# Negotiating using Rewards

Sarvapali D. Ramchurn[1], Carles Sierra[2], Lluis Godo[2], and Nicholas R. Jennings[1]

[1]School of Electronics and Computer Science,
University of Southampton, Southampton,
SO17 1BJ, UK.
{sdr,nrj}@ecs.soton.ac.uk

[2]Institute of Artificial Intelligence, CSIC,
08193 Bellaterra,
Spain.
{sierra,godo}@iiia.csic.es

## ABSTRACT

In situations where self-interested agents interact repeatedly, it is important that they are endowed with negotiation techniques that enable them to reach agreements that are profitable in the long run. To this end, we devise a novel negotiation algorithm that generates promises of rewards in future interactions, as a means of permitting agents to reach better agreements, in a shorter time, in the present encounter. Moreover, we thus develop a specific negotiation tactic based on this reward generation algorithm and show that it can achieve significantly bettter outcomes than existing benchmark tactics that do not use such inducements. Specifically, we show, via empirical evaluation, that our tactic can lead to a 26% improvement in the utility of deals that are made and that 21 times fewer messages need to be exchanged in order to achieve this under concrete settings.

## Categories and Subject Descriptors

I.2.11 [**Distributed Artificial Intelligence**]: Multi-Agent Systems

## General Terms

Algorithms, Experimentation

## Keywords

Persuasive Negotiation, Bargaining, Argumentation

## 1. INTRODUCTION

Negotiation is a fundamental concept in multi-agent systems (MAS) because it enables (self-interested) agents to find agreements and partition resources efficiently and effectively. Recently, a growing body of work has advocated the use of arguments as a means of finding good agreements [9]. Specifically, it is hypothesised that negotiation using persuasive arguments (such as threats, promises of future rewards, and appeals) allows agents to influence each others' preferences to reach better deals either individually or as a group. Most approaches to *persuasive negotiation* (PN), however, either focus mainly on the protocol [6, 8] used to argue and do not give any insight into the negotiation strategies to be used or fail to give clear semantics for the arguments that are exchanged in terms of their relationship with the negotiated issues [11, 5]. Moreover, most PN reasoning mechanisms adopt a defeasible logic approach [1, 9], rather than the utilitarian approach that we use here. The downside of this logic focus is that it cannot cope as well with the many forms of uncertainty that inevitably arise in such encounters and that it can hardly be benchmarked against standard negotiation algorithms [2, 3].

Against this background, in this work we present a novel reasoning mechanism and protocol for agents to engage in persuasive negotiation in the context of repeated games. We choose repeated games because it is a type of encounter where we believe that persuasive techniques are likely to be most effective (as arguments can be constructed to directly impact future encounters). Now, such encounters have been extensively analysed in game theory [7], but are seldom considered by agent-based negotiation mechanisms. This is a serious shortcoming because in many applications agents need to interact more than once. Specifically, our mechanism constructs possible rewards[1] in terms of constraints on issues to be negotiated in future encounters (hence their semantics are directly connected to the negotiated issues) and our protocol is an extension of Rubinstein's alternating offers protocol [12] that allows agents to negotiate by exchanging arguments (in the form of promises of future rewards or requests for such promises in future encounters).

In more detail, our mechanism gives agents a means of influencing current and future negotiations through promises of rewards, rather than just exchanging offers and counter offers that only impact on the outcome of the present encounter [5, 11]. Thus, we make the rewards endogenous to the negotiation process by assimilating a promise of a reward to promised constraints on resources that need to be negotiated in future. In so doing, we directly connect the value of the argument to the value of the negotiated issues and this allows us to evaluate arguments and offers on the same scale. For example, a car seller may reward a buyer (or the buyer might ask for the reward) who prefers red cars with a promise of a discount of at least 10% (i.e. a constraint on the price the seller can propose next time) on the price of her yearly car servicing if she agrees to buy a blue one instead at the demanded price (as the buyer's asking price for the red car is too low for the seller). Now, if the buyer accepts, it is a better outcome for both parties (the buyer benefits because she is able to make savings in the future that match her preference for the red car and the seller benefits in that he reduces his stock and obtains immediate profit).

Such promises are important in repeated interactions for a number of reasons. First, agents may be able to reach an agreement faster in the *present* game by providing some guarantees over the outcome of subsequent games. Thus, agents may find the current offer and the reward worth more than counter-offering (which only delays the agreement and future games). Second, by involving issues from future nego-

[1]We focus on rewards because of their clear impact on agreements in the context we consider and because we expect threats and appeals to follow similar principles to those we elucidate here.


*AAMAS'06* May 8–12 2006, Hakodate, Hokkaido, Japan.

tiations in the *present* game (as in the cost of servicing in the example above), we effectively expand the negotiation space considered in the present game and, therefore, provide more possibilities for finding (better) agreements [4]. For example, agents that value future outcomes more than their opponent (because of their lower discount factors) are able to obtain a higher utility in future games, while the opponent who values immediate rewards can take them more quickly. Thirdly, if guarantees are given on the *next game*, the corresponding negotiation space is constrained by the reward, which should reduce the number of offers exchanged to search the space and hence the time elapsed before an agreement is reached. Continuing the above example, the buyer starts off with an advantage next time she wants to negotiate the price to service her car and she may then not need to negotiate for long to get a reasonable agreement.

Given this, this work advances the state of the art in the following ways. First, we develop a Reward Generation Algorithm (RGA) that calculates constraints (which act as rewards) on resources that are to be negotiated in future games. The RGA thus provides the first heuristics to compute and select rewards to be given and asked for in our new extension of Rubinstein's protocol. Second, we develop a specific Reward Based Tactic (RBT) for persuasive negotiation that uses the RGA to generate combinations of offers and rewards. In so doing, we provide the first persuasive negotiation tactic that considers the current negotiation game as well as future ones, to generate offers and arguments and thus reach better agreements faster than standard tactics in the long run.

The rest of the paper is structured as follows. Section 2 provides the basic definitions of the negotiation games we consider, while section 3 describes how persuasive negotiation can be used in such games. Section 4 details RGA and section 5 shows how offers and promises are evaluated. Section 6 describes RBT and section 7 evaluates its effectiveness. Finally, section 8 concludes.

## 2. REPEATED NEGOTIATION GAMES

Let $Ag$ be the set of agents and $X$ be the set of negotiable issues. Agents negotiate about issues $x_1, \cdots, x_n \in X$ where each one has a value in its domain $D_1, \cdots, D_n$. Then, a contract $O \in \mathcal{O}$ is a set of issue-value pairs, noted as $O = \{(x_1 = v_1), \cdots, (x_m = v_m)\}$.[2] We will also note the set of issues involved in a contract $O$ as $X(O) \subseteq X$. Agents can limit the range of values they can accept for each issue, termed its negotiation range and noted as $[v^{min}, v^{max}]$. Each agent has a (privately known) utility function over each issue $U_x : D_x \rightarrow [0, 1]$ and the utility over a contract $U : \mathcal{O} \rightarrow [0, 1]$ is defined as $U(O) = \sum_{(x_i=v_i)\in O} w_i \cdot U_x(v_i)$, where $w_i$ is the weight given to issue $x_i$ and $\sum w_i = 1$. We consider two agents $\alpha, \beta \in Ag$ having utility functions designed as per the Multi-Move Prisoners' Dilemma (MMPD) (this game is chosen because of its canonical and ubiquitous nature) [13]. According to this game, $\alpha$'s marginal utility $\delta U$ is higher than $\beta$'s for some issues, which we note as $O^\alpha$, and less for others, noted as $O^\beta$, where $O^\alpha \cup O^\beta = O$.

While it is possible to apply rewards to infinitely or finitely repeated games, we focus on the base case of one repetition in this work because it is simpler to analyse and we aim to understand at a foundational level the impact that such promises may have on such encounters. These games are played in sequence and there may be a delay $\theta$ between the end of the first game and the beginning of the second one. In a game, one agent ($\alpha$ or $\beta$) starts by making an offer $O \in \mathcal{O}$ and the opponent may then counter-offer or accept. The agents may then go on counter-offering until an agreement is reached or one of the agents' deadlines ($t^\alpha_{dead}$ or $t^\beta_{dead}$) is reached. If no agrement is reached before the deadline, the agents obtain zero utility (in either the first or second game). We also constrain the games, and further differentiate them from the case where agents play one game each time independently of the previous one, by allowing the second game to happen *if and only if* the first game has a successful outcome (i.e. an agreement is reached within the agents' deadlines and the contract is executed). In so doing, there is no possibility for agents to settle both outcomes in one negotiation round. The agents may also come to an agreement in the first game but fail to reach one in the second one, in which case the agents only obtain utility from the outcome of the first game.

If an agreement is reached, the agents are committed to enacting the deal settled on. This deal comes from the set of possible contracts which, in the first game, is captured by $\mathcal{O}_1$ and, in the second one, by $\mathcal{O}_2$. During these games, as time passes, the value of the outcome decreases for each agent according to their discount factor (noted as $\epsilon_\alpha$ for agent $\alpha$). This factor denotes how much the resources being negotiated decrease in usefulness over time.'The time between each illocution trasmitted is noted as $\tau$. Then, the discount due to time is calculated as $e^{-\epsilon(\theta+t)}$ between the two games and $e^{-\epsilon(\tau+t)}$ between offers [7] where $t$ is the time since the negotiation started (note that we expect $\theta >> \tau$ generally). The value of $\epsilon$ scales the impact of these delays, where a higher value means a more significant discounting of an offer and a lower value means a lower discounting effect. Each agent is also assumed to have a *target utility* to achieve over the two games (noted as $L \in [0, 2]$). This target can be regarded as the agent's aspiration level for the combined outcomes of the two games [3]. This target must, therefore, be less than or equal to the sum of the maximum achievable utility over the two games (2 in the case an agent has a $\epsilon = 0$ and exploits both games completely); that is $L \leq 1 + e^{-\epsilon(\theta+t)}$, where 1 is the maximum achievable utility in an undiscounted game.

Agents use the illocutions $propose(\alpha, \beta, O)$ and $accept(\alpha, \beta, O)$ to make and accept offers respectively. Additionally, they may use persuasive illocutions such as $reward(\alpha, \beta, O_1, O_2)$ and $askreward(\alpha, \beta, O_1, O_2)$. The former means $\alpha$ makes an offer $O_1$ and promises to give reward $O_2$. The latter that $\alpha$ asks $\beta$ for a promise to give reward $O_2$ (we detail the contents of $O_2$ in the next section). Hence, while the promise of a reward aims to entice an opponent to accept a low utility contract in the current encounter, asking for a reward allows an agent to claim more in future negotiations (in return for concessions in the current one).

## 3. APPLYING PERSUASIVE NEGOTIATION

In persuasive negotiation, agents either promise to give rewards to get their opponent to accept a particular offer or ask for such promises in order to accept an offer. In our case, rewards are specified in the second game in terms of a range of values for each issue. Thus, giving a reward equates to specifying a range such as $v_x > k$ (where $k \in D_x$ is a constant) for issue $x$ in $O_2 \in \mathcal{O}_2$ to an agent whose utility increases for increasing values of $x$. Conversely, asking for a reward means specifying $v_x < k$ in $O_2$ for the asking agent (whose utility increases for decreasing values of $x$). Now,

[2] Other operators $\geq, \leq$ could also be used.

agents may find it advantageous to accept such rewards if it costs them more to counter-offer (due to their discount factor) or if they risk passing their deadline (or their opponent's). Here, we do not deal with the issues related to whether the agents keep to their promises or how to tackle the uncertainty underlying this (we simply assume they do), but rather we focus on the reasoning mechanism that the agents require in order to negotiate using rewards.

Specifically, we propose that agents use the level to which they concede in the first game in order to decide on what to offer or ask for as a reward in the second one. This is graphically illustrated in figure 1 where $O_1 \in \mathcal{O}_1$ and $O_2 \in \mathcal{O}_2$ are the proposed offer and reward respectively.

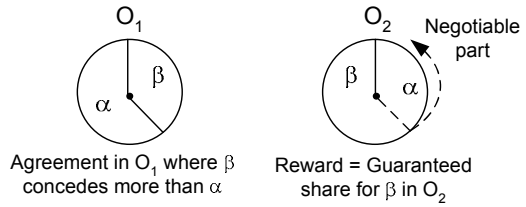


**Figure 1: Representation of an offer $O_1$ made in the first game and a reward $O_2$ for $\beta$ in the second game.**

As can be seen, $\alpha$ exploits $\beta$ (by taking a larger share of the pie) through the offer $O_1$ or alternatively $\beta$ concedes more than $\alpha$ (depending on who makes the offer). The promised reward offered or asked for in the second game then tries to compensate for the exploitation/concession applied to the first game. Now, one strategy that produces this behaviour is the following: the higher the concession, the higher will be the reward demanded, while the lower the concession, the higher will be the reward given.[3] This strategy can be seen as a type of trade-off mechanism whereby agents take gains in the present (or the future) in return for losses in the future (or in the present) [10].

# 4. REWARD GENERATION

Building on the reasoning mechanism presented in section 3, we now develop our reward generation algorithm (RGA) that determines the level of concession made in the first game and hence determines the value of the corresponding reward, and finally decides whether to send it or not. First, we assume that an agent has some means of generating offers $\mathcal{O}_1$. In line with most work on negotiation in the presence of deadlines, we assume the agent's negotiation tactic concedes to some extent until an agreement is reached or the deadline is passed [2]. Then, at each step of the negotiation, based on the concessions made in an offer $O_1 \in \mathcal{O}_1$, RGA computes the reward $O_2 \in \mathcal{O}_2$ and decides if it is to be asked for or given. In more detail, algorithm 1 outlines the main steps of RGA which are then detailed in the following subsections.

## 4.1 Step 1: Compute Concession Degrees

In this context, the degree to which an agent concedes in any game is equivalent to the value it loses on some issues to its opponent relative to what the opponent loses to it on other issues. Assuming $(x = v_x) \in O$ is the value of an issue $x$, and $[v_x^{max}, v_x^{min}]$ is its negotiation range, then we define $\mu_x = U_x(v_x)$, $\overline{U}_x = \max\{U_x(v_x^{min}), U_x(v_x^{max})\}$, and $\underline{U}_x = \min\{U_x(v_x^{max}), U_x(v_x^{min})\}$. From these, we can compute the maximum an agent could get as $\overline{U} = \sum_{x \in X(O)} w_x \overline{U}_x$, the minimum as $\underline{U} = \sum_{x \in X(O)} w_x \underline{U}_x$ and the actual utility as $\Upsilon = \sum_{x \in X(O)} w_x \mu_x$ where $w_x$ is $\alpha$'s relative weight of issue $x$ and $\sum w_x = 1$. These weights can be ascribed the same values given to the weight the issue has in the utility function and can be normalised for the number of issues considered here. Then, an agent $\alpha$ computes the concession degree on the offer $O$ as:

$$con^{\alpha}(O) = \frac{\overline{U} - \Upsilon}{\overline{U} - \underline{U}} \quad (1)$$

It is then possible to calculate concessions on issues with higher and lower $\delta U$ for $\alpha$ using $con^{\alpha}(O^{\alpha})$ and $con^{\alpha}(O^{\beta})$ respectively. Then, the complement of these functions (i.e. $1 - con^{\alpha}(O^{\alpha})$ and $1 - con^{\alpha}(O^{\beta})$) represents how much $\beta$ concedes to $\alpha$ from $\alpha$'s perspective (or how much $\alpha$ exploits $\beta$).

**Algorithm 1** Main steps of the RGA

**Require:** $O_1 \in \mathcal{O}_1, L$

1: Compute concessions in $O_1^{\alpha}$ and $O_1^{\beta}$. {Here the agent determines how much both agents concede on the issues for which they have a higher and lower $\delta U$ than their opponent.}
2: Select $O_2 \in \mathcal{O}_2$ that matches the level of concession in $O_1$
3: Check whether the combination of $O$ and $O_2$ satisfies $L$, adjust $[v^{min}, v^{max}]$ for second game according to values in $O_2$ and send offer and reward.

## 4.2 Step 2: Determine Rewards

To determine which agent concedes more in the game (given that they play a MMPD), $\alpha$ needs to compare its degree of concession on the issues with higher $\delta U$ than $\beta$ (i.e. $O_1^{\alpha}$) and those with lower $\delta U$ than $\beta$ (i.e. $O_1^{\beta}$) (in a zero sum game this is calculated for all issues). To this end, we define three conditions which refer to the case where $\alpha$ concedes as much as $\beta$ ($COOP$), concedes more to $\beta$ ($CONC$), and concedes less than $\beta$ ($EXPL$) respectively as follows:

- $COOP = true$ when $con^{\alpha}(O_1^{\alpha}) + con^{\alpha}(O_1^{\beta}) = 1$ (i.e. $\alpha$ has no grounds to give or ask for a reward).
- $CONC = true$ when $con^{\alpha}(O_1^{\alpha}) + con^{\alpha}(O_1^{\beta}) > 1$ (i.e. $\alpha$ can ask for a reward).
- $EXPL = true$ when $con^{\alpha}(O_1^{\alpha}) + con^{\alpha}(O_1^{\beta}) < 1$ (i.e. $\alpha$ should give a reward).

The above conditions capture the fact that an agent can only ask for a reward if it is conceding in the first game and can only give one if it is exploiting in the first game. It is possible to envisage variations on the above rules as agents may not always want to give a reward to their opponent if they are exploiting in the first game or they may want to ask for one even if they are not conceding. However, these behaviours could be modelled in more complex strategies (which we will consider in future work). But, in so doing, an agent may also risk a failed negotiation. Here, therefore, we focus on the basic rules that ensure agents try to maximise their chances of reaching a profitable outcome.

Now, having determined whether an argument is to be sent or not and whether a reward is to be asked for or given, we can determine the value of the reward. Given that an agent aims to achieve its target $L$, the value chosen for a reward will depend on $L$ and on ($con^{\alpha}(O_1^{\alpha})$, $con^{\alpha}(O_1^{\beta})$) (i.e. the

[3]It should be noted that while the figure pictures a zero-sum game, the applicability of rewards is not limited to such situations. Instead, they can be applied to more complex games such as the MMPD, for which we detail the procedure in the next section (and which we use in our experiments in section 7).

degrees of concession of the agent). We will consider each of these points in turn.

Given $O_1$, the first game standing offer, the minimum utility $\alpha$ needs to get in the second game is $l_2 = L - U(O_1)$. We then need to consider the following two cases (remember $e^{-\epsilon(\theta+t)}$ is the maximum that can be obtained in the second game with discounts). Firstly, if $l_2 \leq e^{-\epsilon(\theta+\tau+t)}$ it is still possible for $\alpha$ to reach its target in the second game (provided the agents reach an agreement in the first one) and, therefore, give (or ask for) rewards as well. The larger $l_2$ is, the less likely that rewards will be given (since less can be conceded in the second game and still achieve $L$). Secondly, if $l_2 > e^{-\epsilon(\theta+\tau+t)}$, it is not possible to give a reward, but an agent may well ask for one in an attempt to achieve a value as close as possible to $l_2$.

For now, assuming we know $l_2 \leq e^{-\epsilon(\theta+\tau+t)}$, it is possible to determine how much it is necessary to adjust the negotiation ranges for all or some issues in $O_2$ in order to achieve $l_2$. Specifically, the agent calculates the undiscounted minimum utility $\frac{l_2}{e^{-\epsilon(\theta+\tau+t)}}$ it needs to get in the second game. Then, it needs to decide how it is going to adjust the utility it needs on each issue, hence the equivalent bound $\nu_{out}$ for each issue, in order to achieve *at least* $\frac{l_2}{e^{-\epsilon(\theta+\tau+t)}}$. Here, we choose to distribute the utility to be obtained evenly on all issues.[4] Thus, the required outcome $\nu_{out}$ of an issue in the second game can be computed as $\nu_{out} = U_x^{-1}\left(\frac{l_2}{e^{-\epsilon(\theta+t)}}\right)$.

Having computed the constraint $\nu_{out}$, the agent also needs to determine how much it should reward or ask for. To this end, the agent computes the contract $\bar{O}$ which satisfies the following properties:

$$con^{\alpha}(\bar{O}_2^{\alpha}) = 1 - con^{\alpha}(O_1^{\beta}) \text{ and } con^{\alpha}(\bar{O}_2^{\beta}) = 1 - con^{\alpha}(O_1^{\alpha})$$

This is equivalent to our heuristic described in section 3 where the level of concession or exploitation in the offer in the first game (i.e. here $O_1 = O_1^{\alpha} \cup O_1^{\beta}$) is mapped to the reward asked for or given in the second one (i.e. here $\bar{O}_2 = \bar{O}_2^{\alpha} \cup \bar{O}_2^{\beta}$). Here also we adopt the same approach as for $\nu_{out}$ and distribute the concessions evenly on all issues. Then, assuming linear utility functions and finite domains of values for the issues, the above procedure is equivalent to reflecting the level of concession on issues with higher $\delta U$ by $\alpha$ onto those with higher $\delta U$ for $\beta$. This is the same as inverting equation 1 given a known $\overline{U}$ and $\underline{U}$ (as defined in step 1), and finding $v_x$ by assigning $\mu_x = \Upsilon$ and inverting $\mu_x$ for each issue (a procedure linear in time with respect to the number of issues considered). Let us assume that for an issue $x$ this results in a **bound** $\nu_r$ (a maximum or minimum according to the type of argument to be sent). Thus, from $\bar{O}_2$, $\alpha$ obtains bounds for all issues in the rewards it can ask from or give to $\beta$. Given this, we will now consider whether to send a reward based on how $\nu_r$ and $\nu_{out}$ compare for an issue $x$.

## 4.3 Step 3: Sending Offers and Rewards

Assume that $\alpha$ prefers high values for $x$ and $\beta$ prefers low ones and that it has been determined that a reward should be offered (the procedure for asking for the reward is broadly similar and we will highlight differences where necessary).

[4] Other approaches may involve assigning a higher $\nu_{out}$ (hence a higher utility) on some issues which have a higher weight in the utility function. In so doing, $\nu_{out}$ may constrain the agent's negotiation ranges so much for such issues that the two agents' ranges may not overlap and hence result in no agreement may be possible. Our approach tries to reduce this risk.

Now, $\alpha$ can determine whether a reward will actually be given and what its value should be according to the following constraints:

1. $\nu_r \geq \nu_{out}$ : $\alpha$ can promise a reward implying an *upper bound* $\nu_r$ on the second game implying that $\alpha$ will not ask for more than $\nu_r$. This is because the target $\nu_{out}$ is less than $\nu_r$ and $\alpha$ can, therefore, negotiate with a revised upper bound of $v^{max'} = \nu_r$ and a lower bound of $v^{min'} = \nu_{out}$. When asking for a reward, $\alpha$ will ask for a *lower bound* $\nu_r$ (i.e. $v^{min'} = \nu_r$) and negotiate with the same upper bound $v^{max'}$ in order to achieve a utility that is well above its target.
2. $\nu_{out} > \nu_r$ : $\alpha$ cannot achieve its target if it offers a reward commensurate with the amount it asks $\beta$ to concede in the first game. In this case, $\alpha$ revises its negotiation ranges to $v^{min'} = \nu_{out}$ (with $v^{max}$ remaining the same). In this case, the agent does not send a reward but simply *modifies its negotiation ranges*. Now, if it were supposed to ask for a reward, $\alpha$ cannot achieve its target with the deserved reward. However, it can still ask $\beta$ for the reward $\nu_r$ (as a lower bound) *and* privately bound its future negotiation to $v^{min'} = \nu_{out}$ while keeping its upper bound at $v^{max}$. In so doing, it tries to gain as much utility as possible.

Now, coming back to the case where $l_2 > e^{-\epsilon(\theta+\tau+t)}$ (implying $\nu_{out} > \nu_r$ as well), the agent that intends to ask for a reward will not be able to constrain its negotiation range to achieve its target (as in point 2 above). In such cases, the negotiation range is not modified and the reward may still be asked for (if $CONC = true$).

Given the above final conditions, we can summarise the rules that dictate when particular illocutions are used and negotiation ranges adjusted, assuming an offer $O_1$ has been calculated and $O_2$ represents the associated reward as shown below:

**Algorithm 2** Step 3 of RGA.

**if** $COOP$ or ($EXPL$ and $\nu_{out} > \nu_r$) for at least one $x \in X(O_2)$ **then**
  $propose(\alpha, \beta, O_1)$.
**end if**
**if** $CONC$ and $l_2 \leq e^{-\epsilon(\theta+\tau+t)}$ **then**
  $askreward(\alpha, \beta, O_1, O_2)$ and modify $[v^{min}, v^{max}]$ for second game.
**end if**
**if** $CONC$ and $l_2 > e^{-\epsilon(\theta+\tau+t)}$ **then**
  $askreward(\alpha, \beta, O_1, O_2)$.
**end if**
**if** $EXPL$ and $\nu_{out} \leq \nu_r, \forall x \in X(O_2)$ **then**
  $reward(\alpha, \beta, O_1, O_2)$ and modify $[v^{min}, v^{max}]$ for second game.
**end if**

With all this in place, the next section describes how the recipient of the above illocutions reasons about their contents.

# 5. EVALUATING OFFERS AND REWARDS

We now describe how an agent evaluates the offers and rewards it receives. Generally, when agents negotiate through Rubinstein's protocol, they accept an offer only when the next offer $O_{next}$ they intend to put forward has a lower (additionally discounted due to time) utility than the offer $O_{given}$ presented to them by their opponent. However, agents using persuasive negotiation also have to evaluate the incoming offer together with the reward they are being asked for or

are being promised. To address this, we follow a similar line of reasoning as above and evaluate a received offer and reward against the offer and reward the agent would have sent in the next negotiation step. From the previous section, we can generally infer that a reward will imply a value $\nu_r$ for a given issue which defines either a lower or an upper bound for that issue in the next negotiation game. Therefore, given this bound, the agent may infer that the outcome $ev$ of any given issue will lie in $[v^{min'}, v^{max'}]$ which might be equivalent to or different from the agent's normal negotiation ranges $[v^{min'}, v^{max}]$ and may take into account the agent's target $\nu_{out}$ (given its target $l_2$) or the value $\nu_r$ itself (as discussed in the previous section).

Specifically, assume $\beta$ is the agent that is the recipient of a reward (given or asked for) and that $\beta$ prefers small values for the issue $x$ being considered. Then, let $\beta$'s negotiable range be $[v^{min}, v^{max}]$ for the issue $x$ and $\beta$'s target be $l_2^\beta$ in the second game (which implies that it needs at least $\nu_{out}$ for the issue in the second game). Now, if $\beta$ receives $reward(\alpha, \beta, O, O_a)$ (or $askreward(\alpha, \beta, O, O'_a)$) for the second game, $O_a$ implies that $\nu_r^\alpha$ is the upper bound proposed for each issue $x$ in $O_a$ ($\nu_r^\alpha$ would be a lower bound in $O'_a$). In the meantime, $\beta$ has calculated another offer $O_{new}$ with a reward $O_b$ in which a bound $\nu_r^\beta$ is to be given to each issue $x$ in $O_b$. Then, for each issue $x$, $\beta$ calculates the negotiable ranges given $\nu_r^\alpha$ as $[v^{min}, \min\{\nu_r^\alpha, \nu_{out}\}]$ (or $[\nu_r^\alpha, \min\{\nu_{out}, v^{max}\}]$ if $O'_a$ is asked for[5]) while it calculates $[\nu_r^\beta, \min\{\nu_{out}, v^{max}\}]$ given $\nu_r^\beta$. We assume $\beta$ can then calculate (e.g. by picking a value over a normal distribution defined in the negotiation range[6]) the expected outcome of each range as $ev_x^\alpha$ for $[v^{min}, \nu_r^\alpha]$ (or $[\nu_r^\alpha, \min\{\nu_{out}, v^{max}\}]$ in the case of $O'_a$) and $ev_x^\beta$ for $[\nu_r^\beta, \min\{\nu_{out}^\beta, v^{max}\}]$ in the case of $O_b$. Given each of these expected outcomes for each issue, the overall expected outcomes, $EO_a \in \mathcal{O}_2$ and $EO_b \in \mathcal{O}_2$, of the second game can be calculated given a reward. Thus, $EO_a$ is the expected outcome of the reward given by $\alpha$ and $EO_b$ is that for the promise of $\beta$. Given that these outcomes have been calculated, the agent then decides to accept or counter offer using the rule below. This evaluates the offer generated against the offer received to decide whether to accept the offer and promise received or send a *reward* illocution (note the addition of discount factors to reflect the time till the next game and between illocutions, that is, sending the counter offer, receiving an accept, and sending the first offer in the second game):

---

**if** $U(O_{new}) \cdot e^{-\epsilon_\beta(\tau+t)} + (U(EO_b) \cdot e^{-\epsilon_\beta(\theta+\tau+t)} \leq U(O) \cdot e^{-\epsilon_\beta(2\tau+t)}) + (U(EO_a) \cdot e^{-\epsilon_\beta(\theta+3\tau+t)}$ **then**
  $accept(\beta, \alpha, O)$
**else**
  $reward(\beta, \alpha, O_{new}, O_b)$
**end if**

---

As can be seen above, if the sum of the utility of the offer and the expected utility of the promise is higher than the offer and reward to be proposed by $\beta$ (discounted over time), $\alpha$'s proposition is accepted. Otherwise, $\beta$ counteroffers with its promise. If instead, a reward $O'_b$ were to be asked for by $\beta$ along with an offer $O_{new}$, then $\beta$ will apply a similar decision rule as above (where $EO'_b$ is the expected outcome $\beta$ calculates for the reward it asks from $\alpha$) in which we simply replace $EO_b$ with $EO'_b$. Finally, in the case where $\beta$ has received a persuasive offer and can only reply with another offer without any argument, $\beta$ calculates the expected outcome of the second game using only its altered negotiation range $[v^{min}, \min\{\nu_{out}, v^{max}\}]$ to elicit $EO''_b$ (which we use to replace $EO_b$ with in the rule above). Note that the second game is left more uncertain in the latter case since the negotiation range has not been tightened by any reward and so the agents may take more time to reach an agreement in the second game (as per section 1).

Having described our mechanism for sending and evaluating rewards and offers, we will now propose a novel tactic that uses it to perform persuasive negotiation.

# 6. THE REWARD BASED TACTIC

As described in section 4, RGA requires an offer to be generated by some negotiation tactic in order to generate the accompanying reward. In this vein, the most common such tactics can be classified as: (i) behaviour-based (BB) – using some form of tit-for tat or (ii) time-based – using Boulware (BW) (concedes little in the beginning before conceding significantly towards the deadline) or Conceder (CO) (starts by a high concession and then concedes little towards the deadline) [2].[7] Now, many of these tactics start from a high utility offer for the proponent (here $\alpha$) and gradually concede to lower utility ones. In our model, this procedure automatically causes RGA to start by promising rewards and then gradually move towards asking for rewards.

To ground our work, we present a novel reward-based tactic (RBT) (based on Faratin's trade-off tactic [3]) that either asks for or gives a reward at any point in the negotiation in order to reach an agreement. To do so, however, the agent needs to know how to evaluate incoming offers and rewards and generate counter-offers accordingly. Given this, we will consider the three main cases in calculating the response to having received an offer and a proposed reward (see algorithm 3).

Case 1: *An offer and a reward have been received and it is possible to counter offer with a reward.*
In this case, $\alpha$ needs to calculate combinations of rewards and offers and choose the combination that it deems most appropriate to send to $\beta$. To calculate these combinations, $\alpha$ first needs to determine the overall utility each combination should have. To achieve this, we use a hill climbing method similar to Faratin et al.'s tactic. In this method, the agent tries to find an offer that it believes is most favourable to its opponent, while not necessarily conceding too much. In our case (particular for the MMPD), this procedure equates to the agent trying to gain more utility on the issues on which it has a higher $\delta U$ and less on those for which it has a lower $\delta U$ than $\beta$.[8] In so doing, the strategy can maximise joint gains in the repeated negotiation encounter.

Thus, the utility to be conceded in the next offer (or utility step), $Su$, is calculated according to the difference that exists between the agent's previous offer and the last one sent by

[5]This range assumes $\nu_{out} \geq \nu_r^\alpha$, but in cases where this is not true the reward proposed by $\alpha$ is automatically rejected.

[6]This is the technique we adopt here. However, other techniques such as fuzzy reasoning or learning mechanisms could also be used to get this value.

[7]Other negotiation tactics might also be resource-based or dependent on other factors. The tactics we select here have been chosen because they have been demonstrated to be relatively successful and are among the most common ones studied in the literature [10, 2].

[8]Note this is different from the point discussed in footnote 4 since here we do not constrain the negotiation ranges, but rather search for offers that may be profitable to both parties.

its opponent.

$$Su(O_1, O_2, O'_1, O'_2) = \begin{array}{l} U(O_1)e^{-t} + U(EO_2)e^{-(\theta+t)} \\ -U(O'_1)e^{-(\tau+t)} + U(EO'_2)e^{-(\theta+2\tau+t)} \end{array}$$

where $O_1$ and $EO_2$ are the previous offer and expected outcome in the second game from $\alpha$'s reward $O_2$ respectively and $O'_1$ and $EO'_2$ are the current offer and the expected outcome of $\beta$'s argument $O'_2$, respectively. If $\alpha$ does not specify a reward $O_2$, $EO_2$ is calculated as per section 5 given the normal negotiation ranges. Similarly, $EO'_2$ is also calculated in the same way if $\beta$ does not specify a reward with the previous offer.

Given the utility step $Su$, it is then possible to calculate the utility $Nu$ of the combination of the next offer and reward using the following equation:

$$\begin{array}{rl} Nu = & U(O_1)e^{-(2\tau+t)} + U(EO_2)e^{-(\theta+3\tau+t)} \\ & -Su(O_1, O_2, O'_1, O'_2, f) \end{array} \quad (2)$$

The next step involves generating combinations of offers and rewards whose combined utility is as close as possible to $Nu$. To this end, we use an optimisation function $OptComb$ : $[0,2] \times \mathcal{O}_1 \times \mathcal{O}_2 \times \mathcal{O}_1 \times \mathcal{O}_2 \rightarrow \mathcal{O}_1 \times \mathcal{O}_2$, based on linear programming, that calculates the reward and offer whose values are most favourable to $\beta$ (but still profitable for $\alpha$). $OptComb$ therefore runs through RGA to find the best possible rewards and the associated offers whose combined utility is less than or equal to $Nu$ and that concede more on issues for which $\beta$ has a higher marginal utility. RGA also informs RBT whether the reward is to be asked for or given and whether negotiation ranges need to be modified (as described in section 4). However, $OptComb$ can also *fail* to find an optimal output (as a result of the constraints being too strong (e.g. the target $L$ being too high) or the optimizer not being able to find the solution in the specified number of steps) and in these cases, we resort to another procedure described next (i.e. Cases 2 and 3).

CASE 2: *OptComb fails and the last offers made involved rewards.*
The agent cannot find a combination of a proposal and a reward whose utility matches $Nu$. Therefore, it calculates an offer using the time-based heuristics presented earlier.[9]

CASE 3: *OptComb fails and the last offers made did not involve rewards.*
It is possible to continue the same step-wise search for an agreement as in case 1. Here, our tactic calculates the offer whose utility is as close as possible to $Nu$ (without $U(EO'_2)$ or $U(EO_2)$). Moreover, the offer calculated is such that it is the one that is most similar to the offer by $\beta$. This is achieved by running an optimization function $OptProp : [0,2] \times \mathcal{O}_1 \times \mathcal{O}_1 \rightarrow \mathcal{O}_1$ that calculates an offer $O_1$ such that $O_1$ maximises the level of concession on issues with higher marginal utility for the opponent (as in case 1) while still achieving $Nu$. In case the issues being negotiated are qualitative in nature, the similarity based algorithm by [3] may be used.

[9]In this case, BB tactics would not be appropriate to generate an offer given previous offers by the opponent. This is because some offers have been proposed in combination with a reward such that the concessions in the offers may not be monotonic (an asked for reward may compensate for a concession in the offer or a concession in the given reward may be compensated for by the higher utility of the offer). The latter property is a requirement for BB (or even all hill-climbing tactics [3]) to work . Therefore, either BW or CO is used to generate the offer since these are independent of the previous offers made by the opponent.

**Algorithm 3** The RBT algorithm.

**Require:** $O_1, O_2, O'_1, O'_2$
1: Use a mechanism to calculate $EO_2, EO'_2$ {$\alpha$ calculates the expected outcomes of the arguments as discussed in section 5.}
2: step = $Su(O_1, O_2, O'_1, O'_2)$ {calculate the utility concession.}
3: nu = $U(O_1)e^{-(2\tau+t)} + U(EO_2)e^{-(\theta+3\tau+t)}$ − step {calculate the utility of the combination of offer and reward to be generated.}
4: $(O''_1, O''_2) = OptComb(\text{nu}, O_1, O_2, O'_1, O'_2)$ s. t. $U(O''_1)e^{-(2\tau+t)} + U(EO''_2)e^{-(\theta+3\tau+t)} \leq$ nu {here the values in the combination are optimised to be more favourable to $\beta$ and as close as possible to nu.}
5: **if** $OptComb$ succeeds **then** {**Case 1**}
6: send $O''_1$ and $O''_2$ {RGA decides whether the reward is asked from or given to $\beta$.}
7: **else if** $OptComb$ fails & (both or one of $O_2$ or $O'_2$ is not null) **then** {**Case 2**}
8: use BW or CO to generate $O''_1$
9: send offer $O''_1$ and modify $[v^{min}, v^{max}]$ to achieve $L$ as in RGA.
10: **else if** $OptComb$ fails & (both $O_2$ and $O'_2$ are null) **then** {**Case 3**}
11: step' = $Su(O_1, null, O'_1, null)$ {calculate the step in utility.}
12: nu' =$U(O_1)e^{-(2\tau+t)}$ − step' {calculate the utility of the offer to be generated.}
13: $O''_1$ = $OptProp(\text{nu}', O_1, O'_1)$ s.t. $U(O''_1) \leq$ nu' {find the offer that is most favourable to $\beta$ but as close as possible to nu'.}
14: send offer $O''_1$ and modify $[v^{min}, v^{max}]$ to achieve $L$ as in RGA.
15: **end if**

We capture all the above three cases in algorithm 3. As can be seen, RBT only generates offers and rewards in the first game. In the second one, we use a time-based or behaviour-based heuristic to calculate offers. While it is certainly possible to generate offers using the optimisation function of RBT in the second game, we do not do so in order to focus our analysis on the effect the bounds imposed by rewards have on the outcome of the second game when agents use basic tactics.

# 7. EXPERIMENTAL EVALUATION

In this section, we describe a series of experiments that aim at evaluating the effectiveness and efficiency of our PN model in repeated interactions. To this end, we evaluate it against basic tactics using standard benchmark metrics. In the following sections, we first detail the experimental settings and then provide the results of these experiments.

## 7.1 Experimental Settings

Agents $\alpha$ and $\beta$ negotiate over 4 issues $x_1, ..., x_4$ and their preferences are as per a MMPD. Thus, $\delta U^{\alpha}_x > \delta U^{\beta}_x$, where $x \in x_1, x_2$ such that $x_1$ and $x_2$ are more valued by $\alpha$ than $\beta$, while $x_3$ and $x_4$, are more valued by $\beta$ than $\alpha$ (i.e. $\delta U^{\alpha}_y < \delta U^{\beta}_y$, where $y \in x_3, x_4$). $t_{max}$ is set to 2 seconds which is equivalent to around 300 illocutions being exchanged between the two agents (in one game).[10] The agents' deadlines, $t^{\alpha}_{dead}$ and $t^{\beta}_{dead}$, are defined according to a uniform distribution between 0 and 2 seconds. The discount factors, $\epsilon_{\alpha}$ and $\epsilon_{\beta}$, are set to a value between 0 and 1 and are drawn from a uniform distribution. The targets of the agents $L^{\alpha}$ and $L^{\beta}$ are drawn from a uniform distribution between 0 and 2. We set $\theta = 0.5$ and $\tau = 0.0001$ to simulate instantaneous replies and set the degree of intersection of the negotiation ranges

[10]Preliminary experiments with the negotiation tactics suggest that if the agents do not come to an agreement within this time period, they never achieve any agreement (even if the maximum negotiation time is extended).

to 0.8 (which means that $[v_\alpha^{min}, v_\beta^{max}]$ overlap $[v_\alpha^{min}, v_\alpha^{max}]$ and $[v_\beta^{min}, v_\beta^{max}]$ by 80%).

| | Utility function and weight of each issue | | | |
|---|---|---|---|---|
| | $U_{x_1}$, $w_{x_1}$ | $U_{x_2}$, $w_{x_2}$ | $U_{x_3}$, $w_{x_3}$ | $U_{x_4}$, $w_{x_4}$ |
| $\alpha$ | $0.4x_1$, 0.5 | $0.9x_2$, 0.2 | $1 - 0.2x_1$, 0.2 | $1 - 0.6x_2$, 0.1 |
| $\beta$ | $1 - 0.2x_1$, 0.4 | $1 - 0.6x_2$, 0.1 | $0.9x_2$, 0.3 | $0.4x_1$, 0.2 |

**Table 1: Utility functions and weights of issues for each agent.**

We will further assume that the first offer an agent makes in any negotiation is selected at random from those that have the highest utility. Also, the agent that starts the negotiation is chosen at random. This reduces any possible first-mover advantage that one strategy may have over another (i.e. which loses less utility due to discount factors). Moreover, in order to calculate the expected outcome of the second game (as discussed in section 5), agents draw the outcome for each issue from a normal distribution with its mean centred in the middle of the agent's negotiation range for the second game with a variance equal to 0.5. Finally, in all our experiments we use ANOVA (ANalysis Of VAriance) to test for the statistical significance of the results obtained.

Given these game settings, we define the populations of negotiating agents in terms of the tactics they use. As discussed in section 6, a number of tactics are available in the literature for experimentation and we will use BB tactics, as well as BW and CO, to generate offers for the RGA algorithm. Moreover, we will compare the performance of these with RBT. The settings of the strategies (i.e. the combination of tactics for the two games) played by the agents is given in table 2. Here, the populations of standard non-persuasive agents (i.e. disconnected from RGA) using only BB, BW, or CO in both games are noted as NT (negotiation tactics), while those that are connected to RGA are noted as PNT (persuasive negotiation tactics). The population of agents using RBT is noted as RBT.

| Game | Strategies | | |
|---|---|---|---|
| | Non-Persuasive | Persuasive | |
| Type | NT | PNT | RBT |
| 1 | BB, BW, CO | PBB, PBW, PCO | RBT |
| 2 | BB, BW, CO | BB, BW, CO | ANY |

**Table 2: Settings for agents' tactics.**

As can be seen from the above table, agents can use rewards in the first game and revert to standard tactics for the second one. For example, a PNT agent, using BW with RGA in the first game, uses BW in the second game. For RBT agents, we randomly select among the three standard tactics.

Given that persuasive strategies like PNT and RBT can constrain their rewards and negotiation ranges according to their target $L$ (as shown in section 4.2), we also need to allow other non-persuasive tactics to constrain their ranges accordingly to ensure a fair comparison. Thus, we allow all tactics to constrain the ranges of the issues in the second game according to their target whenever they reach agreements without the use of any arguments. The procedure to do so is similar to that described in section 4.2.[11] In the following experiments, we use homogeneous populations of 80 agents for each of NT, PNT, and RBT and also create a population of equal numbers of RBT and PNT agents (40 each) which we refer to as PNT&RBT to study how RBT and PNT agents perform against each other.

[11]The difference between the constraint applied by the reward and by the target is that the former applies the constraint to both agents, while the latter only applies separately to each agent according to their individual targets.

Given the populations of agents described above we next define the means used to decide whether PN indeed achieves *better* agreements *faster* than standard negotiation mechanisms. We therefore apply the following metrics:

1. Average number of offers — the average number of offers that agents need to exchange before coming to an agreement. The smaller this number the less time the agents take to reach an agreement.
2. Success rate — the ratio of agreements to the number of times agents meet to negotiate.
3. Average utility per agreement — the sum of utility of both negotiating agents over all agreements divided by the number of agreements reached.
4. Expected utility — the average utility weighted by the probability that an agreement is reached.

Given these requirements, in the following subsection we detail experiments with populations defined above and evaluate their performances.

## 7.2 Empirical Results

In this section, we postulate a number of hypotheses regarding the performance of RGA and RBT and describe the results which validate them.

**H1** *Negotiation tactics that use RGA are more time efficient than those that do not.*

This hypothesis follows from the fact that we expect arguments to help agents find agreements faster. Here we record the average number of offers (the lower this number the more time efficient the agents are) an agent makes in order to reach an agreement. For all populations of tactics, each agent meets another agent 50 times and this is repeated 15 times and the results averaged. Thus it was found that NT takes an average of 547 offers to reach an agreement, while PNT strategies take 58 and PNT&RBT takes 56.5 offers per agreement (nearly 10 times less than NT). Thus, the performance of RBT is significantly better than the other populations since it reaches agreements within only 26 offers (which is less than NT by a factor of 21). Now, the reason for the superior performance of persuasive tactics in general is that the rewards make offers more attractive and, as we expected, the shrinkage of negotiation ranges in the second game (following from the application of the rewards) further reduces the negotiation space to be searched for an agreement. The additional improvement by RBT can be attributed to the fact that both negotiating agents calculate rewards and offers (through the hill-climbing algorithm) that give more utility to their opponent on issues for which they have a higher marginal utility (as explained in section 6). Hence, this is faster than for PNT&RBT in which only one party (the RBT) performs the hill-climbing.

These results suggest the outcomes of RBT and PNT populations should be less discounted and should also reach more agreements (since they take less time to reach an agreement and hence do not go over the agents' deadlines). However, it is not clear whether the utility of the agreements reached will be significantly higher than for NT agents.

**H2** *Negotiation tactics that use the RGA achieve a higher success rate, expected utility, and average utility than those that do not.*

To test this hypothesis, we run the same experiments as above and record the average utility per agreement and the number of agreements reached. Thus, it is possible to calculate the expected utility, average utility per encounter, and the success rate per game as explained earlier.

It was found that the success rate of persuasive strategies is generally much higher than NT (0.87/encounter for NT, 0.99/encounter for PNT only, 1.0/encounter for PNT&RBT, and 1.0/encounter for RBT). This result[12] clearly shows that the use of RGA increases the probability of reaching an agreement. The similar performance of RBT and PNT&RBT and the difference between PNT&RBT and PNT shows that RBT agents, as well as being able to find agreements readily with their similar counterparts, are also able to persuade PNT agents with more attractive offers. This is confirmed by the fact that the average utility of persuasive strategies is generally higher[13] (i.e. 1.9/encounter for PNT, 1.95/encounter for PNT&RBT, and 2.03/encounter for RBT) than NT (i.e. 1.84/encounter). Note that the difference in utility between NT and other tactics would be much greater if discount factors $\epsilon_\alpha$ and $\epsilon_\beta$ were bigger (given the high average number of offers NT uses (i.e. 547)).

Given the trends in success rate and average utility, the expected utility followed a similar trend with NT agents obtaining 1.6/encounter, PNT 1.88/encounter (i.e. a 17.5% improvement over NT), PNT&RBT 1.95/encounter, and 2.03/encounter for RBT agents only (representing a 26% better performance than NT). Generally speaking, from the above results, we can infer that RGA, used together with basic tactics, allows agents to reach better agreements much faster and more often.

These results also suggest that PNT agents reach broadly similar agreements (in terms of their utility) to NT agents (if we discount the fact that rewards significantly reduce the time to reach agreements and increase the probability of reaching an agreement). Now, as discussed in section 6, PNT agents usually generate offers first (starting from high utility ones as for the NT agents) and then calculate the rewards accordingly. Given this, the agents tend to start by giving rewards and end up asking for rewards. As the negotiation proceeds (if the offers are not accepted), the offers generally converge to a point where agents concede nearly equally on all issues (irrespective of the marginal utilities of the agents) and the rewards converge to a similar point. This, in turn, results in a lower overall utility over the two games than if each agent exploits the other one in each game in turn. Now, if rewards are selected in a more intelligent fashion, as in RBT, the agents reach much higher overall utility in general. This is because agents exploit each other more on the issues for which they have a higher marginal utility than their opponent. This is further demonstrated by the results of the RBT agents which suggest they reach agreements that have high utility for both participating agents. However, it is not apparent whether RBT agents are able to avoid being exploited by their PNT counterparts in such agreements which RBT tries to make more favourable to PNT agents (as described in section 6).

**H3** *Agents using RBT are able to avoid exploitation by standard tactics connected to RGA (i.e. PNT).*

In order to determine which tactic is exploited, we recorded PNT's and RBT's average utility separately. Thus, it was found that on average, both RBT and PNT agents obtained about the same average utility per agreement (i.e. 0.96/agreement). This result[14] validates **H3** and suggests that the hill-climbing mechanism of RBT agents calculates offers that can convince the opponent without reducing the utility of both RBT and PNT agents significantly (i.e. in small steps) and also that it maximises joint gains through *OptComb*.

# 8. CONCLUSIONS

In this paper we introduced a novel persuasive negotiation protocol that allows agents in the present encounter to give and ask for rewards in future encounters. To complement this protocol, we also developed a reasoning mechanism that consists of a reward generation algorithm (RGA) and a reward based tactic (RBT). We then showed that RGA can improve the utility gain of standard negotiation tactics by up to 17%, and that RBT provides an additional utility gain of 26% while using 21 times fewer messages to reach a deal in our context.

Future work will look at extending RGA and RBT to more than two games and exploring other strategies to generate rewards, as well as other types of arguments such as threats and appeals. Furthermore, using trust models, we will develop techniques to deal with agents that may not fulfill their promises.

# 9. REFERENCES


[1] L. Amgoud and H. Prade. Formal handling of threats and rewards in a negotiation dialogue. In *Proceedings of the Fourth International Conference on Autonomous Agents and Multi-Agent Systems, Utrecht*, pages 529–536, 2005.

[2] P. Faratin, C. Sierra, and N. R. Jennings. Negotiation decision functions for autonomous agents. *International Journal of Robotics and Autonomous Systems*, 24(3-4):159–182, 1998.

[3] P. Faratin, C. Sierra, and N. R. Jennings. Using similarity criteria to make trade-offs in automated negotiations. *Artificial Intelligence*, 142(2):205–237, 2002.

[4] N. R. Jennings, P. Faratin, A. R. Lomuscio, S. Parsons, C. Sierra, and M. Wooldridge. Automated negotiation: prospects, methods and challenges. *International Journal of Group Decision and Negotiation*, 10(2):199–215, 2001.

[5] S. Kraus, K. Sycara, and A. Evenchik. Reaching agreements through argumentation: A logical model and implementation. *Artificial Intelligence*, 104(1–2):1–69, 1998.

[6] P. McBurney, R. M. van Eijk, S. Parsons, and L. Amgoud. A dialogue-game protocol for agent purchase negotiations. *Journal of Autonomous Agents and Multi-Agent Systems*, 7(3):235–273, 2003.

[7] A Muthoo. *Bargaining Theory with Applications*. Cambridge University Press, 1999.

[8] S. Parsons, C. Sierra, and N. R. Jennings. Agents that reason and negotiate by arguing. *Journal of Logic and Computation*, 8(3):261–292, 1998.

[9] I Rahwan, S D Ramchurn, N R Jennings, P McBurney, S D Parsons, and L Sonenberg. Argumentation-based negotiation. *The Knowledge Engineering Review*, 18(4):343–375, 2003.

[10] H. Raiffa. *The Art and Science of Negotiation*. Belknapp, 1982.

[11] S. D. Ramchurn, N. R. Jennings, and C. Sierra. Persuasive negotiation for autonomous agents: A rhetorical approach. In C. Reed, editor, *Workshop on the Computational Models of Natural Argument, IJCAI*, pages 9–18, 2003.

[12] A. Rubinstein. Perfect equilibrium in a bargaining model. *Econometrica*, 50:97–109, 1982.

[13] G. Tsebelis. Are sanctions effective? A game theoretic analysis. *Journal of Conflict Resolution*, 34:3–28, 1990.


[12] Using ANOVA, it was found that for a sample size of 15 for each population of PNT, PNT and RBT, and PNT only, with $\alpha = 0.05$, $F = 8.8 > F_{crit} = 3.15$ and $p = 4.41 \times 10^{-4}$. These results prove that there is a significant difference between the means of PNT and the other strategies. The success rate of NT agents was found to be always lower than the other populations.

[13] These results were validated statistically using ANOVA, where it was found that $F = 3971 > F_{crit} = 2.73$, and $p = 7.36 \times 10^{-80}$, for a sample size of 15 per population and $\alpha = 0.05$. These results imply that there is a significant difference between the means of the populations.

[14] We validated this result using ANOVA with a sample of size 15 per strategy and $\alpha = 0.05$. Thus it was found that the null hypothesis (i.e. equal means for the two samples) was validated with $F0.13 < F_{crit} = 4.10$ and $p = 0.71 > 0.05$.