Deep Learnable Strategy Templates for Multi-Issue Bilateral Negotiation

Extended Abstract

Pallavi Bagga
Royal Holloway, University of London
Egham, United Kingdom
pallavi.bagga@rhul.ac.uk

Nicola Paoletti
Royal Holloway, University of London
Egham, United Kingdom
nicola.paoletti@rhul.ac.uk

Kostas Stathis
Royal Holloway, University of London
Egham, United Kingdom
kostas.stathis@rhul.ac.uk

ABSTRACT

We propose the notion of deep reinforcement learning-based strategy templates for multi-issue bilateral negotiation. Each strategy template consists of a set of interpretable parameterized tactics that are used to decide an optimal action at any time. This contrasts with existing work that only estimates the threshold utility for those tactics that require it. As a result, we build automated agents for multi-issue negotiations that can adapt to different negotiation domains without the need to be pre-programmed.

KEYWORDS

Multi-Issue Negotiation; Deep Reinforcement Learning; Bilateral Automated Negotiation; Interpretable Negotiation Strategies

ACM Reference Format:
Pallavi Bagga, Nicola Paoletti, and Kostas Stathis. 2022. Deep Learnable Strategy Templates for Multi-Issue Bilateral Negotiation: Extended Abstract. In *Proc. of the 21st International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2022), Online, May 9–13, 2022*, IFAAMAS, 3 pages.

1 INTRODUCTION

We study the problem of modelling a self-interested agent negotiating with an opponent over multiple issues while learning to optimally adapt its strategy. Recent work [3] develops interpretable strategy templates to guide the use of a series of tactics whose optimal use can be learned during negotiation. The structure of such templates depends upon a number of learnable choice parameters, determining which acceptance and bidding tactic to employ at any time during negotiation. As these tactics represent hypotheses to be tested by the strategy developer, they can be explained to a user, and can depend on learnable parameters.

The benefit of the above work is that it can combine three different approaches: hand-crafted predefined heuristics, meta-heuristics and machine learning algorithms. Heuristics are used for the components of the template and meta-heuristics or machine learning are used for evaluating the choice parameter values of these components. At first, the choice parameters of the components for the acceptance and bidding templates were learned once (during training) and used in all the different negotiation settings (during testing) [3]. However, such a one-size-fits-all mechanism for learning the choice parameters does not accumulate experience and abstracts away from what is learned in a specific domain once the negotiation has finished, so it cannot transfer experience to new domains or unseen opponents. To address these issues, we propose the idea of using Deep Reinforcement Learning (DRL) to estimate the choice parameter values of components in strategy templates. We name the proposed interpretable strategy templates as "Deep Learnable Strategy Templates (DLST)".

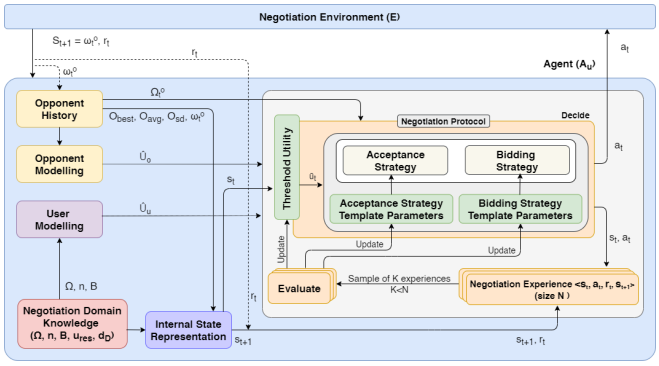

Figure 1: The DLST Agent Negotiation Model

2 DLST-BASED NEGOTIATION MODEL

Given the negotiation setting and agent architecture of [3], we assume our agent A_u is situated in an environment E containing the opponent agent A_o. As in [3], at any time t our agent A_u senses the current state S_t of E, but unlike [3] the state now is explicitly represented as a set of internal state attributes (see Fig. 1, purple box). We use the internal state attributes to estimate the threshold utility below which no bid should be accepted/offered from/to the opponent agent [3]. These state attributes include information derived from the sequence of previous bids offered by A_o (e.g., utility of the most recently received bid from the opponent ω_t^o, utility of the best opponent bid so far O_{best}, average utility of all the opponent bids O_{avg} and their variability O_{sd}) and information stored in A_u's knowledge base (e.g., number of bids B in the given partial order, d_D, u_{res}, Ω, and n), and the current negotiation time t. This internal state representation, denoted with s_t, is used by A_u (in acceptance and bidding strategies) to decide what action a_t to execute from the set of *Actions* based on the negotiation protocol P at time t. Action execution then changes the state of the environment to S_{t+1}. For the acceptance strategy, s_t requires additional state attributes as follows: fixed target utility u, dynamic and learnable target utility $\bar{u}_t$, utility $U(\omega)$ of received bid ω w.r.t. U, q quantile value which changes w.r.t time t, and quantile function $Q_{\widehat{U}(\Omega_t^o)}(q)$. Similarly, for the bidding strategy, s_t requires the additional state attributes as

Proc. of the 21st International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2022), P. Faliszewski, V. Mascardi, C. Pelachaud, M.E. Taylor (eds.), May 9–13, 2022, Online.

follows: $b_{Boulware}$, *PS* Pareto-optimal bid, $b_{opp}(\omega_t^o)$ a bid by greedily manipulating the last bid received from the opponent ω_t^o, and a random bid from the uniform distribution above $\bar{u}_t$.

The action a_t is derived via two functions, f_a and f_b, for the acceptance and bidding strategies, respectively, as in [3]. The function f_a takes as inputs s_t, a *dynamic threshold utility* $\bar{u}_t$, the sequence of past opponent bids Ω_t^o, and outputs a discrete action a_t among *accept* or *reject*. When f_a returns *reject*, f_b computes what to bid next, with input s_t and $\bar{u}_t$, see (1–2). This separation of acceptance and bidding strategies is not rare, see for instance [1]. Also, f_a and f_b consist of a set of tactics as defined in [3].

$$f_a(s_t, \bar{u}_t, \Omega_t^o) = a_t, a_t \in \{accept, reject\} \quad (1)$$

$$f_b(s_t, \bar{u}_t, \Omega_t^o) = a_t, a_t \in \{offer(\omega), \omega \in \Omega\} \quad (2)$$

We assume incomplete opponent preference information, therefore, *Decide* uses the estimated model $\widehat{U}_o$. In particular, $\widehat{U}_o$ is estimated at time t using information from Ω_t^o. Unlike [3], our *Decide* component employs DRL in both the Acceptance as well as the Bidding Strategy Templates Parameters, in addition to Threshold Utility (see the three green coloured boxes in Fig. 1). Each DRL component is based on the actor-critic architecture [6] and has its own *Evaluate* and *Negotiation Experience* components as in [2, 3].

Evaluate refers to a critic helping our agent learn the dynamic threshold utility $\bar{u}_t$, acceptance template parameters and bidding template parameters, with the new experience collected against each opponent. This is a function of random K ($K < N$) experiences fetched from the agent's memory. Here, learning is *retrospective*, since it depends on the reward r_t obtained from E by performing a_t at s_t. The reward values for every critic that are used for estimating the threshold utility (i.e., $r_t^{\bar{u}_t}$) as well as choice parameter values of acceptance (i.e., r_t^{bid}) and bidding templates (i.e., r_t^{acc}) depend on the discounted user utility of the last opponent ω_t^o, or of the accepted bid ω^{acc} and defined as (3), (4) and (5) respectively.

$$r_t^{\bar{u}_t} = \begin{cases} U_u(\omega^{acc}, t), & \text{on agreement} \\ U_u(\omega_t^o, t), & \text{on received offer} \\ -1, & \text{otherwise.} \end{cases} \quad (3)$$

$$r_t^{bid} = \begin{cases} U_u(\omega^{acc}, t), & \text{on agreement} \\ -1, & \text{otherwise.} \end{cases} \quad (4)$$

$$r_t^{acc} = \begin{cases} U_u(\omega^{acc}, t), & \text{on agreement and } U_o(\omega^{acc}, t) \leq U_u(\omega^{acc}, t) \\ U_u(\omega_t^o, t), & \text{on rejection and } U_o(\omega_t^o, t) \geq U_u(\omega_t^o, t) \\ -1, & \text{otherwise.} \end{cases} \quad (5)$$

$r_t^{\bar{u}_t}$ (3) and r_t^{bid} (4) are straight-forward. In (5), $U_o(\omega, t)$ is used as the reward value because the reward is received from the environment E where the opponent agent resides. In other words, we assume that E has access to A_o's real preferences, i.e., U_o, but these preferences are not observable by our agent A_u. The first case of the r_t^{acc} deals with an agreed bid and returns a positive reward value, if the bid gives higher utility to our agent than the opponent. The second case deals with a rejected bid and returns a positive reward value, if the bid gives lower utility to our agent than the opponent. In all other cases, it returns a negative value. Also, in (3), (4) and (5), $U_u(\omega, t)$ is the discounted reward of ω defined as (6).

$$U_u(\omega, t) = U_u(\omega) \cdot (d)^t, d \in [0, 1] \quad (6)$$

In (6), d is a temporal discount factor to encourage the agent to negotiate without delay. We should not confuse d, which is typically unknown to the agent, with the discount factor used to compute the utility of an agreed bid (d_D).

Negotiation Experience stores historical information about N previous interactions of an agent with other agents. Experience elements are of the form $\langle s_t, a_t, r_t, s_{t+1} \rangle$, where s_t is the internal state representation of the negotiation environment E, a_t is the performed action, r_t is a scalar *reward* received from the environment and s_{t+1} is the new agent state after executing a_t.

3 STRATEGY TEMPLATES

We consider the same admissible tactics as [3]. The key difference is that our approach evolves the entire strategy (within the space of strategies entailed by the template) at every negotiation, which makes it more adaptable and generalizable. Below, we give an example of an acceptance strategy learned with our model in one domain (*Party*) and we show how the strategy adapts in the other domain (*Grocery*) against the opponent strategy [3].

(a) Party Domain

$$t \in [0.000, 0.0361) \rightarrow U_u(\omega_t^o) \geq \max\left(Q_{U_{\Omega_t^o}}(-0.20 \cdot t + 0.22), \bar{u}_t\right)$$

$$t \in [0.0361, 1.000] \rightarrow U_u(\omega_t^o) \geq \max\left(u, Q_{U_{\Omega_t^o}}(-0.10 \cdot t + 0.64)\right)$$

(b) Grocery Domain

$$t \in [0.000, 0.2164) \rightarrow U_u(\omega_t^o) \geq \max\left(U_u(\omega_t), Q_{U_{\Omega_t^o}}(-0.55 \cdot t + 0.05), \bar{u}_t\right)$$

$$t \in [0.2164, 0.3379) \rightarrow U_u(\omega_t^o) \geq \max\left(U_u(\omega_t), Q_{U_{\Omega_t^o}}(-0.60 \cdot t + 1.40)\right)$$

$$t \in [0.3379, 1.000] \rightarrow U_u(\omega_t^o) \geq \max\left(Q_{U_{\Omega_t^o}}(-0.22 \cdot t + 0.29), \bar{u}_t\right)$$

Observe that the duration learned in the left-hand side of the tactics is different for different domains, e.g., initially in the *Party* domain, the first rule triggers when $t \in [0.0, 0.0361)$, while in the *Grocery* domain the first rule triggers at $t \in [0.0, 0.2164)$ Similarly, for the parameters on the right-hand side, e.g., in the *Party* domain, during the very early phase of the negotiation, the strategy uses a quantile tactic and dynamic threshold utility. However, in the *Grocery* domain, the strategy now employs future bid utility along with the quantile bid and the dynamic threshold utility. The experimental evaluation of this work is available in the full version of this paper [4].

4 CONCLUSIONS

We have used DRL based on an actor-critic architecture to support negotiation in domains with multiple issues. In particular, we have exploited "interpretable" strategy templates used in the state-of-the-art to learn the best combination of acceptance and bidding tactics at any negotiation time. All tactics, including an adaptive threshold utility, are learned using the DRL [5] algorithm deriving an initial neural network strategy via supervised learning.

REFERENCES

[1] Tim Baarslag, Koen Hindriks, Mark Hendrikx, Alexander Dirkzwager, and Catholijn Jonker. 2014. Decoupling negotiating agents to explore the space of negotiation strategies. In *Novel Insights in Agent-based Complex Automated Negotiation*. Springer, 61–83.

[2] Pallavi Bagga, Nicola Paoletti, Bedour Alrayes, and Kostas Stathis. 2020. A Deep Reinforcement Learning Approach to Concurrent Bilateral Negotiation. In *IJCAI*.

[3] Pallavi Bagga, Nicola Paoletti, and Kostas Stathis. 2020. Learnable strategies for bilateral agent negotiation over multiple issues. *arXiv preprint arXiv:2009.08302* (2020).

[4] Pallavi Bagga, Nicola Paoletti, and Kostas Stathis. 2022. Deep Learnable Strategy Templates for Multi-Issue Bilateral Negotiation. arXiv:2201.02455 [cs.MA]

[5] Timothy Paul Lillicrap, Jonathan James Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. 2016. Continuous control with deep reinforcement learning. In *Proceedings of the 4th International Conference on Learning Representations (ICLR 2016)*.

[6] Richard S Sutton and Andrew G Barto. 2018. *Reinforcement learning: An introduction*. MIT press.