
            Wins: 8
            Score: 286.18650793650795
        B
            Wins: 2
            Score: 246.18650793650795

Total Points for A: 1677
Total Points for B: 1301
```

Closing Thoughts

This assignment was largely an exercise in discovery through trial and error. Early on in the process, we were throwing many ideas around and discussing potential situations where they may fit well and others where they may not. The best fit line idea we had early on in the process ended up yielding fewer fruits than we had hoped and we were only able to discover this after building it and testing it against several scenarios. In the end we wanted a bot that, by default, would be "relaxingly aggressive" in that it would start with our preferences and then make offers yielding gradually-declining utility. This would give us a good basis to beat weak opponents who might be eager to accept offers. We had to make some changes to handle the case when we were playing aggressive opponents in scenarios with vastly different preference lists. This resulted in our scheme to try to guess opponents preferences. In the end, the Asymptotic Negotiator fit the bill as it embodied many of the qualities that worked well in basic negotiations, while still handling many of the particularly one-sided and tricky bots as well.