Argumentation-Based Reinforcement Learning for RoboCup Soccer Takeaway

(Extended Abstract)

Yang Gao
Imperial College London, UK
yg211@imperial.ac.uk

Francesca Toni
Imperial College London, UK
ft@imperial.ac.uk

ABSTRACT

Reinforcement Learning (RL) is widely regarded as a generic and effective technique to learn coordinated behaviours in cooperative multi-agent systems (CMAS), but it suffers from slow convergence speed due to the huge joint action space. Incorporating domain knowledge has shown to be an effective method to tackle this problem, but little research has investigated how to propose high-quality heuristics. We consider a widely used CMAS application, RoboCup Takeaway, and use value-based argumentation to extract heuristics from conflicting domain knowledge therein.

1. INTRODUCTION

Reinforcement Learning (RL), which enables agents to learn optimal behaviour by interacting with the environment, has been regarded as an effective machine learning technique to achieve coordinated behaviours in cooperative multi-agent systems (CMAS) [1, 2]. However, RL can be very slow in CMAS, mainly because of the huge joint action space which is exponential in the number of agents [1]. An effective methodology to tackle this problem is to integrate domain knowledge into RL so as to reduce the exploration time [3]. However, even though it has been reported that the effectiveness of this methodology is sensitive to the quality of the heuristics, little research has been devoted to investigate how to facilitate people to extract high-quality heuristics from conflicting domain knowledge. In this work, we consider a widely used CMAS: RoboCup Takeaway, and build a variant of value-based argumentation frameworks [4] to solve the conflicts within the domain knowledge so as to provide heuristics to achieve coordinated behaviours without searching the joint action space.

2. ABSTRACT AND VALUE-BASED ARGUMENTATION FRAMEWORKS

An abstract argumentation framework (AF) is a pair (Arg, Att) where Arg is a set of arguments and Att ⊆ Arg × Arg is a binary relation ((A, B) ∈ Att is read ‘A attacks B’). Suppose S ⊆ Arg and B ∈ Arg and some member of S attacks B. S is conflict-free iff S attacks none of its members. S defends B iff S attacks all arguments attacking B. Semantics of AFs are defined as sets of "rationally acceptable" arguments, known as extensions. For example, given some F = (Arg, Att), S ⊆ Arg is an admissible extension for F iff S is conflict-free and defends all its elements; S is a complete extension for F iff S is conflict-free and S = {a|S defends a}; S is the grounded extension for F iff S is minimally (wrt. ⊆) complete for F. The (possibly empty) grounded extension is guaranteed to be unique, consisting solely of the uncontroversial arguments and being thus “sceptical”.

In some contexts, the attack relation between arguments is not enough to decide what is “rationally acceptable”, and the “values” promoted by arguments must be considered. Value-based argumentation frameworks (VAFs) [4] incorporate values and preferences over them into AFs. Their key idea is to allow for attacks to succeed or fail, depending on the relative worth of the values promoted by the competing arguments. Given a set V of values, an audience Valpref is a strict partial order over V (corresponding to the preferences of an agent), and an audience-specific VAF is a tuple (Arg, Att, V, Valpref), where (Arg, Att) is an AF and Val : Arg → V gives the values promoted by arguments. In VAF, the ordering over values, Valpref, is taken into account in the definition of extensions. The simplification of an audience-specific VAF is the AF (Arg, Att−), where (A, B) ∈ Att− iff (A, B) ∈ Att and Val(B) is not higher than Val(A) in Valpref. (A, B) ∈ Att− is read ‘A defeats B’. Then, (acceptable) extensions of a VAF are defined as (acceptable) extensions of its simplification (Arg, Att−). We refer to (Arg, Att−) as the simplified AF derived from (Arg, Att, V, Valpref).

3. MOTIVATING EXAMPLE: ROBOCUP TAKEAWAY GAME

The Takeaway game is proposed by [5]. In a N-Takeaway game, N + 1 (N ∈ N, N ≥ 1) hand-coded keepers are competing with N learning takers on a fixed-size field. Keepers attempt to keep possession of the ball, whereas takers attempt to win possession of the ball. Since only takers are learning in Takeaway, their learning task is to win possession of the ball as fast as possible.

Iscen and Eroglu [5] proposed two macro actions for takers:

- **TackleBall:** move directly towards the ball to tackle it
- **MarkKeeper(i):** go to mark keeper Ki, i ≠ 1

where Ki represents the i-th closest keeper to the ball (so that K1 is the keeper in possession of the ball). When a taker marks a keeper, the taker blocks the path between the ball and that keeper. Thus, a taker is not allowed to mark the ball holder, and in N-Takeaway, each taker can choose among M = N + 1 actions.

The observation of each taker is represented by a state vector. We use the same state vector as in our previous work [6].

Consider a scenario in 2-Takeaway as illustrated in Fig 1(a). We may propose the following advice: (a) T1 should tackle the ball, because it is closest to the ball; (b) T2 should mark K3, because it is closest to K3; and (c) T3 should mark K3, because the angle between K3 and T1, with vertex at K1, is smallest. Even only consid-
ering these three recommendations, we can see that there exist both internal conflicts and external conflicts: item (a) and (e) internally conflict with one another because they are both recommendations for $T_1$, but suggest $T_1$ to perform different actions; item (b) and (e) externally conflict with one another because they are recommendations for different agents, but suggest them to perform the same action, which, in Takeaway, is believed wasteful in terms of efficiency. We are going to use value-based argumentation frameworks to solve these conflicts, as shown next.

4. ARGUMENTATION FRAMEWORK FOR TAKEAWAY GAMES

In line with Section 3, we give the following domain knowledge for any taker $T_i$, $i \in \{1, \ldots, N\}$:

1. $T_1$ should tackle the ball if it is closest to the ball holder;
2. If the angle between $T_1$ and a keeper, with vertex at the ball holder, is the smallest, $T_1$ should mark this keeper;
3. If $T_1$ is closest to a keeper, $T_1$ should mark this keeper.

Note that this knowledge is action-based, i.e. recommending actions to agents. Given state variables (Table 2 in [6]) we “translate” the knowledge above into 3 categories of candidate arguments:

1. $T_1$ TackleBall() IF $i = \arg\min_{1 \leq t \leq N} \text{dist}(T_1 K_1, K_2)$
2. $T_1$ MarkKeeper($j$) IF $i = \arg\min_{1 \leq t \leq N} \text{ang}(K_j, T_i)$
3. $T_1$ MarkKeeper($j$) IF $i = \arg\min_{1 \leq t \leq N} \text{dist}(K_j, T_i)$

where $j \in \{2, \ldots, N+1\}$ for arguments referred to as $T_1$ A($j$) and $T_1$ C($j$), because $K_1$ cannot be marked. Overall, for a $N$-Takeaway game, there are $2N^2 + N$ candidate arguments. We will use Arg* to denote the set of candidate arguments, and given these candidate arguments, we build argumentation framework as follows:

For taker $T_i$, $T_i$ TK gives one argument and the other two categories of arguments each give $N$ (as there are $N$ free keepers to be marked). So there are $N \times (2 \times N + 1)$ candidate arguments in total.

Remark. To build a SCAF, we first select the applicable arguments, namely the arguments whose premises are true in this scenario. Then we build attacks between these applicable arguments to represent the conflicts between the domain knowledge. Item 2(a) and item 2(b) models the internal and external conflicts, resp.

SCAF models the conflicts within domain knowledge, but does not solve them. To this end, we incorporate values into SCAF:

Definition 2. Given a SCAF ($\text{Arg, Att}$), a value-based scenario specific cooperative argumentation framework (VSCAF) is a tuple ($\text{Arg, Att, V, val, Valpref}$) s.t.:

1. $V$ is a set of values
2. val : Arg* $\rightarrow$ V is a function from Arg* to V
3. Valpref is a strict partial order over V

We denote val($A$) = $v$ as $A \rightarrow v$, and say that $A$ promotes $v$. Given values and their preferences, a simplified AF can be derived from a VSCAF, as in standard VAF. We denote the new set of attacks as Att−, and let $AF^{-} = (A, Att^{-})$ be this simplified AF derived from VSCAF. Then we compute the grounded extension of AF− to obtain the “rationally accepted” arguments, and these arguments can be used as heuristics to instruct agents.

As a concrete example, we build the SCAF for the scenario depicted in Fig. 1(a). First, we compute the applicability of each candidate arguments. To illustrate, we first consider argument $T_1$ TK. Since $T_1$ is closest to the ball, this argument is applicable. For the same reason, we can see that $T_2$ TK is not applicable. The applicability of other candidate arguments can be decided similarly. We then build attack relationships (Att) between these applicable arguments. To illustrate Att, consider $T_1$ TK and $T_1$ A(3): they are both applicable for $T_1$ but recommend different actions, so they attack each other. Consider also $T_1$ A(3) and $T_1$ C(3): they are applicable for different agents but recommend the same action, so they attack each other. The resulting $SCAF$ is depicted in Fig. 1(b).

Then we incorporate values into the SCAF to build a VSCAF and derive its simplified AF. We propose the following values:

1. VT: Prevent the ball being held by the keepers;
2. VA: Ensure that each pass can be quickly intercepted;
3. VC: Ensure that, after each pass, the ball can be quickly tackled. The mapping from arguments to values (val) is defined as follows: $T_1$ TK $\rightarrow$ VT, $T_1$ A(1) $\rightarrow$ VA, $T_1$ C(1) $\rightarrow$ VC. Further, according to our domain knowledge, let VT $\succ$ VA $\equiv$ VC. Given these values and their rankings, we can obtain the simplified AF, as illustrated in Fig 1(c). Its grounded extension is $(T_1$ TK, $T_1$ A(2), $T_1$ C(2))$), so we will recommend $T_1$ to tackle the ball and recommend $T_2$ to mark keeper $K_2$.

5. REFERENCES