EdInferno.2D
Team Description Paper for RoboCup 2011
2D Soccer Simulation League

Majd Hawasly and Subramanian Ramamoorthy

Institute of Perception, Action and Behaviour
School of Informatics, The University of Edinburgh
Edinburgh, United Kingdom
M.Hawasly@sms.ed.ac.uk

Abstract. In this description document, we outline the main ideas behind our entry to the RoboCup 2011 2D Soccer Simulation League competition. This simulated soccer team, EdInferno.2D, represents our debut in this league. The major research issue that drives our effort is that of cooperative sequential decision making and online learning in environments that continually change. We outline a framework for high-level decision making and discuss how this is implemented in a fully functional simulated team.

1 Introduction

EdInferno.2D is competing for the first time in RoboCup 2011. The team is the outcome of research within the Robust Autonomy and Decisions group[1], led by Dr. Subramanian Ramamoorthy, at the Institute of Perception, Action and Behaviour in the School of Informatics, at The University of Edinburgh. Research within the RAD group is primarily concerned with autonomous decision making over time and under uncertainty by single or multiple agents interacting with a continually changing world, typically involving large state/action spaces, strategic objectives and sophisticated task specifications/constraints. This 2D soccer team is part of a larger RoboCup effort within the RAD group, including a new SPL team, EdInferno, which has already secured its place in international competitions at RoboCup 2011. The objective in developing a 2D simulation team is to study the space of problems arising from the multiagent platform of Soccer Server which enables exploration of cooperative decision making under uncertainty and in the presence of adversaries, in a way that goes beyond the physical robot leagues that are still constrained by the hardware and low-level systems issues. The potential for strategic sophistication in the simulation platform makes it a suitable test bed for our research.

[1] http://wcms.inf.ed.ac.uk/ipab/autonomy

2 Scientific focus

We come from a solid background in machine learning and reinforcement learning for robotic systems. Our research interest lies in the problem of learning by interaction within large, somewhat arbitrarily evolving, worlds using resource limited agents; and especially in the problem of cooperative decision making within such environments. A number of different methodologies have been developed to tackle change in decision problems, which we view as a central feature in domains such as soccer. Usually, change is posed as perturbation to an underlying stationary process that governs the decision problem [3, 6, 7]. This assumption, however, is not always valid. In multiagent systems particularly, small changes in behaviour of individual players (usually due to learning) could cause large deviations from a global perspective. Trying to model this in terms of a stationary process is not always possible.

Moreover, in adversarial situations, it's essential to discover and exploit opportunities and limitations of the unknown adversary. The conservative method of behaving for a worst-case scenario or using canned reactive behaviours would usually hinder performance. Normally, this is tackled by utilising a bank of complete or partial plans that the team can choose from online, but this does not produce real robustness because it only covers situations anticipated by the designers, not all the ones that the team can potentially handle. Low level online planning and coordination in dynamic partially-observable environments, on the other hand, is not generally applicable to interesting decision problems of sizes that render the computation prohibitive [4, 5].

Our approach to this problem is based on online synthesis (in the spirit of [2]) of robust team *capabilities*. Capabilities are limited multiagent tasks with outcomes that are robust locally; i.e. that encode what the team is able do within some coarse situation (context or *team state*) in response to noise and variation up to some robustness threshold. Capabilities work to maintain or transform between team states, and they are small enough so that they can be scripted or learnt offline. Using this representation, we propose a collaborative algorithm that uses only local communication to generate online plans that achieve an arbitrary goal, and admits scalability without explosion in computational complexity.

3 Framework

3.1 Definitions

A team state is *active* if the system satisfies its *predicate*. We assume that tasks can be described using a suitably sparse set of team states, and the goal of the task can be described by a proper subset. From each team state, few *local games* are available. A local game is a multiagent controller that robustly moves the system between team states. Using team states and local games, multi-robot domains can be described using a *domain graph*: nodes represent the team states, and the links represent the games. Note that the domain graph is task

independent. One or more nodes (team states) are usually active at any time, ignoring transient effects. The objective of the team is to drive the system, through activation of links (local games), into the region of goal states. An *active agent* is an agent who maintains an active state while an *inactive agent* has maintainable states that are all inactive. Note that inactive agents still contribute significantly to team performance.

3.2 Application: soccer offence

Team states The team states that define this domain are the holding states H_i; $i : 1, 2, ..., 11$; each representing the corresponding player actively holding the ball, *independent of physical location*. L corresponds to the ball being stolen by the other team. In addition to these, the absorbing state GO is the state of scoring a goal.

Local games The local games that encode the team capabilities are: `hold` (a single-agent behaviour for holding the ball and preventing other players from taking it, a maintainability game for H_i,); `dribble` (a single-agent behaviour for dribbling, another maintainability game for H_i,); `pass` (a passing behaviour that involves the ball holder and one other team mate. This behaviour links any two different holding states); `get ball` (a behaviour that makes the player seek the ball and try to bring it into possession. This behaviour links L to the three holding states); and `shoot` (a single-agent behaviour that links H_i states to GO.)

Domain graph The ideal domain graph of the offence task is shown in Figure 1 (states and games of only two players are shown for clarity). The domain graph is a dynamic structure; the links are updated online and can become broken or new ones may get created. For example, if one of the players is facing a tough defender, its `hold` behaviour will eventually lead to state L.

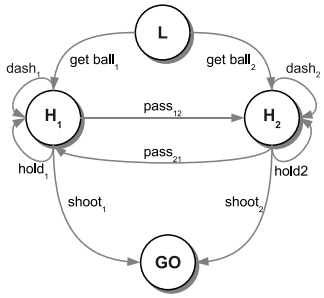

Fig. 1. Domain graph for 2-player soccer offence. The nodes represent the team states, and the links are the robust local games that encode team capabilities.

Task The offence task can be described as achieving state GO and avoiding state L. The agent holding the ball is an active agent, choosing to hold ball,

dribble, pass or shoot. Other players are inactive agents who act to optimise the task accomplishment.

3.3 Algorithm

Notation

- Team state $x = \{s \in S \mid F_x(s) = true\}$ where F_x is a boolean function over the state space, $F_x : S \to \{0, 1\}$, and X is the team state space.
- Local game g: $S_g \times A \times \Psi_g \to [0, 1]$, is a policy over $S_g \subseteq S$, joint actions A and the uncharacterised exogenous dynamics Ψ_g. S_g is the subset of state space where g is applicable, and G is the collection of all games. $G_x = \{g \in G \mid x \subseteq S_g\}$ is the collection of games available at team state x.
- Domain graph DG is the tuple $\langle X, RB \rangle$ where nodes are $x \in X$, and $RB : X \times G \times X \times S \to [0, 1]$ is the robustness matrix, where $RB(x_1, g, x_2) = f(s)$ is a measure of certainty, under state s, that local game $g \in G_{x_1}$ will succeed in moving the system to x_2 starting from x_1.
- *Value* of team state $V : X \times T \to \mathcal{R}$, where $V(x, t)$ is the robustness, at time t and under state s^t, of the best path from x to any of the goal states.

Values and sequential decision making The value of a team state measures how reliably the goal can be reached starting from that state, and decisions are taken on the basis of local optimisation of value. The value function can change arbitrarily reflecting the changes in the environment/adversary and the estimated reliability of local games. The goal states have a fixed value of 1, and losing states have a fixed value of 0. The value of any other team state is calculated using the estimated robustness of outbound links and reached values:

$$V(x, t) = \max_{g \in G_x,\, x' \in X} [RB(x, g, x')(s^t) * V(x', t - 1)] \tag{1}$$

The value reflects the goodness of the full path from any state to some goal state, up to the collective knowledge of the team.

Local communication In a changing environment, any plan needs continuous revision. Local communication is used to forge the complete value process as it changes. Every agent is responsible for estimating the values of a small subset of team states and communicating them to neighbouring agents. Local updates arrive quickly, while farther ones require more time to disperse in the graph, expressing the nature of locality in multi-robot interaction.

Adaptation to changing environments Robustness of links is learnt offline in standard situations. To account for unknown dynamics and adversarial strategies, continuous estimation of link robustness by the means of online learning is utilised. This allows the system to adapt to the changing dynamics. Robustness is estimated as the game's probability of success under the obtaining state.

Currently, a simple online learning rule is implemented. When the system moves from state x_1 to x_2 under the game g, the robustness of that outcome is reinforced and normalised:

$$RB(x_1, g, x_2)(s) = \frac{RB(x_1, g, x_2)(s) + \alpha}{\sum_{x' \in X} RB(x_1, g, x')(s) + \alpha} \tag{2}$$

$$RB(x_1, g, x)(s) = \frac{RB(x_1, g, x)(s)}{\sum_{x' \in X} RB(x_1, g, x')(s) + \alpha}; x \neq x_2 \tag{3}$$

Global collaboration Inactive agents act to enhance the flow of value in the graph. That is, they, alternatively, optimise the robustness of the links that connect them to active states on one hand (e.g., ball holder), and to the high-valued states (e.g., goal state) on the other hand. Achieved using local communication only, this is a form of team-level collaboration based on common values, beyond the local collaboration that happens inside the games.

Algorithm 1 EdInferno.2D high-level decision making

for $t = 1 \ldots$ till the end of the task,

1. agent computes values of its states:
   $V(x, t) = \max_{g \in G_x,\, x' \in X} [RB(x, g, x')(s^t) * V(x', t-1)]$
2. agent broadcasts values to state inbound neighbours:
   $N_x^{in} = \{x' \in X | RB(x', ., x)(.) > threshold\}$, for all x.
3. Inactive agent optimises the links from active states: $\arg\max_{g \in G_x} [RB(\tilde{x}, g, x)]$, $\tilde{x} \in N_x^{in} \cap \tilde{X}^t$, and to best neighbours: $\arg\max_{g \in G_x} [RB(x, g, x^*)]$.
4. Active agent chooses the next game by the optimisation:
   $\widehat{g} = \arg\max_{g \in G_x} [RB(x, g, \widehat{x_g})(s^t) * V(\widehat{x_g}, t-1)]$
   where,
   $\widehat{x_g} = \arg\max_{x' \in X} [RB(x, g, x')(s^t)]$, for all $g \in G_x$
5. agent plays its role for all games under way.
6. If the agent's game terminates, robustness estimate is updated:
   $RB(x_1, g, x_2)(s) = (RB(x_1, g, x_2)(s) + \alpha) / (\sum_{x' \in X} RB(x_1, g, x')(s) + \alpha)$
   $RB(x_1, g, x)(s) = RB(x_1, g, x)(s) / (\sum_{x' \in X} RB(x_1, g, x')(s) + \alpha);\ x \neq x_2$

4 EdInferno.2D structure

The team builds on agent2d base code [1], which is freely available under LGPL. The framework uses only the basic modules offered by agent2d and Librcsc as low level controllers and communication primitives, and learns how to synthesise useful plans using them. The positioning control of agent2d is currently included, but will later be replaced with the inactive agents optimisation on the domain graph. The structure of the team, and how that relates to agent2d, are shown in Figure 2.

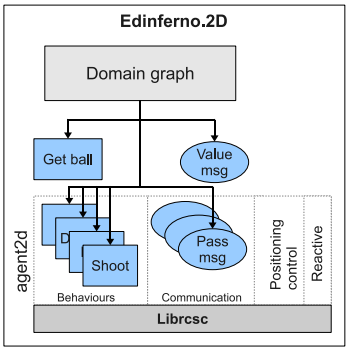

Fig. 2. Structure of EdInferno.2D.

5 Future Work

In addition to offence, the defensive task is to be modelled using the framework. Internally, inactive agents optimisation of value flow in the graph will eventually replace the heuristic position control of agent2d in the implementation. Achieving better coordination inside local games requires exploiting other forms of local communication, like pointing, beside the verbal communication which is used for global coordination. Finally, the coach global view of the decision problem can be utilised to change the structure of the domain graph (e.g., by enforcing links) or the value process (e.g., by enforcing nodes) to rectify team strategy.

References

1. agent2d; http://rctools.sourceforge.jp/pukiwiki/index.php?agent2d.
2. R.R. Burridge, A.A. Rizzi, and D.E. Koditschek. Sequential composition of dynamically dexterous robot behaviors. *The International Journal of Robotics Research*, 18(6):534, 1999.
3. C. Guestrin, D. Koller, and R. Parr. Multiagent planning with factored MDPs. *Advances in Neural Information Processing Systems*, 2:1523–1530, 2002.
4. E.A. Hansen, D.S. Bernstein, and S. Zilberstein. Dynamic programming for partially observable stochastic games. In *Proceedings of the National Conference on Artificial Intelligence*, pages 709–715. Menlo Park, CA; Cambridge, MA; London; AAAI Press; MIT Press; 1999, 2004.
5. A. Kumar and S. Zilberstein. Dynamic Programming Approximations for Partially Observable Stochastic Games. In *Proc. of the 22nd International FLAIRS Conference*, 2009.
6. A. Nilim and L.E. Ghaoui. Robust control of Markov decision processes with uncertain transition matrices. *Operations Research*, 53(5):780–798, 2005.
7. Jia Yuan Yu and Shie Mannor. Online learning in markov decision processes with arbitrarily changing rewards and transitions. In *Proceedings of the First ICST international conference on Game Theory for Networks*, GameNets'09, pages 314–322, Piscataway, NJ, USA, 2009. IEEE Press.