INSIGHT: Intelligent Negotiation Strategy for Information Games with Hidden Tactics

by Justin Haddad, Maksym Bondarenko, Phebe Lew

jrh2234@columbia.edu, mb5018@columbia.edu, phebelewyu@gmail.com

Abstract

INSIGHT (Intelligent Negotiation Strategy for Information Games with Hidden Tactics) is the culmination of our research in the use of reinforcement learning and game theory to create optimal bidding strategies in environments with hidden information. The broader goal of this research was to devise strategies applicable to a wide variety of real-world situations, such as trade negotiations, financial markets, and other bidding games. We created a discrete Markov model to simulate bidding episodes in which multiple sellers compete to maximize profit for each buyer. INSIGHT combines different RL and heuristic strategies from existing research to learn over time the most profitable strategies in any given situation. Then, we investigated how fine-tuning of parameters and reward functions within the INSIGHT model affects its performance in different scenarios, competing against opponents with a fixed stochastic strategy, as well as opposing agents with reinforcement learning strategies. Our results show that using this combination of strategies, we can create sellers to outperform any given static opponent, but that two competing Q-Learning sellers which both attempt to adapt to each other fail to cooperate, thus resulting in low pareto efficiency. This mirrors a situation in prisoner's dilemma where both agents play based on the myopic strategy of choosing only to defect, maximizing short term payoffs which eventually lead to a far worse outcome for both agents involved. The INSIGHT model shows the powers and limitations of reinforcement learning for these negotiation situations, and leaves room for a breadth of future research involving other RL algorithms that could learn to cooperate. The code for the simulation, heuristic and RL agents is available at github [LINK].

Keywords: reinforcement learning, game theory, negotiation strategy, Markov Model, reward function, stochastic strategy, Q-Learning

Department of Mathematics, Columbia University and Barnard College

August 8, 2023

1 Introduction

From politics to business, economics and finance, the art of negotiation has always been seen as a crucial means of “dividing utility” between multiple parties. There are two broad elements in negotiation - more qualitative aspects such as linguistic and emotive cues, versus more quantitative aspects such as the optimization of prices proposed/offers made.

Our model, INSIGHT (Intelligent Negotiation Strategy for Information Games with Hidden Tactics), focuses on the latter - employing machine learning and game theory techniques to develop a model able to make optimal negotiation bids, where all offers can be measured entirely quantitatively (e.g. in terms of price). Our aim was to train INSIGHT to extract information from an unknown environment, especially in regard to the demographic of its negotiating opponent as well as the buyer environment, to come up with an optimal bidding strategy, as well as test its ability to compete with other strategic bidding models. INSIGHT utilises modular Q learning - a form of reinforcement learning - to learn how to play a price bargaining game against other bargaining agents. In essence, we simulate a bargaining environment, consisting of buyer types and their distributions, that is foreign to our model and our model learns the best bidding responses in the unknown environment.

2 Literature Review

The issue of bargaining has been studied extensively in game theory; Sequential Veto Bargaining with Incomplete Information[1] examines two forms of equilibria - a skimming equilibria and a leapfrogging equilibria and a skimming equilibrium given a scenario with a proposer, a vetoer, and a discount factor for every unsuccessful negotiation made. This paper provides a framework to study veto bargaining, i.e., bargaining over a one-dimensional policy between two players who have single-peaked preferences. Only one player, Proposer, has the power to make proposals; the other player, Vetoer, decides whether to accept a proposal or reject it and preserve the status quo. Such a game is examined over an infinite horizon. The paper showed that if players are patient, Proposer can achieve a payoff that is arbitrarily close to his payoff with commitment powers.

Similarly, the paper Game Theory Models of Pricing [4] exploring theoretical strategies in state-dependent dynamic pricing games involving Bertrand competition - where there is no product differentiation and companies compete for demand based on pricing strategy. It also investigates competitive strategies like tit-for-tat, along wigth state-dependent dynamic pricing games, where the current payoff is dependent on the game history, and feedback equilibria, where strategies are not just functions of time but dependent on previous states of the game - much like our negotiation game.

In Multiagent reinforcement learning in the Iterated Prisoner's Dilemma [5], Q-learning is used to play a prisoner's dilemma; results found that Q-learning was able to compete against other fixed game theory strategies, but not against other agents with a non-static strategy.

Where reinforcement learning is concerned, Evolution with Reinforcement Learning in Negotiation [6] suggests that the use of reinforcement learning can lead to the evolution of long-term collective performance and strategy and Evaluating Persuasion Strategies and Deep Reinforcement Learning methods for Negotiation Dialogue agents [3] examined reinforcement learning for the board game Settlers of Catan.

3 Methodology

3.1 Negotiation model

Because of the broader applications of game theory to real world negotiations, we decided to model out negotiation as a game, between one or more sellers and buyers one after another hoping to purchase a product. The rules of the game are as follows: every episode a new buyer is generated with a max price they are willing to spend for the product. The sellers are aware of the range that this price was generated in, but do not know the exact value. For the single seller situation, the seller will make an offer every turn, which the buyer will accept if it is at or below their max price, and will reject otherwise. For the multi seller situation, both sellers make a price, and the buyer will accept the lowest price at or below their max price, and will choose at random to break ties. Every time a buyer rejects an offer, they have a chance to simply walk away from the deal, modeled by their "impatience", which will increase every turn.

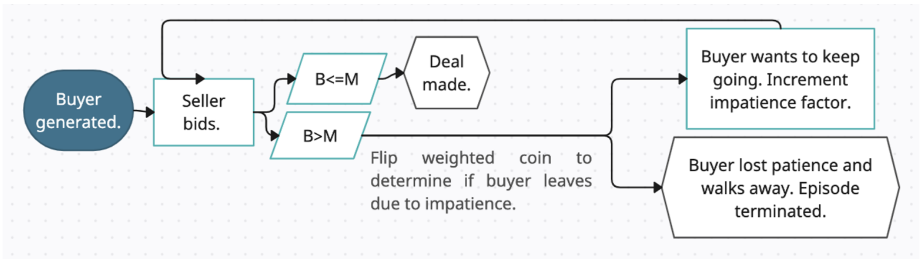
Buyer generated.
Seller bids.
B<=M
Deal made.
B>M
Flip weighted coin to determine if buyer leaves due to impatience.
Buyer wants to keep going. Increment impatience factor.
Buyer lost patience and walks away. Episode terminated.

Figure 1: Flowchart for Single Seller Scenario

Every assumption in this model was made reflect real-world bartering situations as closely as possible. Buyers are generated randomly within a certain range which the sellers are aware of because in practice, most products being sold have market value that both buyers and sellers are generally aware of; it is unlikely that, for example, the same car could be worth anywhere between ten thousand and ten million depending on the buyer. Impatience was chosen as a way to

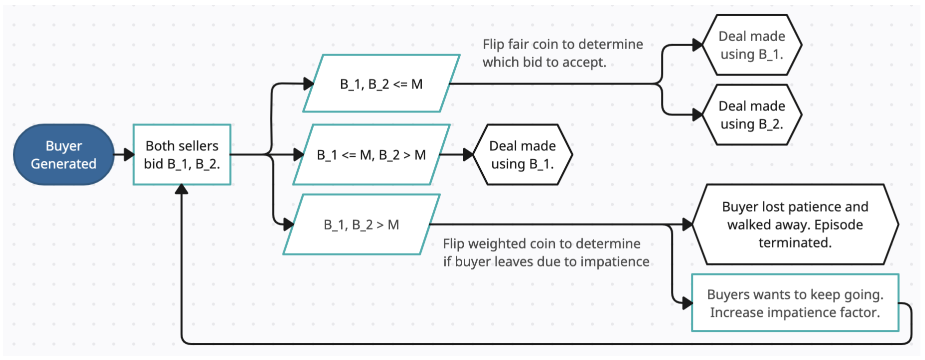
Buyer Generated
Both sellers bid B_1, B_2.
B_1, B_2 <= M
B_1 <= M, B_2 > M
B_1, B_2 > M
Flip fair coin to determine which bid to accept.
Deal made using B_1.
Deal made using B_2.
Deal made using B_1.
Flip weighted coin to determine if buyer leaves due to impatience
Buyer lost patience and walked away. Episode terminated.
Buyers wants to keep going. Increase impatience factor.

Figure 2: Flowchart for 2-Seller scenario

model the time value of a deal. There is risk involved with delaying a deal to increase profit, and thus any given strategy must balance the risk of losing a deal with the possibility of making a higher deal. Of course, for a multi seller situation in the real world, most buyers are not perfectly rational agents and will not always go to the lowest price as there are other conflating variables, but it would simply detract from our model to attempt to simulate that type of irrational behavior.

3.2 Heuristic Agents

Our research utilizes three heuristic agents to examine against our Q-Learning agent.

Random Seller: This is a common benchmark in reinforcement learning and game theory research [5]. It formulates a bid randomly within the confines of the previous bid and the lowest possible bid.

Descent-by-One Seller: This agent is rather naive and may not yield high profits, especially when dealing with a broad range of prices. It offers exactly one less than its previous bid every turn. It serves as a sanity check for comparison against other strategies.

With the aid of game theory, we developed an optimal agent for single seller games. By applying rollback on all potential outcomes, we discovered the optimal strategy, irrespective of the model parameters. This includes a sequence of offers that decrease with a diminishing step size. For instance, the optimal sequence of offers for a range of 11-30 with an impatience factor of 0.1 is *[Insert Sequence]*, resulting in an average profit per round of *[Insert Value]*. We will get value and sequences shortly.

Given the model parameters we used in the final simulation, determining the optimal sequence of offers was computationally demanding. Instead, we recognized that the optimal sequence for 11-30 is decreasing with a diminishing step size and established our third heuristic agent, the DescentArithmetic agent.

Descent-Arithmetic Agent: This agent receives an initial offer and initial descent as parameters, and subsequently, its sequence of offers is determined as follows:

offer_0 = initial,
$\text{offer}_n = \text{offer}_{n-1}$ - (descent - (n-1))

A potential strategy for a range of 11-30 could be [30,24,19,15,12,11]. This algorithm outperformed other heuristic agents in a variety of scenarios and is less computationally intensive than calculating the game-theoretically optimal sequence of offers for each set of parameters.

The following results were obtained for the agents in a one-seller scenario with a range of 11-100 and a HeuristicSeller with a descent set to 40.

3.3 RL Agent

For our model to implement reinforcement learning, it must adhere to a Markov assumption. Each turn must be composed of a (state, action) pair. In a single seller scenario, we defined our state as lastoffer and numoffers, a realistic approximation of the data agents would have access to in real life. This allowed us to capture both the seller's impatience (based on our model, it depends on numoffers) and the current maximum threshold for buyer price. For a double seller scenario, we also included the competitor's last offer in our state.

However, this was not the optimal way to parameterize our model. Treating states as discrete led to a vast number of possible states. In hindsight, treating the state as continuous would have been better, providing a viable direction for future research.

After reviewing existing literature and experimenting with our model, we chose tabular Q learning as our reinforcement learning algorithm. It performed well for one seller scenario, despite the large state-space size. However, with two sellers, it did not converge to the optimal solution within the given number of steps, even though the solution was still very satisfactory.

We employed an epsilon-greedy max Q learning algorithm. Utilizing a different type of Q learning might enhance our model set up, presenting another interesting direction for future research.

It is important to note that we assume our audience is familiar with the basics of Q and reinforcement learning.

In addition, to aid with learning, we implemented learning rate and exploration rate descent. Formulas for both are as follows:

Reinforcement learning, a subset of machine learning, is predicated on rewarding desirable behaviours and punishing undesired ones. It's about how intelligent agents maximize cumulative reward in an environment by receiving positive feedback for good actions and negative feedback for bad ones.

Q-learning, a variant of reinforcement learning, uses Q-values to progressively refine the agent's behaviour. For every state in the game (containing information about the environment that the agent knows at a given time), the agent can perform a certain action, with the resulting state-action pair assigned a reward value.

Q-values for all state-action pairs are stored in a Q-table and updated iteratively through many rounds of play using the formula below, thereby empirically determining the Q-table.

$$Q(s,a) \leftarrow Q(s,a) + \alpha \left[r + \gamma \max_{a'} Q(s',a') - Q(s,a) \right] \tag{1}$$

Where:

- $Q(s,a)$ is the estimated value of action a in state s.
- α is the learning rate.
- r is the immediate reward received after performing action a in state s.
- γ is the discount factor, which determines the importance of future rewards.
- $\max_{a'} Q(s',a')$ is the maximum estimated future reward after performing action a' in state s' (the next state).

3.4 Assumptions

3.4.1 Markov Assumption

The model assumes that each round of a negotiation can be modelled as a state containing only the information of the previous proposed price and the number of propositions that have occurred previously.

This is a key assumption that every round in a negotiation can be predicted given the previous round, and that an optimal action can be taken in each state.

3.4.2 Stochastic Buyer

Buyers in our model behave stochastically in that they have a given percentage chance to walk away from any deal. They will always take a deal that is at or below their max price. Their max price is determined by uniform random distribution within the price range. These are deliberate choices made to model real world buyer behavior as accurately as possible.

3.4.3 Random Variability in the Data

Because we are not working with outside data, but rather determining parameters within our model to randomly simulate buyer behavior, there was no preprocessing or transformation of the data needed.

However, because of the many different elements of the model which contain randomness (buyer impatience, buyer price determination, q-learning exploration vs exploitation, opposing seller behavior, etc...) there was large variability within the final data. These could obfuscate results over the short term, but the trends always emerged over the long term.

3.4.4 Other Assumptions

In our model, we assume that prices are discrete, i.e. integers. We also assume that every product is equal in value and infinite in inventory. This means there is no "opportunity cost" of potentially selling a product to a different buyer for a higher profit. These are assumptions that were made for simplicity's sake, and can easily be modified in the future.

4 Model Development and Validation

Our model was created entirely in Python, implementing Object Oriented programming practices to ensure scalability. This was crucial given that we had an ongoing list of potential add-ons as we brainstormed more ideas.

First, we created a single seller model, which implemented different heuristic sellers, random sellers, and eventually a Q-Learning seller. Then, we were able to use these same seller behaviors within a multi-seller model, and compared the behaviors of each in every situation.

The Q-Learning sellers utilize Q-Tables which must contain every (state,action) pair, meaning that it would grow in size at a rate of n^3 given the range of prices. At first, we were using smaller ranges, so this did not pose an issue; however, as our model evolved, we decided to use dictionaries to represent the Q-Tables, and only include relevant entries within the table, as many (state,action) pairs were rare enough they rarely entered the table.

Our model was trained to optimize average profit over a large number of rounds. As a seller in a new environment, it is evident that the end goal should be to maximize profit over time. Whether in a single or multi-seller environment, the goal was always to find optimal strategies for any given round. As was previously stated, there was short term variability, given the sheer amount of randomness in the model (which accurately represents the chaos present in real world negotiations) but the trends would eventually emerge in the final results, so long as adequate trials were ran for every strategy.

5 Model Results

5.1 One Seller Case

We first tested different sellers and their performance individually (i.e. not competing against any other seller).

5.1.1 Random

Our random seller bids a random amount from within the price range every round. It will always bid lower than its previous bid.

5.1.2 Heuristic (arithmetic)

Our heuristic seller begins bidding at the maximum P_{max} of our price range, then follows with sequential bids of

$$P_{max} - c, P_{max} - (c-1), P_{max} - (c-2) \cdots$$

where c is the initial increment specified heuristically based on the price range and impatience increment.

5.1.3 Q-learning

On the graph, we have plotted the red which shows the evolution of the Q-learning algorithm as it learns the optimal solution, and then the green plot, where the strategy implemented has already been calculated and the optimal move is chosen in every situation. This is to help visualize the explore vs exploit tradeoff in Q-Learning.

5.1.4 Results

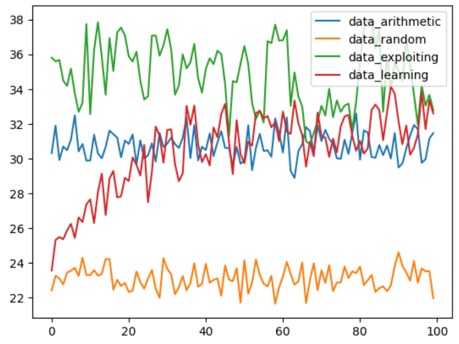

Figure 3: Profit of 4 Different Negotiation Strategies (Q-learning, Q-learning exploit only, Arithmetic Descent, and Random)

5.2 Two Seller Case

Our team tested sellers against each other as they competed for one buyer. Should both sellers offer a price less than or equal to the minimum price the buyer is willing to pay, the buyer accepts the lowest of both offers, and randomly chooses between two equally acceptable offers. Using this two seller model, we can study cooperation and competition between sellers.

5.2.1 Heuristic vs Random

Here, a heuristic algorithm uses an arithmetic descent algorithm. Clearly, the Heuristic Seller is more successful than a Random Seller, though it is worth noting a Random Seller still sometimes wins against a the Heuristic Seller, as it averages around \$2 profit per buyer for a price range of 10-30.

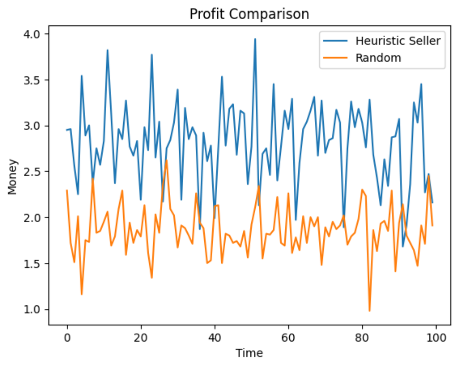

Figure 4: Heuristic vs Random

5.2.2 Q-Learning vs Random

When the Q-Learning Seller plays against the Random Seller, it outperforms the Random Seller by a bigger margin than the Heuristic Seller did. However, in terms of pure profit, both the Q-Learning Seller and the Heuristic Seller eventually settle on around \$3 profit per buyer for a price range of 10-30. We would have expected the Q-Learning seller to have been more profitable, but it seems to have to be more competitive given its competition and settle on a lower price.

5.2.3 Q-Learning vs Heuristic

Here we pit the same Heuristic Seller against the Q-Seller, and we notice the Q-Seller eventually learns to out-perform its opponent.

As can be seen here in the first model begins by offering the minimum price to gain any sort of reward, and thus automatically beat out a heuristic seller

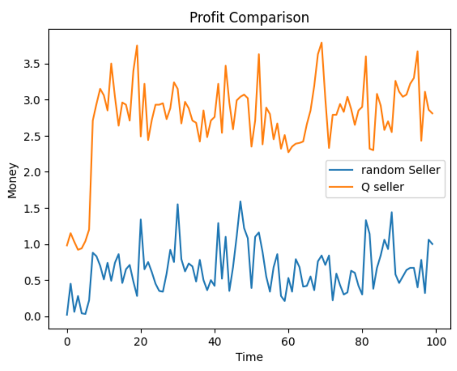

Figure 5: Q-seller vs Random

who likely began with a price higher than the minimum to eventually "work its way down". That is why we can see the first few rounds the Q-Seller earns \$1. However, it eventually explores alternative strategies, which it learns are even more fruitful, and decided to stick to those better strategies.

In the second image, the explorative move the q-seller took at the beginning was not the same minimum price, and thus was not always beating the heuristic seller. Eventually, however, the Q-learning seller learned the optimal strategy to beat its opponent once again, which is shown by the fact that they both converge to the same price. This price both maximizes the average profit and minimized the opponents gain, which is interesting to notice since we only trained the RL algorithm to maximize profit, not minimize opponent's profit.

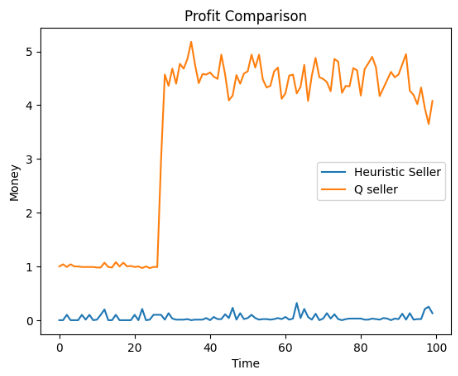

Figure 6: Heuristic vs Q-seller

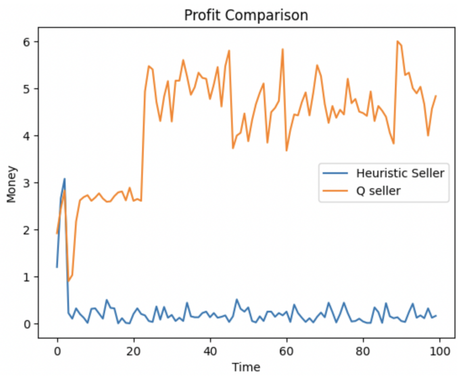

Figure 7: Heuristic vs Q-Seller 2

6 Discussion

Based on the graphs, it is clear the Q-Learning seller outperforms all other fixed strategies, and will eventually learn to beat the heuristic opponents with proven strategies from previous research. It is also worth noting the pareto efficiency of most situations are high, as the combination of the Q-Learning seller and its opponent learn to effectively exploit the environment. However, it is clear that the "worst case" situation arises when two Q-Learning sellers are pitted against each other. Both sellers learn that the only way they can earn any buyers is by bidding lower than their opponent, so both sellers lower their bid every turn until the effective minimum possible bid. This models a similar situation in Prisoner's dilemma, where both prisoners are short-sighted and calculate only one turn in the future, which results in their best option to always defect against their opponent. If our Q-Learning agents were to have more foresight, and understand cooperation is the most effective strategy for combined gain, then both would benefit more in this environment.

Previous literature had attempted similar situations with no results [5] and we had intended to build upon their research. We showed that the shortsightedness of the Q-Learning algorithm cannot effectively exploit any situation in which its opponents are evolving faster than it can learn. This shows the limitations of a Q-Learning algorithm in the real world, as in most cases opposing agents are likely to evolve and adapt to its opponent. One fix for this would be to include multiple episodes of the game within a single round of Q-Learning; however, this would drastically increase the size of the Q-Table and would render them computationally incapable of maintaining usability for more complicated models.

7 Conclusions and Future Research

The model shows us both the benefits and limitations of using reinforcement learning and specifically Q-Learning for iterative negotiations in competitive environments, how it is more successful against stronger yet more consistent and predictable strategies, such as the set heuristics, and worse yet still outperforming less predictable strategies, such as the Random Seller. It also shows that Q-Learning sellers do not learn to cooperate against each other, and instead eventually reach a point of minimum profit for both of them in a situation where they could theoretically both profit far more. The pareto efficiency of this situation is extremely low, as both agents are profiting at close to the minimum possible value.

The Q-Learner does not always produce the optimal result, but it does help inform decisions for situations in which no information about the environment is present, as it balances the tradeoff between exploitation and exploration. By implementing a combination of heuristical algorithms and Q-Learning/RL algorithms, our sellers can learn to maximize their profit in any given situation, which can be directly beneficial to new sellers in unknown environments. By focusing our attention on the pareto efficiency of a given situation, we are able to determine which strategies benefit the sellers as a whole more, which is useful in many real world situations, such as where both competitors are owned by the same umbrella company, or simply to understand how well a given environment is exploited.

There many other heuristic sellers we wish to implement and test against as well. For example, Tit-for-Tat seller [5] will offer slightly below their opponents previous offer every turn. Copycat Seller [2] will remember their opponents bidding strategy and mimic it one step in advance for the next round. Competitive seller notes its opponent's previous bid and increment of decrease and mimics it but increases its decrease increment by one for the next turn. Reactive Seller behaves like Competitive Seller, but drops its price to a non-profit-making price should its opponent decrease its bid by more than 5 in one turn. This seller is intended to simulate how overly-aggressive pricing strategies may force competitors to react with similar aggression and result in a loss of profit for both parties by eroding producer surplus. Should we find a way to increase the state of a seller, we hope to also simulate a limited number of cars on the part of sellers, in hopes that it would be able to maintain a balance between the immediate profit of a completed transaction for the uncertain but possibly larger future profit.

We hope to recreate some version of tit-for-tat where our model is able to play cooperatively and concede a degree of profits to its opponent. Possible methods would include creating a seller to pit against our Q-learning seller that mimics its sequence of bids for future rounds. Another interesting point of study is the rate at which our Q-Seller agent learns. By tuning the parameters of the Q-learning algorithm, we could study how fast the Q-Seller eventually lands on the "optimal", or most profitable, strategy in any situation. This is highly valuable as in real world situations, there are limited numbers of buyers, which forces the Q-Seller to sacrifice early profit for information to learn from, and understanding when it is best to switch from an exploration to an exploitation strategy could reap large rewards.

8 Selected References

References

[1] S. N. Ali, N. Kartik, and A. Kleiner. Sequential veto bargaining with incomplete information. 2023.

[2] A. Dixit, S. Skeath, and D. H. Reiley. *Games of Strategy.* W. W. Norton & Company, 2014.

[3] Simon Keizer, Markus Guhe, Heriberto Cuayáhuitl, Ioannis Efstathiou, Klaus-Peter Engelbrecht, Mihai Dobre, Alex Lascarides, and Oliver Lemon.

Evaluating persuasion strategies and deep reinforcement learning methods for negotiation dialogue agents. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 480–484, Valencia, Spain, April 2017. Association for Computational Linguistics.

[4] P. Kopalle and R. A. Shumsky. Game theory models of pricing. In *Handbook of Pricing Management*. Oxford University Press, 2010.

[5] Tuomas W. Sandholm and Robert H. Crites. Multiagent reinforcement learning in the iterated prisoner's dilemma. *Biosystems*, 37(1):147–166, 1996.

[6] Y. Zou, W. Zhan, and Y. Shao. Evolution with reinforcement learning in negotiation. *PLOS ONE*, 9(7):e102840, 2014.