Super Agent

Yaniv Zimmer
zimmer.yaniv@gmail.com

Ben Nageris
Bennageris@gmail.com

Inbar Hefer
inbarchef@gmail.com

June 2022

Abstract

Our goal is building a negotiation agent, to compete ANL 2022 (Automated Negotiation League). The was build in Python, with compatibility to geniusweb python library and environment.

In the following article we will describe our efforts to build an agent who wins both the highest average utility and social welfare criterions

Our code can be found here: https://github.com/YanivZimmer/Agent-Nego

1 Introduction

The ANC 2022 is a competition aims to measure performance of autonomous agents, in the environment of genius-web framework, using 2 criteria:

1.1 Max Personal Utility

Get highest rewards from all the bids we participate in

1.2 Max Social Welfare

Get highest shared reward for us and our from all the bids we participate in
This two goal are pursuit simultaneously after, therefore we have built an agent we believe to achieve both of these goals.

1.3 Introduction

In the world of autonomous negation, a smart agent should balance between playing hard to get with his opponent, and loosing as little bids as possible. In order to do so we divided out biding strategy into learning phase (where we try to learn to which utilities our opponent is willing to accept), regular bidding (where we try to offer an optimal bid for us, regrading bid our opponent is willing to accept) and near end bidding (when we are close to deadline, sometime offer more generous deal in order not to lose the deal).

2 Bidding Strategy

In each step of the negotiation session, we are required to offer a bid (by protocol) therefore bidding strategy is crucial! For each bid, the opponent needs to approve or make a counteroffer. In the negotiation initialization, we calculate a sorted list to save the bids list. The list is in descending order and sorted by the utility of the bid. We save the best offer the opponent gave us and of course approve if it's our optimal offer. As we described in the learning phase, in the beginning, we mainly learn the opponent's preferences and measure his bids to understand his utility threshold and valuable items. And after static time delta from start, we split the time into timeslots where we save the offers for each timeslot and calculate its average. Those phases we mainly split into two categories: 1. From the start of the phase to the very last turns 2. Very last turns to the end The strategy changes in those cases. In the first case, we calculate the index of the first bid below the threshold (list in descending order) and return a randomized good offer with the best utility for our opponent (in order to increase social welfare). In the second case, we are very closed to the negotiation end. We assume that most agents will surrender and will accept our deal therefore until 97 percent of the time we offer them the best offer they offered us. From 97 percent to 99percent we offer the first offer with utility above average. Eventually, from 99 percent to 100 percent of the time we offer a randomized good offer.

2.1 Near deadline bidding

When we are close to deadline, we have less benefit from discarding average offers. Yet we don't want our opponents to learn we are easy close to the deadline and exploit it. We used Friendly mode as following: Accept offers which we don't see as extortion (above 0.5 utility) if its out turn to accept. Offer deal with opponent utility much higher than his threshold, as long as it is not lower than 0.5 utility for us.

We have installed 2 threshold to determine when to use friendly mode: 1. Very Final Time Constant- we defined that after 0.998 of the negotiation time has passed, we try our "friendly" mode is certain probability 2.Probability for Friendly action- even when all other condition are fulfilled, we make friendly action in 0.5 prob. We do it so our opponent won't learn to keep try pushing us close to our time limit

3 Opponent Model

Our opponent model is frequency-based. We assume that the opponent offers for himself the items that give the higher utility. Therefore, we created a frequency map for each negotiation session to try to predict the opponent's most valuable items. In addition, we used the standard deviation to predict the issue's weights. The formula to estimate the opponent utility:

$s_i = \frac{std(v_i)}{\sum_k s\ t\ d(v_k)+\varepsilon}$

${v_j}^{s_i} = \frac{f({v_j}^{s_i})}{max_k f({v^{s_i}}_k)}$

To understand the opponent's behavior, we split the negotiation into 2 phases. The first phase is the first phase we mainly try to understand the opponent's behavior by saving his bids. In the second phase, we sliced the time into timeslots where we save for each timeslot the opponent bids to understand his decrease of bids utility to reach agreements. We assume that the opponent will reduce his threshold as time goes on to gain some utility. At the end of each negotiation, we save the average utility for each timeslot to offer bids around its threshold but with a higher gain for our agent. The data is saved for each of the agent independently so we can determine the right utility for each opponent character.

4 Acceptence Strategy

Each turn one side is offering the other side a bid. We accept the opponent's bid in the negotiation session if the utility is above a certain threshold. The threshold is calculated for each opponent independently and changes over time. For each negotiation session, we randomize another utility function (one out of five) to make it more difficult to learn our policy and to mislead the opponent's thoughts when negotiating with us. All functions are based on last year's winner functions. To describe all the functions, we better start with definitions. $\bar{V}_t$ represents the average utility until time t and $\bar{V}_o$ is the average utility against opponent o.

$V_{min} = 0.55 * \bar{V} + 0.4 * \bar{V}_o + 0.5 * std\left(v\right)^2$

$\Delta Val = Vmax - Vmin$

Where V_{max} is a hyperparameter and set to 0.95. As we already mentioned we randomize the utility threshold function for each negotiation session out of a predefined list. All formulas have the same baseline:

$T\left(t\right) = Val_{max} - \Delta Val \frac{(e^{\alpha} * t-1)}{(e^{\alpha}-1)} * (func(t, \alpha))$

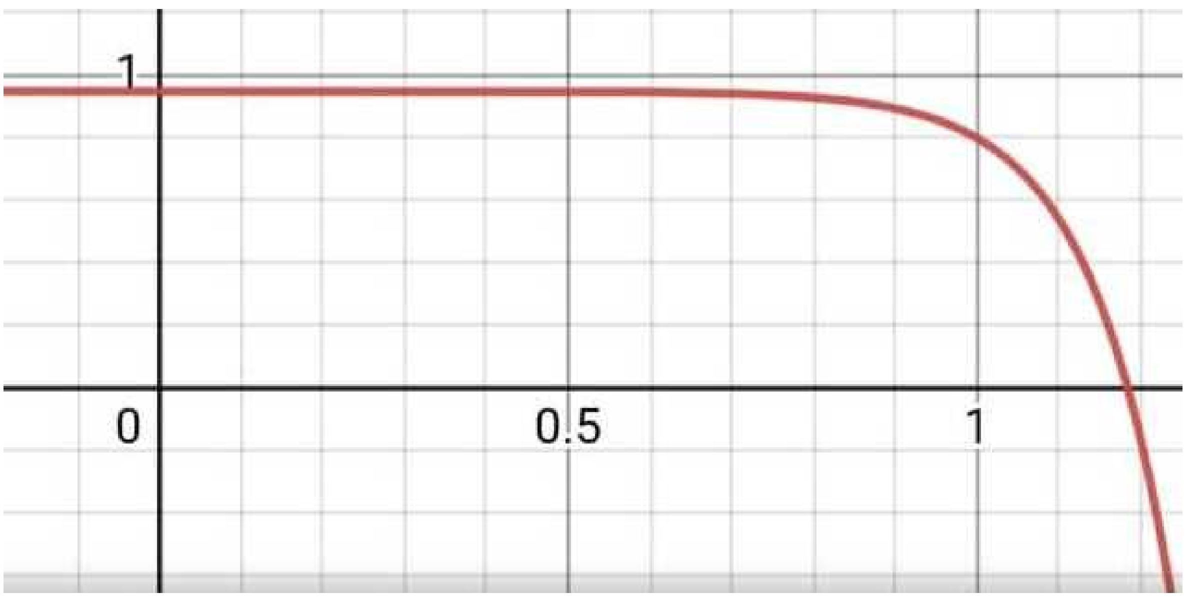
In order to confuse the opponent $func\left(t, \alpha\right)$ is randomized out of a predefined list. The key parameter is alpha which is the time of compromising, the higher the value of alpha the later the time of compromising. The value of *alpha* is approximated and updated after each negotiation session. We used the same *alpha* approximation like last year's winner. After calculating alpha and selecting the utility threshold function for each bid given by the opponent, we first calculate the utility threshold and then check if the bid's utility is above this threshold. If the bid's utility is above the calculated threshold we agree to this bid.

5 Experiments

We run our agent in a tournament, agent given agents, with the following results:

	avg_utility	avg_nash_	avg_social	avg_num_	count	agreemen	failed	ERROR		
SuperAger	0.760205	0.437389	1.292774	2810.227	44	40	4	0		1.489229
Agent26	0.749949	0.449074	1.333294	1853.068	44	41	3	0		1.456033
Agent32	0.729023	0.45723	1.389421	542.1136	44	43	1	0		1.431556
Agent25	0.706084	0.31516	1.031024	2732.386	44	32	12	0		1.400829
Agent24	0.702533	0.437667	1.350939	2072.795	44	43	1	0		1.366831
Agent3	0.694745	0.500405	1.457853	2623.386	44	44	0	0		1.344698
Agent2	0.664299	0.457944	1.40381	1990.932	44	44	0	0		1.30562
Agent27	0.649953	0.462824	1.378566	454.2955	44	43	1	0		1.278043
Agent11	0.641321	0.480671	1.343914	2563.068	44	40	4	0		1.228567
Agent29	0.62809	0.501077	1.446251	1039.386	44	44	0	0		1.191085
Agent52	0.587246	0.47403	1.41344	2692.955	44	44	0	0		0.587246
Template/	0.562995	0.44626	1.311601	883.75	44	40	4	0		0.562995

Threshold graph a function over time:



$$0.95 - \frac{0.15e^{10x} - 1}{e^{10} - 1}$$

6 Summary and conclusion

We introduces smart negotiation agent, using opponent modeling and complex random techniques, to maximize average utility, social welfare and confuse our opponent.