NEGOTIATING TEAM FORMATION USING DEEP REINFORCEMENT LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

When autonomous agents interact in the same environment, they must often cooperate to achieve their goals. One way for agents to cooperate effectively is to form a team, make a binding agreement on a joint plan, and execute it. However, when agents are self-interested, the gains from team formation must be allocated appropriately to incentivize agreement. Various approaches for multi-agent negotiation have been proposed, but typically only work for particular negotiation protocols. More general methods usually require human input or domain-specific data, and so do not scale. To address this, we propose a framework for training agents to negotiate and form teams using deep reinforcement learning. Importantly, our method makes no assumptions about the specific negotiation protocol, and is instead completely experience driven. We evaluate our approach on both non-spatial and spatially extended team-formation negotiation environments, demonstrating that our agents beat hand-crafted bots and reach negotiation outcomes consistent with fair solutions predicted by cooperative game theory. Additionally, we investigate how the physical location of agents influences negotiation outcomes.

1 INTRODUCTION

Multiple agents inhabiting the same environment affect each other, and may gain by coordinating their actions. Indeed, many tasks are effectively intractable for any single agent, and so can only be solved by a team of collaborators. Examples include search and rescue (Kitano et al., 1999), multi-robot patrolling (Agmon et al., 2008), security (Tambe, 2011) and multiplayer first-person video games (Jaderberg et al., 2018). Despite the need to cooperate, stakeholders have different abilities and preferences which affect the chosen course of action. Agents must therefore negotiate to form teams that are both fairly aligned with individual interests and capable of achieving the task at hand.

This problem can formalized as a *team-formation negotiation* task as follows (Kraus, 1997; Shehory & Kraus, 1998). By definition, no single agent can perform the task on their own, but there may be several teams of agents who are capable of doing so, and therefore each agent must decide who to collaborate with. The reward for accomplishing the task is awarded to the first team that solves it. Hence, agents need to interact with one another to simultaneously form a team and agree on how to share the joint reward. To solve this abstract problem, one must provide a concrete environment in which agents can negotiate and reach an agreement. More specifically, we must specify a *negotiation protocol* that encodes the allowed negotiation actions and determines the agreement reached (Rosenschein & Zlotkin, 1994).

Team-formation negotiation tasks are natural objects of study in game theory. More precisely, *cooperative* game theory focuses on interactions between agents who form teams and make enforceable agreements about outcomes (Brandenburger, 2007; Chalkiadakis et al., 2011). [1] *Weighted voting games* are an archetypal problem, in which every agent has a weight and a team of agents is successful if the sum of the weights of its participants exceeds a fixed threshold (Banzhaf III, 1964; Holler, 1982). Weighted voting games also offer a simple model of coalition formation in legislative bodies (Leech, 2002; Felsenthal et al., 1998). Cooperative game theory seeks to predict the agreements negotiated by agents in such settings, proposing several solution concepts. Some solutions, such as the

[1] This setting is akin to *ad hoc teamwork* (Stone et al., 2010), except it has an initial phase where agents negotiate before solving the problem together, and make binding agreements about sharing a joint reward.

core and nucleolus (Schmeidler, 1969), have focused on identifying stable agreements. Other solutions have tried to identify *fair* agreements between agents, reflecting the strength of the negotiation position of each agent, or measuring their relative ability to affect the outcome of the game. The most prominent of these is the Shapley value (Shapley, 1953b) which has been widely studied for weighted voting games (Shapley & Shubik, 1954; Straffin, 1988). In particular, it has been used to estimate political power (Leech, 2002; Felsenthal et al., 1998). In Appendix A we provide a detailed motivating example, showing how the Shapley value fairly measures power in such settings.

There remains a pragmatic question for the design of multi-agent systems. How should one construct a negotiating agent that maximizes the reward obtained? Many researchers have borrowed ideas from cooperative game theory to hand-craft bots (Zlotkin & Rosenschein, 1989; Aknine et al., 2004; Ito et al., 2007), often requiring additional human data (Oliver, 1996; Lin & Kraus, 2010). Such bots are tailored to specific negotiation protocols, so modifying the protocol or switching to a different protocol requires manually re-writing the bot (Jennings et al., 2001; An et al., 2010). As a result, algorithms based purely on cooperative game theory are neither generally applicable nor scalable.

Moreover, negotiation and team formation in the real world is significantly more complex than in the game theoretic setting, for several reasons: (1) negotiation protocols can be arbitrarily complicated and are rarely fixed; (2) enacting a negotiation requires a temporally extended policy; (3) the idiosyncrasies of the environment affect the negotiation mechanics; (4) Players must make decisions based on incomplete information about others' policies.

We propose multi-agent reinforcement learning as an alternative paradigm which may be applied to arbitrary negotiation protocols in complex environments. Here, individual agents must learn how to solve team formation tasks based on their experiences interacting with others, rather than via hand-crafted algorithms. Our RL approach is automatically applicable to Markov games (Shapley, 1953a; Littman, 1994), which are temporally and spatially extended, similar to recent work in the non-cooperative case (Leibo et al., 2017; Lerer & Peysakhovich, 2017; Foerster et al., 2017).

Some previous work in multi-agent (deep) reinforcement learning for negotiation has cast the problem as one of communication, rather than team formation (e.g. Georgila & Traum (2011); Lewis et al. (2017); Cao et al. (2018b)). In particular, the environments considered involved only two agents, sidestepping the issue of coalition selection. Closer to our perspective is the work of Chalkiadakis & Boutilier (2004); Matthews et al. (2012), which propose a Bayesian reinforcement learning framework for team formation. However, they do not consider spatially extended environments and the computational cost of the Bayesian calculation is significant.

We evaluate our approach on a team formation negotiation task using a direct negotiation protocol, showing that agents trained via independent reinforcement learning outperform hand-crafted bots based on game-theoretic principles. We analyze the reward distribution, showing a high correspondence with the Shapley value solution from cooperative game theory. We show that the slight deviation is not due to lack of neural network capacity by training a similar-sized *supervised* model to predict the Shapley value. We also introduce a more complicated spatial grid-world environment in which agents must move around to form teams. We show that the correspondence with the Shapley value persists in this case, and investigate how spatial perturbations influence agents' rewards.

2 Definitions

Cooperative Games We provide key definitions from cooperative game theory on which our analysis relies (for a full review see Osborne & Rubinstein (1994) and Chalkiadakis et al. (2011)). A (transferable-utility) *cooperative game* consists of a set A of n agents, and a *characteristic function* $v : P(A) \to \mathbb{R}$ mapping any subset $C \subseteq A$ of agents to a real value, reflecting the total utility that these agents can achieve when working together. We refer to a subset of agents $C \subseteq A$ as a *team* or *coalition*. A *simple cooperative game* has v take values in $\{0, 1\}$, where $v(C) = 1$ iff C can achieve the task, modelling the situation where only certain subsets are viable teams. In such games we refer to a team X with $v(X) = 1$ as a *winning team*, and a team Y with $v(Y) = 0$ as a *losing team*. Given a winning team C and an agent $a \in C$, we say that a is *pivotal* in C if $v(C \setminus \{a\}) = 0$, i.e. a's removal from C turns it from winning to losing. An agent is pivotal in a permutation π of agents if they are pivotal in the set of agents occurring before them in the permutation union themselves. Formally, let $S_\pi(i) = \{j | \pi(j) < \pi(i)\}$. Agent i is called pivotal in π if they are pivotal for the set $S_\pi(i) \cup \{i\}$.

Shapley Value The Shapley value characterizes fair agreements in cooperative games (Shapley, 1953b). It is the only solution concept fulfilling important fairness axioms, and is therefore an important quantity in cooperative games (Dubey, 1975; Dubey & Shapley, 1979; Straffin, 1988). The Shapley value measures the proportion of all permutations in which an agent is pivotal, and is given by the vector $\phi(v) = (\phi_1(v), \phi_2(v), \ldots, \phi_n(v))$ where

$$\phi_i(v) = \frac{1}{n!} \sum_{\pi \in \Pi} \left[v(S_\pi(i) \cup \{i\}) - v(S_\pi(i)) \right]. \tag{1}$$

Weighted voting games A *weighted voting game* $[w_1, w_2, \ldots, w_n; q]$ is a simple cooperative game described by a vector of weights $(w_1, w_2, \ldots, w_n)$ and a quota (threshold) q. A coalition C wins iff its total weight (the sum of the weight of its participants) meets or exceeds the quota. Formally $v(C) = 1$ iff $\sum_{i \in C} w_i \geq q$. By abuse of notation, we identify the game description with the characteristic function, writing $v = [w_1, w_2, \ldots, w_n; q]$. The Shapley value of weighted voting games has been used to analyze political power in legislative bodies, and for such settings it is referred to as the Shapley-Shubik power index (Shapley & Shubik, 1954).

Multi-agent Reinforcement Learning An n-player *Markov game* specifies how the state of the an environment changes as the result of the joint actions of n individuals, as follows. The game has a finite set of states $\mathcal{S}$. The observation function $O : \mathcal{S} \times \{1, \ldots, n\} \to \mathbb{R}^d$ specifies each player's d-dimensional view of the state space. We write $\mathcal{O}^i = \{o^i \mid s \in \mathcal{S}, o^i = O(s, i)\}$ to denote the observation space of player i. From each state, players take actions from the set $\mathcal{A}^1, \ldots, \mathcal{A}^n$ (one for each player). The state changes as a result of the joint action $a^1, \ldots, a^n \in \mathcal{A}^1, \ldots, \mathcal{A}^n$, according to a stochastic transition function $\mathcal{T} : \mathcal{S} \times \mathcal{A}^1 \times \cdots \times \mathcal{A}^n \to \Delta(\mathcal{S})$, where $\Delta(\mathcal{S})$ denotes the set of discrete probability distributions over $\mathcal{S}$. Each player receives an individual reward defined as $r^i : \mathcal{S} \times \mathcal{A}^1 \times \cdots \times \mathcal{A}^n \to \mathbb{R}$ for player i. In *reinforcement learning* each agent independently learns, through its own experience of the environment, a behavior policy $\pi^i : \mathcal{O}^i \to \Delta(\mathcal{A}^i)$ (denoted $\pi(a^i|o^i)$) based on its own observation o^i and reward r^i. Each agent's goal is to maximize a long term γ-discounted payoff (Sutton & Barto, 2018). Since agents are independent, the learning process is completely *decentralized* (Bernstein et al., 2002).

3 Game Setting

3.1 Overview

We propose a method for training agents to negotiate team formation under a diverse set of negotiation protocols. In this setting, many different combinations of agents can successfully perform a task. This is captured by an *underlying cooperative game*, given by a characteristic function $v : P(A) \to \{0, 1\}$ (which maps viable teams $C \subseteq A$ to the value 1, and non-viable teams to 0). When a viable team is formed, it obtains a reward r, to be shared between the team's individual members. The outcome of the team formation process is an agreement between agents of a viable team regarding how they share the reward r (with each agent trying to maximize its share). Cooperative game theory can characterize how agents would share the joint reward, by applying solution concepts such as the Shapley value. However, cooperative game theory abstracts away the mechanism by which agents negotiate. To allow agents to interact, form teams and reach an agreement regarding sharing the reward, one must use a *negotiation protocol*, forming an environment with specific rules governing how agents interact; this environment consists of the actions agents can take, and their semantics, determining which team is formed, and how the reward is shared.

We examine two simple negotiation protocols, a non-spatial environment where agents take turns making offers and accepting or declining them, and a spatial environment where agents control their demanded share of the reward in a grid-world setting. Overlaying the underlying cooperative game with a specific negotiation protocol yields a *negotiation environment*; this is a Markov game, which may be analyzed by non-cooperative game theory, identifying the equilibrium strategies of agents. However, solving even the simplest such games is computationally infeasible: even the restricted case of an unrepeated two-player general-sum game is computationally hard to solve, being PPAD-complete (Chen & Deng, 2006) (see Appendix C). Instead, we propose training independent RL agents in the negotiation environment. While our methodology can be applied to any cooperative game, our experiments are based on a weighted voting game as the underlying cooperative game. We

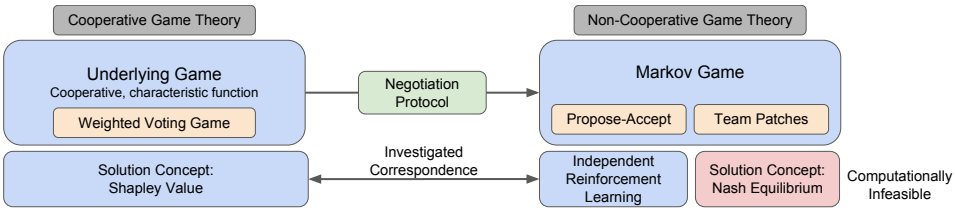


Figure 1: An overview of our methodology. We examine a team formation task defined by an underlying cooperative game. Applying different negotiation protocols to the same underlying task generates different environments (Markov games). Instead of hand-crafting negotiation bots to each such environment, we train independent RL negotiation agents. We compare the agreements RL agents arrive at to game theoretic solutions to the underlying cooperative game.

examine the relation between the negotiation outcome that RL agents arrive at and the cooperative game theoretic solution, as is illustrated in Figure 1.

3.2 Negotiation Environments

Our negotiation environments use a weighted voting game $[w_1, \ldots, w_n; q]$ as the underlying game, offering a total reward $r \in \mathbb{N}$. Each agent i is assigned a weight $w_i \in \mathbb{R}$ at the beginning of each episode, and the goal of the agents is to construct a team C whose total weight $\sum_{i \in C} w_i$ exceeds a fixed quota q. If a successful team forms, the *entire* team is assigned the fixed reward r. The agents comprising the team must decide how to allocate this reward amongst themselves by agreeing on shares $\{r_i\}_{i \in C}$ such that $\sum_{i \in C} r_i = r$, with $r_i \in \mathbb{N}_0$. We say a team $C \subseteq A$ is *viable* if $\sum_{i \in C} w_i \geq q$. Though there are many viable teams, not all teams are viable (depending on the weights). Only one viable team is chosen, and non-members all get a reward of 0. In such settings, agents face opportunities to reach agreements for forming teams, along with their share of the gains. If they do not agree, they stand the risk that some other team would form without them (in which case they get no reward); on the other hand, if they agree to a low share of the reward, they miss out on the opportunity to reach an agreement with others giving them a potentially higher share.

Our two negotiation environments have an identical underlying game, but employ different protocols for negotiating over the formed team and reward allocation. The non-spatial environment uses direct proposals regarding formed teams and reward allocations, while our spatial environment has a similar interpretation to that of recent work applying multi-agent deep reinforcement learning to non-cooperative game theory settings like spatially extended seqential social dilemmas (e.g. Leibo et al. (2017); Perolat et al. (2017); Hughes et al. (2018)).

3.2.1 Propose-Accept (Non-Spatial)

In the Propose-Accept environment, agents take turns in proposing an agreement and accepting or declining such proposals. The underlying game, consisting of the weights and threshold, is public knowledge, so all agents are aware of $v = [w_1, \ldots, w_n; q]$ chosen for the episode. Each turn within the episode, an agent is chosen uniformly at random to be the *proposer*. The proposer chooses a viable team and an integer allocation of the total reward between the agents. The set of actions for the proposer agent consists of all n-tuples $(r_1, r_2, \ldots, r_n)$, where $r_i \in \mathbb{N}_0$ and $\sum_{i=1}^{n} r_i = r$. By convention, the selected team C consists of the agents with non-zero reward under this allocation; that is to say $C = \{i | r_i > 0\}$. We refer to the agents in the team chosen by the proposer as the *proposees*. Once the proposer has made a proposal, every proposee has to either *accept* or *decline* the proposal. If all proposees accept, the episode terminates, with the rewards being the ones in the proposed allocation; if one or more of the proposees decline, the entire proposal is declined. When a proposal is declined, with a fixed probability p a new round begins (with another proposer chosen uniformly at random), and with probability $1 - p$ the game terminates with a reward of zero for all agents, in which case we say the episode terminated with agents *failing to reach an agreement*.

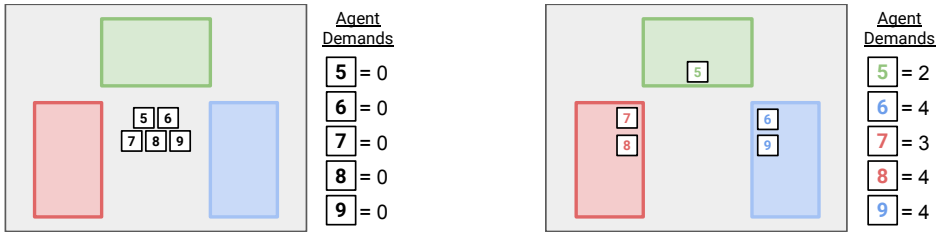
All proposals consist of a viable team and an allocation $(r_1, \ldots, r_n)$ such that $\sum_{i=1}^{n} r_i = r$, so the total reward all the agents obtain in each episode, is either exactly r (when some proposal is accepted), or exactly zero (when agents fail to reach an agreement). Interactions in the Propose-

Accept environment can be viewed as a *non-cooperative* game, however solving this game for the equilibrium behavior is intractable (see Appendix C for details).

3.2.2 TEAM PATCHES (SPATIAL)

We construct a spatial negotiation environment, based on the same underlying weighted voting game as in the Propose-Accept environment, called the Team Patches environment (shown in Figure 2). This is a 15×15 grid-world that agent can navigate, including several colored rectangular areas called *patches*. Agents can form a team with other agents occupying the same patch, demanding a share of the available reward to be a part of the team. Similar to Propose-Accept, the underlying game (i.e. the agent weights and the quota) are fully observable by all agents. Additionally, as the environment is spatial, agents also observe their surrounding area (see the Appendix for details).

At the start of an episode, agents are randomly initialized in the center of the map. Each step, they can move around the environment (forwards, backwards, left, right), rotate (left, right), and set their *demand* ($d_i \in \{1, 2 \ldots, r\}$), indicating the minimal share of the reward they are willing to take to join the team. To form teams, agents must move into patches, and we refer to all agents occupying the same patch as the patch's team. The team T^j in patch j is *viable* if the total weight of the agents in T^j is greater than the threshold ($\sum_{i \in T^j} w_i \geq q$). The demands of the patch's team are *valid* if the total demanded is less than the total reward available ($\sum_{i \in T^j} d_i \leq r$). An agreement is reached once there is a patch whose team is both viable and with a valid reward allocation. We show an example of this in Figure 2(b). An episode terminates when an agreement is reached on one of the patches j, giving each agent i in the team T^j their demanded reward (d_i). Unlike the Propose-Accept environment, the agents don't necessarily use up all the reward available, since $\sum_i d_i < r$ is allowed. If the agents fail to reach an agreement after 100 steps, the episode is terminated, and all agents receive 0 reward.



(a) Environment. (b) Example team formation.

Figure 2: The Team Patches environment for a game with threshold $q = 15$ and a total reward of $r = 7$. Agents are represented by the squares containing their weights. In this example, we have $n = 5$ agents, with the weights $v = [5, 6, 7, 8, 9]$, and three patches where agents can form teams (red (left), green (top), and blue (right)). (a) At the start of an episode, the agents randomly begin in the center of the map with no assigned team or demands. (b) At the end of an episode, the agents have moved inside the grid-world to each of the patches. Agents in red (7 and 8) form a viable team with a valid reward allocation as their weights are above the required threshold ($7 + 8 \geq 15$) and their demands are equal to the availability ($3 + 4 \leq 7$). The team in green is not viable as the total weight is not sufficient ($5 \ngeq 15$), and the blue team has an invalid reward allocation as their demands are higher than the availability ($4 + 4 \nleq 7$). Agents 7 and 8 receive 3 and 4 reward respectively.

3.3 LEARNING AGENTS

For Propose-Accept, each agent independently learns a policy using the SARSA(λ) algorithm (A. Rummery & Niranjan, 1994) with $\lambda = 0.1$. We use a function approximator to learn the Q function, reducing the intractable state space to manageable number of features. The weights of the network are trained with the the Adam optimizer (Kingma & Ba, 2014) to minimize the error associated with the online temporal difference. The function approximator is a multi-layer perceptron with 3 hidden layers, each of size 64.

For Team Patches, we use an advantage actor-critic algorithm (Mnih et al., 2016) with the V-trace correction (Espeholt et al., 2018), allowing us to learn on experience from 16 parallel copies of the

environment using an on-policy algorithm. Each agent is trained independently according to their own experience of states and actions. We use the Adam optimizer (Kingma & Ba, 2014) to adjust the weights of a neural network. The neural network uses a convolutional layer with 6 channels, kernel size 3 and stride 1, followed by a multi-layer perspective with 2 hidden layers of size 32. The policy and value function heads are linear layers. Our training setup is analogous to that of Jaderberg et al. (2018), but with the population size being the number of players, and using no evolution.

4 EXPERIMENTS

4.1 EXPERIMENTAL SETUP

Our experiments are based on a distribution D over underlying weighted voting games. We sample $k = 20$ weighted voting games, $(v^1, \ldots, v^k)$, and refer to them as the experiment's *boards*; each board consists of its weights and threshold $(v^i = [w_1^i, \ldots, w_n^i; q^i])$. Each such board has $n = 5$ agents, a threshold of $q = 15$, and weights sampled from a Gaussian distribution $w_i \sim \mathcal{N}(6, 1)$. We train and evaluate our agents across all $k = 20$ boards, requiring them to learn a general strategy for negotiating under different weights and players for each weight.

4.2 EXPERIMENT 1: COMPARISON WITH HAND-CRAFTED BOTS

We compare the negotiation performance of RL agents trained in our framework with hand-crafted bots. While hand-crafted bots can form excellent negotiators, they can only be used for a specific negotiation protocol. For simplicity, we examine the Propose-Accept protocol where the bot faces two kinds of decisions: (1) what offer to put in as a proposer, and (2) which offers to accept as a proposee. As our baseline, we use a *weight-proportional* bot. Given a proposal, the bot uses the proportion of its weight in the proposed team as a "target" share. The more that is proposed beyond this target, the more likely it is to accept. Bots with a similar design have shown good negotiation performance against both other bots and people (Lin et al., 2008; Lin & Kraus, 2010; Baarslag et al., 2012; Mash et al., 2017). As a proposer, bot i chooses a viable team $C \ni i$ with the weight-proportional allocation $p_i = rw_i/(\sum_{i \in C} w_i)$. As a proposee, bot i computes its share of the same target allocation p_i and compares it with the amount r_i offered to it, and accepts with probability $\sigma(c(r_i - p_i))$ where σ is the logistic function (see the Appendix for full details of the bot).

We create two agent teams (called a *team pair*), one consisting of 5 RL agents (all-RL group), and one consisting of 4 RL agents and one bot (bot group). Each group is co-trained over random boards from the distribution, and evaluated for 500 episodes on an evaluation board. We compare the amount won by the bot and RL agent with the same weight in the board. We repeat the analysis over our 20 evaluation boards, creating 200 team pairs for each evaluation board. In the all-RL group, each agent makes on average $0.2 \cdot r$, whereas in the one-bot group the bot makes on average $0.175 \cdot r$. The difference is statistically significant at the $p < 0.005$ level, using a Mann-Whitney-Wilcoxon test. Additionally, against a random bot that selects valid actions uniformly we get even more significant results ($p < 0.001$). These results indicate that RL-trained agents can outperform sensible hand-crafted bots, without requiring any tailoring for a specific negotiation protocol.

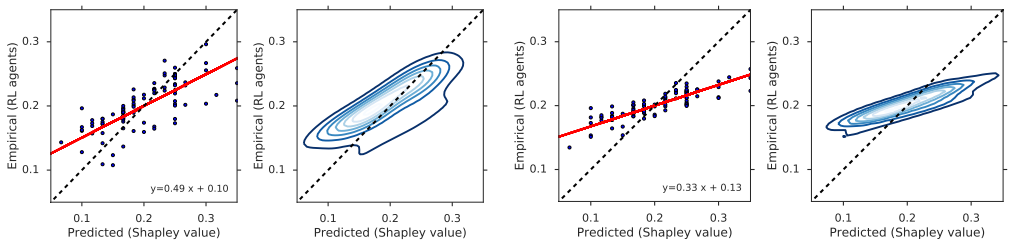
4.3 EXPERIMENT 2: CONSISTENCY WITH GAME-THEORETIC PREDICTIONS

In this experiment, we investigate whether our trained agents negotiate in a way that is consistent with solution concepts from cooperative game theory, focusing on the Shapley value. As the Shapley value was proposed as a "fair" solution, one can view this as studying whether independent-RL agents are likely share the joint gains from cooperation in a fair manner.

To investigate the negotiation policies of our RL agents, we train and evaluate them on our two environments using the same set of $k = 20$ boards. We denote the average amounts won by weight i on board j as s_i^j, and refer to these as *empirical agent rewards*.[2] Given the game v^j we can compute the Shapley values, denoted as $\phi(v^j) = (\phi_1^j, \ldots, \phi_n^j)$. We view ϕ_i^j as the game theoretic prediction of the fair share the agent with weight w_i^j should get in a negotiation in the game v^j. We apply the

[2]For Propose-Accept, we average over 200 independent runs of 5000 episodes each, and for Team Patches we average over 100 independent runs of 1000 episodes each.

analysis to each of our $k = 20$ boards and obtain $k \cdot n$ pairs $\{(\phi_i^j, s_i^j)\}_{i \in [n], j \in [k]}$ of Shapley value predictions and empirical agent rewards. In Figure 3, we scatter and density-plot these pairs.



(a) Propose-Accept.

(b) Team Patches.

Figure 3: Share of reward received by agents in both environments. Shapley points (left) and corresponding density-estimation contours (right). The x-axis is the fair share prediction using the Shapley value, and the y-axis is the empirical reward share of the RL co-trained agents. We include $y = x$ (black dashed) to show the reward our agents would receive if they exactly matched the predictions, as well as a trend line (in red).

Our results indicate a strong correlation between the Shapley value based predictions and the empirical gains of our independent RL trained negotiator agents. We see that the majority of the mass occurs in close proximity to the line $y = x$, indicating a good fit between the game theoretic prediction and the point to which our RL agents converge. RL agents exhibit a stronger deviation from the game theoretic "fair" reward on the boards where there are players with significantly more power or significantly less power than others, i.e. boards where there is a high inequality in Shapley values. This deviation is marginally stronger in our spatial Team Patches environment. While for these cases the high-Shapley weights do empirically get a higher share of the rewards than low-Shapley weights, the gap in empirical reward is lower than the one predicted by game theory. In other words, "strong" players (according to the game theoretic prediction) do get a higher share than "weak" players, but not to the extent predicted; for these rare boards, the empirical allocation of reward achieved by the RL-agents is more equal than predicted by the Shapley value.

In summary, our independent RL negotiation agents empirically achieve rewards that are consistent with the outcomes predicted by the Shapley value from cooperative game theory. Notably, this consistency extends to our spatial negotiation environment even with its different negotiation protocol and spatial interactions, with deviations occurring on boards where players have significantly more or less power than others. In Section 4.5 we investigate potential reasons for these variations.

4.4 Experiment 3: Influence of spatial perturbations

As the Team Patches environment is a spatial one, we can examine how spatial aspects, that are abstracted away in the underlying cooperative game, affect negotiation outcomes. How does changing the spatial structure of the environment influence the "power" of agents? Can we reduce the share of the total reward received by the "most powerful" agent (with the highest weight)?

We consider two patches and vary the starting position of the agent with the highest weight. Specifically, we measure how the agent's share of the total reward changes as we incrementally change its starting position from directly touching a patch to being 10 steps away from the average patch location. Every other agent always starts 3 steps away from the nearest patch. We visualize the results of this in Figure 4 (more details are given in Appendix D). When the highest-weight agent is closer to the patch than the other agents, it can start negotiating a team earlier, and due to its high weight it can demand a high reward. However, as we move the agent further away, it takes longer for the agent to reach a patch, so the other (weaker) agents have more time to form a team; thus, its share of the reward significantly drops.

4.5 Experiment 4: Representational capacity or learning dynamics?

While for most boards our RL agents do indeed converge to an outcome that approximates the Shapley value, there is a deviation on boards where players have particularly high Shapley values ("strong

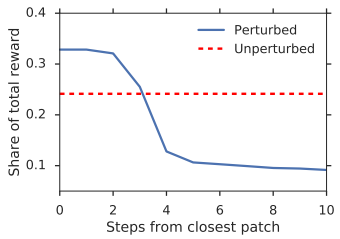


Figure 4: Influence of spatial changes (perturbed, blue) versus no changes (unperturbed, red) on the highest-weighted agent's reward.

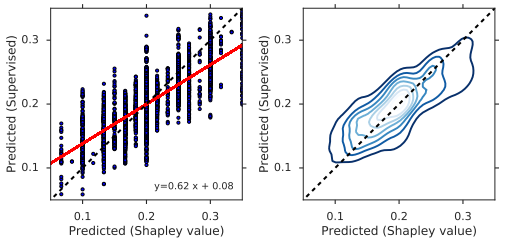


Figure 5: Prediction of the Shapley values from the game weights and threshold using supervised learning.

players") or have particularly low Shapley values ("weak players"). We observe that very weak players get more than predicted by the Shapley value, while very strong players get less. Multiple factors may contribute to this. One potential factor relates to the *representational capacity* of the RL agents; computing the Shapley value in a weighted voting game is known to be an NP-hard problem (Elkind et al., 2009), so perhaps such a function cannot be easily induced with a neural network. Second, it may be difficult to express a policy exactly fitting the Shapley value in the negotiation protocols we've selected. Finally, the deviation might be caused by the *learning dynamics* of independent RL agents; the highly non-stationary environment induced by agents learning at the same time may lead agents to agreements deviating from the cooperative game theoretic predictions for extreme boards.

We rule out the representational capacity factor by conducting a supervised learning experiment. We train a model to take the parameters of a weighted voting game (i.e. the weights of the agents and the threshold), and output the Shapley values of each player. We generate $3,000$ *unique* boards from the same distribution with Gaussian weights used for the experiments in Section 4.3, and apply a train/test partition (80%/20% of the data). We then fit a simple 3-layer MLP with 20 hidden units (smaller than used in the RL agents), minimizing the mean-squared error loss between the model's predictions and the Shapley values. The results of this are shown in Figure 5.

We find that the MLP is not only able to generalize to unseen boards, but moreover is able to better match the game theoretic predictions for a wider distribution of boards. As the deviations from the Shapley value have a similar shape for both the two negotiation protocols we used, we conjecture that the key reason why our RL negotiation agents deviate from the Shapley value for extreme boards is the non-stationary environment induced by the learning dynamics of independent RL agents.

5 Conclusions and Future Work

Team formation is an important problem for multi-agent systems, since many real-world tasks are impossible without the cooperation and coordination of multiple agents. Our contributions are as follows: (1) we introduced a scalable method for team-formation negotiation based on deep reinforcement learning which generalizes to new negotiation protocols and does not require human data, (2) we showed that negotiator agents derived by this method outperform simple hand-crafted bots, and produce results consistent with cooperative game theory, 3) we applied our method to spatially and temporally extended team-formation negotiation environments, where solving for the equilibrium behavior is hard, and (4) we showed that our method makes sensible predictions about the effect of spacial changes on agent behavioral and negotiation outcomes.

This work opens up a new avenue of research applying deep learning to team-formation negotiation tasks. In particular, it would be interesting to analyze how team formation dynamics affect emergent language in reinforcement learning agents, naturally extending the work of (Cao et al., 2018a) and (Lazaridou et al., 2016). Indeed, it has been suggested that the human ability to negotiate and form teams was critical in the evolution of language (Thomas & Kirby, 2018). One might also consider creating tasks that interpolate between the fully cooperative game-theoretic setting and the purely non-cooperative one. Fundamentally, binding contracts are managed by dynamic institutions, whose behavior is also determined by learning. In principle, we could extend our method to this hierarchical case, perhaps along the lines of (Greif, 2006). Perhaps such an analysis would even have implications for the social and political sciences.

REFERENCES

G A. Rummery and Mahesan Niranjan. On-line q-learning using connectionist systems. 11 1994.

Noa Agmon, Sarit Kraus, and Gal A Kaminka. Multi-robot perimeter patrol in adversarial settings. In *Robotics and Automation, 2008. ICRA 2008. IEEE International Conference on*, pp. 2339–2345. IEEE, 2008.

Samir Aknine, Suzanne Pinson, and Melvin F Shakun. An extended multi-agent negotiation protocol. *Autonomous Agents and Multi-Agent Systems*, 8(1):5–45, 2004.

Bo An, Victor Lesser, David Irwin, and Michael Zink. Automated negotiation with decommitment for dynamic resource allocation in cloud computing. In *Proceedings of the 9th International Conference on Autonomous Agents and Multiagent Systems: volume 1-Volume 1*, pp. 981–988. International Foundation for Autonomous Agents and Multiagent Systems, 2010.

Tim Baarslag, Koen Hindriks, Catholijn Jonker, Sarit Kraus, and Raz Lin. The first automated negotiating agents competition (anac 2010). In *New Trends in agent-based complex automated negotiations*, pp. 113–135. Springer, 2012.

John F Banzhaf III. Weighted voting doesn't work: A mathematical analysis. *Rutgers L. Rev.*, 19: 317, 1964.

D. Bernstein, R. Givan, N. Immerman, and S. Zilberstein. The complexity of decentralized control of markov decision processes, 2002.

Adam Brandenburger. Cooperative game theory. *Teaching Materials at New York University*, 2007.

Kris Cao, Angeliki Lazaridou, Marc Lanctot, Joel Z. Leibo, Karl Tuyls, and Stephen Clark. Emergent communication through negotiation. *CoRR*, abs/1804.03980, 2018a. URL `http://arxiv.org/abs/1804.03980`.

Kris Cao, Angeliki Lazaridou, Marc Lanctot, Joel Z Leibo, Karl Tuyls, and Stephen Clark. Emergent communication through negotiation. *arXiv preprint arXiv:1804.03980*, 2018b.

Georgios Chalkiadakis and Craig Boutilier. Bayesian reinforcement learning for coalition formation under uncertainty. In *Proceedings of the Third International Joint Conference on Autonomous Agents and Multiagent Systems, AAMAS 2004*, volume 3, pp. 1090– 1097, 02 2004. ISBN 1-58113-864-4.

Georgios Chalkiadakis, Edith Elkind, and Michael Wooldridge. Computational aspects of cooperative game theory. *Synthesis Lectures on Artificial Intelligence and Machine Learning*, 5(6):1–168, 2011.

Xi Chen and Xiaotie Deng. Settling the complexity of two-player nash equilibrium. In *Proceedings of the 47th Annual IEEE Symposium on Foundations of Computer Science*, pp. 261–272, 2006.

Pradeep Dubey. On the uniqueness of the shapley value. *International Journal of Game Theory*, 4 (3):131–139, 1975.

Pradeep Dubey and Lloyd S Shapley. Mathematical properties of the banzhaf power index. *Mathematics of Operations Research*, 4(2):99–131, 1979.

Edith Elkind, Leslie Ann Goldberg, Paul W Goldberg, and Michael Wooldridge. On the computational complexity of weighted voting games. *Annals of Mathematics and Artificial Intelligence*, 56(2):109–131, 2009.

L. Espeholt, H. Soyer, R. Munos, K. Simonyan, V. Mnih, T. Ward, Y. Doron, V. Firoiu, T. Harley, I. Dunning, S. Legg, and K. Kavukcuoglu. IMPALA: Scalable Distributed Deep-RL with Importance Weighted Actor-Learner Architectures. *ArXiv e-prints*, February 2018.

Dan S Felsenthal, Moshé Machover, et al. The measurement of voting power. *Books*, 1998.

Jakob N. Foerster, Richard Y. Chen, Maruan Al-Shedivat, Shimon Whiteson, Pieter Abbeel, and Igor Mordatch. Learning with opponent-learning awareness. *CoRR*, abs/1709.04326, 2017.

Kallirroi Georgila and David Traum. Reinforcement Learning of Argumentation Dialogue Policies in Negotiation. In *The 12 thAnnual Conference of the International Speech Communication Association (InterSpeech)*, Florence, Italy, August 2011. URL http://ict.usc.edu/pubs/Reinforcement%20Learning%20of%20Argumentation%20Dialogue%20Policies%20in%20Negotiation.pdf.

A. Greif. *Institutions and the Path to the Modern Economy: Lessons from Medieval Trade*. Political Economy of Institutions and Decisions. Cambridge University Press, 2006. ISBN 9781139447065. URL https://books.google.co.uk/books?id=zcrMde1vhLAC.

Manfred J Holler. Forming coalitions and measuring voting power. *Political studies*, 30(2):262–271, 1982.

Edward Hughes, Joel Z Leibo, Matthew G Phillips, Karl Tuyls, Edgar A Duéñez-Guzmán, Antonio García Castañeda, Iain Dunning, Tina Zhu, Kevin R McKee, Raphael Koster, et al. Inequity aversion improves cooperation in intertemporal social dilemmas. In *Advances in neural information processing systems (NIPS)*, Montreal, Canada, 2018.

Takayuki Ito, Hiromitsu Hattori, and Mark Klein. Multi-issue negotiation protocol for agents: Exploring nonlinear utility spaces. In *IJCAI*, volume 7, pp. 1347–1352, 2007.

Max Jaderberg, Wojciech M Czarnecki, Iain Dunning, Luke Marris, Guy Lever, Antonio Garcia Castaneda, Charles Beattie, Neil C Rabinowitz, Ari S Morcos, Avraham Ruderman, et al. Human-level performance in first-person multiplayer games with population-based deep reinforcement learning. *arXiv preprint arXiv:1807.01281*, 2018.

Nicholas R Jennings, Peyman Faratin, Alessio R Lomuscio, Simon Parsons, Michael J Wooldridge, and Carles Sierra. Automated negotiation: prospects, methods and challenges. *Group Decision and Negotiation*, 10(2):199–215, 2001.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Hiroaki Kitano, Satoshi Tadokoro, Itsuki Noda, Hitoshi Matsubara, Tomoichi Takahashi, Atsuhi Shinjou, and Susumu Shimada. Robocup rescue: Search and rescue in large-scale disasters as a domain for autonomous agents research. In *Systems, Man, and Cybernetics, 1999. IEEE SMC'99 Conference Proceedings. 1999 IEEE International Conference on*, volume 6, pp. 739–743. IEEE, 1999.

Sarit Kraus. Negotiation and cooperation in multi-agent environments. *Artificial intelligence*, 94 (1-2):79–97, 1997.

Angeliki Lazaridou, Alexander Peysakhovich, and Marco Baroni. Multi-agent cooperation and the emergence of (natural) language. *CoRR*, abs/1612.07182, 2016. URL http://arxiv.org/abs/1612.07182.

Dennis Leech. Designing the voting system for the council of the european union. *Public Choice*, 113(3-4):437–464, 2002.

Joel Z Leibo, Vinicius Zambaldi, Marc Lanctot, Janusz Marecki, and Thore Graepel. Multi-agent reinforcement learning in sequential social dilemmas. In *Proceedings of the 16th Conference on Autonomous Agents and MultiAgent Systems*, pp. 464–473. International Foundation for Autonomous Agents and Multiagent Systems, 2017.

Adam Lerer and Alexander Peysakhovich. Maintaining cooperation in complex social dilemmas using deep reinforcement learning. *CoRR*, abs/1707.01068, 2017.

Mike Lewis, Denis Yarats, Yann N Dauphin, Devi Parikh, and Dhruv Batra. Deal or no deal? end-to-end learning for negotiation dialogues. *arXiv preprint arXiv:1706.05125*, 2017.

Raz Lin and Sarit Kraus. Can automated agents proficiently negotiate with humans? *Communications of the ACM*, 53(1):78–88, 2010.

Raz Lin, Sarit Kraus, Jonathan Wilkenfeld, and James Barry. Negotiating with bounded rational agents in environments with incomplete information using an automated agent. *Artificial Intelligence*, 172(6):823, 2008.

Michael L. Littman. Markov games as a framework for multi-agent reinforcement learning. In *In Proceedings of the Eleventh International Conference on Machine Learning*, pp. 157–163. Morgan Kaufmann, 1994.

Moshe Mash, Yoram Bachrach, and Yair Zick. How to form winning coalitions in mixed human-computer settings. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence (IJCAI)*, pp. 465–471, 2017.

Tim Matthews, Sarvapali Ramchurn, and Georgios Chalkiadakis. Competing with humans at fantasy football: Team formation in large partially-observable domains. 2, 01 2012.

Volodymyr Mnih, Adrià Puigdomènech Badia, Mehdi Mirza, Alex Graves, Timothy P. Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. Asynchronous methods for deep reinforcement learning. *CoRR*, abs/1602.01783, 2016. URL `http://arxiv.org/abs/1602.01783`.

John Nash. Equilibrium points in n-person games. *Proceedings of the National Academy of Sciences*, 36(1):48–49, 1950.

Jim R Oliver. A machine-learning approach to automated negotiation and prospects for electronic commerce. *Journal of management information systems*, 13(3):83–112, 1996.

Martin J Osborne and Ariel Rubinstein. *A course in game theory*. MIT press, 1994.

Julien Perolat, Joel Z Leibo, Vinicius Zambaldi, Charles Beattie, Karl Tuyls, and Thore Graepel. A multi-agent reinforcement learning model of common-pool resource appropriation. In *Advances in Neural Information Processing Systems (NIPS)*, Long Beach, CA, 2017.

Jeffrey S Rosenschein and Gilad Zlotkin. *Rules of encounter: designing conventions for automated negotiation among computers*. MIT press, 1994.

David Schmeidler. The nucleolus of a characteristic function game. *SIAM Journal on applied mathematics*, 17(6):1163–1170, 1969.

Lloyd Shapley. Stochastic games. *Proceedings of the National Academy of Sciences of the United States of America*, 39:1095–1100, 1953a.

Lloyd S Shapley. A value for n-person games. *Contributions to the Theory of Games*, 2(28):307–317, 1953b.

Lloyd S Shapley and Martin Shubik. A method for evaluating the distribution of power in a committee system. *American political science review*, 48(3):787–792, 1954.

Onn Shehory and Sarit Kraus. Methods for task allocation via agent coalition formation. *Artificial intelligence*, 101(1):165–200, 1998.

Y. Shoham and K. Leyton-Brown. *Multiagent Systems: Algorithmic, Game-Theoretic, and Logical Foundations*. Cambridge University Press, 2009.

Peter Stone, Gal A. Kaminka, Sarit Kraus, and Jeffrey S. Rosenschein. Ad hoc autonomous agent teams: Collaboration without pre-coordination. In *Proceedings of the Twenty-Fourth Conference on Artificial Intelligence*, July 2010.

P Straffin. The shapleyshubik and banzhaf power indices as probabilities. *The Shapley value. Essays in honor of Lloyd S. Shapley*, pp. 71–81, 1988.

R. Sutton and A. Barto. *Reinforcement Learning: An Introduction*. MIT Press, 2nd edition, 2018.

Milind Tambe. *Security and game theory: algorithms, deployed systems, lessons learned*. Cambridge University Press, 2011.

James Thomas and Simon Kirby. Self domestication and the evolution of language. *Biology & Philosophy*, 33(1):9, Mar 2018. ISSN 1572-8404. doi: 10.1007/s10539-018-9612-8. URL https://doi.org/10.1007/s10539-018-9612-8.

Gilad Zlotkin and Jeffrey S Rosenschein. Negotiation and task sharing among autonomous agents in cooperative domains. In *IJCAI*, volume 89, pp. 20–25, 1989.

A POWER INDICES POLITICAL SETTINGS: A MOTIVATING EXAMPLE

Our results compare the agreements of RL agents to the Shapley value in a weighted voting game setting. We provide a simple illustration of how such indices formalize power in a way that depends on the possible teams that can form, rather than the direct parameters of the game (such as the weights). Cooperative game theory has been applied to measuring agent power in decision making bodies (Straffin, 1988; Shapley & Shubik, 1954; Felsenthal et al., 1998). One example of such a domain that has been widely investigated is the formation of a coalition government following an election. Consider a set of political parties who each obtained a certain number of parliament seats in an election, and where a quota of the majority of the seats is required to form a ruling government. If no single party has won the majority of the votes on its own, multiple parties would have to join so as to form a ruling coalition government. This is rare in the UK, and has never happened in the USA, but is the typical situation in many other countries.

For intuition, consider a parliament of 100 seats, two big parties with 49 seats each and a small party with 2 seats; a majority requires 50 seats (half of the total number of seats), so no party can form a government on its own. While each big party has far more seats than the small party, any team of two or more parties has the required majority of seats. Under such a framing of the domain, any team of at least two out the three parties is sufficient for completing the task of forming a government, and due to this symmetry between agents one may claim they all have equal power. In other words, what matters in making a party powerful is not its number of parliament seats, but rather the number of opportunities it has to form teams. Intuitively, one would say that the small party has a strong negotiation position. It could, for instance, demand control of a dis-proportionally high part of the budget (i.e. although it only has 2% of the seats, it is likely to get control of a much larger share of a the budget).

We note that for the above example, the Shapley value would give each party the same share. In fact, the Shapley value as a function does not even take the weights or threshold as input, but rather the characteristic function of the game. We say that agents i, j are *equivalent* in a game with characteristic function v if for any coalition C that contains neither i nor j (i.e. any C such that $i \notin C$ and $j \notin C$) we can add either i or j to the coalition and obtain the same value (i.e. $v(C \cup \{i\}) = v(C \cup \{j\})$). We note that in the above example, all agents are equivalent, and that the Shapley value allocates them the same share. This is not a coincidence, one of the fairness axioms characterizing the Shapley value is that equivalent agents get the same share (Shapley, 1953b; Dubey, 1975). Indeed, the Shapley value is the only index fulfilling a small set of fairness axioms such as this one (see textbooks and papers on axiomatic characterization of the Shapley value for a more detailed picture (Straffin, 1988; Chalkiadakis et al., 2011)).

B BEATING BASELINE BOTS: FULL EXPERIMENT DETAILS

Section 4.2 describes our experiment comparing the negotiation performance of RL agents and hand-crafted bots. We now provide a more detailed discussion of this experiment. We first note that although the negotiation protocols differ across environments, the essence of the decisions agent face is similar. When agents fail to reach an agreement they obtain no reward, so agents who almost never reach an agreement would be bad negotiators. On the other hand, reaching agreement easily by itself does not make an agent a strong negotiator. For instance, an agent who accepts any offer is likely to reach agreement easily but perform poorly, as they would take even very low offers (even when they have a high relative weight and thus a strong negotiation position).

Although the essence of the task is similar across environments, the observations, action spaces and semantics of interaction differ considerably across negotiation protocols. The key advantage of using RL to train negotiation agents is not having to hand-craft rules relating to the specifics of the negotiation environment. Indeed, as hand-crafting a bot is a time consuming process, we focused on the simpler propose-accept environment; creating a hand-crafted bot for the team patches environment is more difficult, as there are many more decisions to be made (how to find a good patch, which share to demand, how to respond if the total reward demanded is too high, etc.)

Our simplest baseline is a *random bot*. As a proposer, it selects an allowed proposal uniformly at random from the set of all such proposals. As a proposee it accepts with probability $\frac{1}{2}$ and rejects with probability $\frac{1}{2}$.

A more sensible basedline is the *weight-proportional bot*, which is an adaptation of previously proposed negotiation agents to the propose-accept protocol (Lin et al., 2008; Lin & Kraus, 2010; Baarslag et al., 2012; Mash et al., 2017) (the propose-accept protocol itself bears some similarity to the negotiation protocols used in such earlier work on negotiation agents).

Section 4.2 provides a short description of the actions taken by the weight-proportional bot. We provide a longer description in this appendix, trying to elaborate on the intuition and principles behind the design of this bot. As a proposer, the weight-proportional bot randomly chooses a viable team C which contains itself from the set of all such teams. It then proposes an allocation of the reward that is proportional to each the team agents' weights. Formally, given a board $[(w_1, \ldots, w_n); q]$, we denote the total weight of a team C by $w(C) = \sum_{i \in C} w_i$. For the team it chose, the bot uses the target allocation $p_i = \frac{w_i}{w(C)} \cdot r$ where r is the fixed total reward. [3]

As a proposee, the bot computes its share of the same target allocation $p_i = \frac{w_i}{w(C)} \cdot r$ in the proposed team C, and compares it with the amount r_i offered to it in the proposal. We denote by $g_i = r_i - p_i$ the amount by which the offer r_i exceeds p_i, the bot's expectations under the weight-proportional allocation. A high positive amount g_i indicates that the bot is offered much more than it believes it deserves according to the weight-proportional allocation, while a negative amount indicates it is offered less than it thinks it deserves (when offered exactly the weight-proportional share we have $g_i = 0$). The probability of the bot accepting the offer is $\sigma(c \cdot g_i)$ where σ denotes the logistic function $\sigma(x) = \frac{1}{1+e^{-x}}$, and where $c = 5$ is a constant correcting the fact that g_i only ranges between -1 and $+1$ rather than between $-\infty$ and $+\infty$. Thus the bot accepts a "fair" offer (according to the weight-proportional target) with probability of $\frac{1}{2}$, and the probability convexly increases as g_i increases (and decreases as g_i decreases).

As discussed in Section 4.2, we compare the performance of the bot and RL trained agents by creating many pairs of groups. We examine each of the 20 evaluation boards (sampled from the distribution of Gaussian weigh boards). For each evaluation board, we create $t = 200$ *pairs* of agent groups, where in each pair we have one group of $n = 5$ independent RL agents (the *all-RL group*), and one set consisting of 1 bot and 4 independent RL agents (the *single-bot group*). Each group is co-trained for $m = 500,000$ episodes, where during training each episode uses a different board sampled at random from the board distribution D. During the evaluation step, we let each agent group play $5,000$ games episodes on the evaluation board. We repeat the same analysis, each time allowing the bot to play a different weight in the board (i.e. letting the bot be the agent in the first position, in the second position and so on).

We investigate the performance of the RL agent from the all-RL group and the bot from the single-bot group, when playing the same weight on the same evaluation board. We average the fraction of the total fixed reward achieved by the RL-agent and bot over the $t = 200$ group pairs and over the 5000 episodes played with each evaluation board, and examine the difference d between the amount won by the RL-agent and the amount won by the bot. A positive value for d indicates the RL-agent has a better performance than the bot, and a negative number shows a higher performance for the bot.

On average (across boards and episodes played on each evaluation board), the RL-agent outperforms the bot, achieving $0.025r$ more of the total reward. In the all-RL group, each agent makes on average $0.2 \cdot r$, whereas in the one-bot group the bot makes on average $0.175 \cdot r$. In other words, the RL-agent obtains 10% more reward than the bot. We performed a Mann-Whitney-Wilcoxon test, which shows the difference is significant at the $p < 0.005$ level. [4] Unsurprisingly, performing a similar analysis for the random bot, that selects valid actions uniformly at random, we get even more significant results ($p < 0.001$).

We note that our result indicates that RL agents can outperform *some simple heuristic* hand-crafted bots. This result certainly does not mean that it is impossible to create well-crafted bots, tailored to a specific negotiation protocol, that would out-negotiate RL agents. In other words, we view this analysis as an indication to RL-agents make at least somewhat sensible negotiation decisions. The

[3] Our framework's action space includes only integral reward allocations, so the bot proposes chooses the integral reward allocation that minimizes the L_1 distance to the target allocation.

[4] This is a non-parametric test similar to Student's T-test, but resistant to deviations from a Gaussian distribution.

key advantage of the methodology we propose is its generality and ability to tackle new negotiation protocols, without relying on hand-crafted solutions. In other words, our technique offers a way of automatically constructing at least reasonable negotiators, without requiring fitting a bot to a given negotiation protocol.

C ON THE COMPLEXITY OF COMPUTING NASH EQUILIBRIA IN THE NEGOTIATION ENVIRONMENTS

We used the Shapley value from *cooperative* game theory to examine likely outcomes in the underlying weighted voting game. The Propose-Accept environment and the Team Patches environment are both based on this underlying game, but define the possible actions agents can take and the outcomes for various action profiles (i.e. they employ a specific negotiation protocol). Thus, they can be viewed as a *non-cooperative* (Markov) game. For such setting, one can also examine solution concepts from *non-cooperative* game theory, such as the Nash equilibrium (Nash, 1950).

The resulting Markov game is not computationally tractable to solve for the Nash equilibrium, as it is an infinitely-repeated n-player general-sum extensive-form game. This is among the hardest class of games to compute an equilibrium for. Even the restricted case of an unrepeated two-player general-sum game is "hopelessly impractical to solve exactly" (Shoham & Leyton-Brown, 2009, Section 4.3), being PPAD-complete (Chen & Deng, 2006). We thus choose to apply cooperative game theoretic solutions, namely the Shapley value.

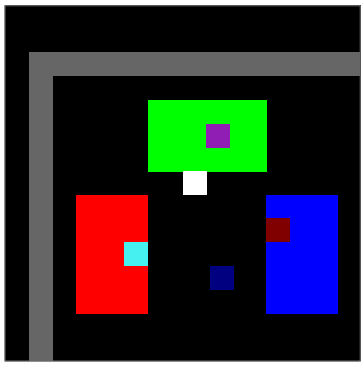
D TEAM PATCHES ENVIRONMENT

Our spatial Team Patches environment is a 15×15 grid-world where each entity has an assigned color. The agents observe this world from their own perspective, as shown in Figure 6(a), and also observe the weights and demands of every other agent, as well as their own index.

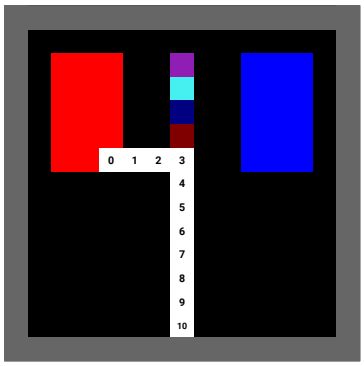
In Experiment 3 (Section 4.4) we change this world in two ways:

1. We set the total number of patches in the world to 2 (red and blue).
2. We modify the starting position of the agent with the highest weight, investigating how this spatial perturbation influences their share of the total reward.

In Figure 6(b) we visualize this with a full view of the environment where the white agent is moved between squares 0 and 10, corresponding to being N steps away from the nearest patch in L^1 distance.



(a) Example agent visual observation, from the perspective of the white agent (centered).



(b) Visualization at the start of an episode for the spatial perturbations experiment.

Figure 6: Team Patches – (a) Example RGB observation an agent receives in an episode. The white square is the observing agent (centered). Other agents are observed as maroon, cyan, purple, and navy, patches as red, green, and blue, and the environment's border is observed as gray. In addition to these observations, the agent also receives its index as a one-hot vector of size number of players, the weights of all agents, and their current demands. (b) Visualization of spatial perturbations where the agent with the highest weight is initialized at 0 to N squares from the nearest patch.