Cooperating with Unknown Teammates in Robot Soccer

Samuel Barrett Peter Stone

University of Texas at Austin
{sbarrett,pstone}@cs.utexas.edu

MIPC Workshop
July 28, 2014

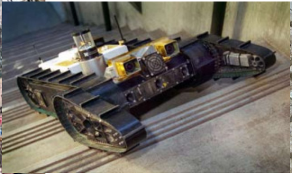
Disclaimer

- Added results from paper
- Different planning algorithm
- Still ongoing work

Example

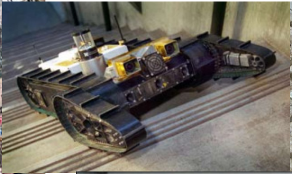
Example

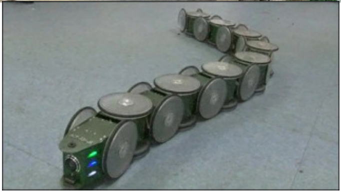
Example



LARG
Learning Agents Research Group
The University of Texas at Austin

Example





Example

Learning Agents Research Group
The University of Texas at Austin

Ad Hoc Teamwork

- Only in control of a single agent
- Unknown teammates
- Shared goals
- No pre-coordination

Examples in humans:

- Pick up soccer
- Accident response

Learning Agents Research Group
The University of Texas at Austin

Motivation

- Agents are becoming more common and lasting longer
  - Both robots and software agents
- Pre-coordination may not be possible
- Agents should be robust to various teammates
- Need to adapt quickly!

Motivation

- Agents are becoming more common and lasting longer
  - Both robots and software agents
- Pre-coordination may not be possible
- Agents should be robust to various teammates
- Need to adapt quickly!

Research Question:

How can an agent cooperate with unknown teammates to play robot soccer?

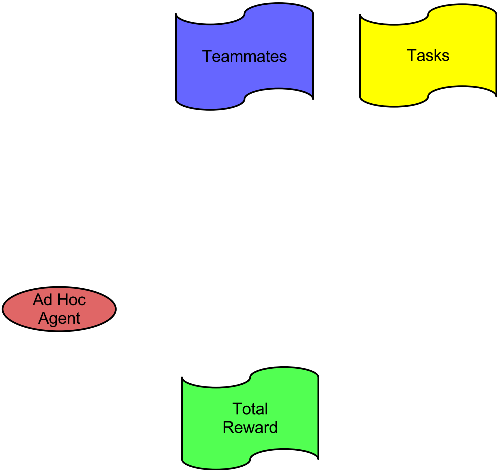
Learning Agents Research Group
The University of Texas at Austin

Ad Hoc Agent Evaluation

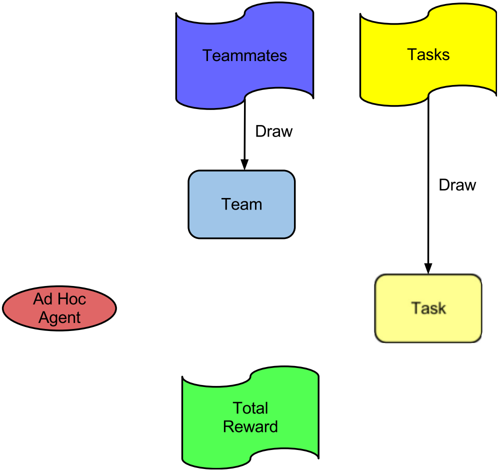
- Can the ad hoc agent replace any teammate on the team?
- Compare against other ad hoc agents
- Depends on possible tasks
- Depends on possible teammates

LARG
Learning Agents Research Group
The University of Texas at Austin

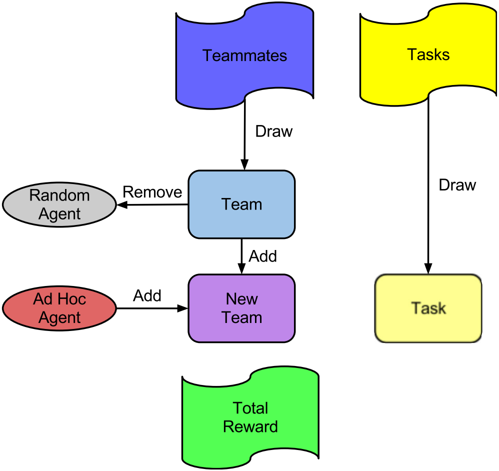
Ad Hoc Agent Evaluation



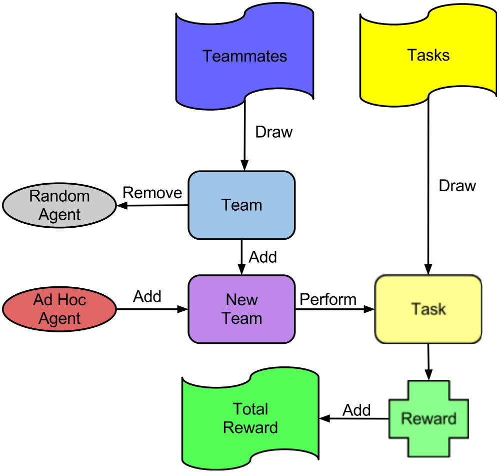
Ad Hoc Agent Evaluation



Ad Hoc Agent Evaluation



Ad Hoc Agent Evaluation



Half Field Offense

- 4 offensive players
- 5 defensive players
- Noisy observations and actuators
- Offense tries to score
- Defense are Helios agents
- Episode ends when:
  - Score
  - Ball leaves half field
  - Ball captured by defense

LARG
Learning Agents Research Group
The University of Texas at Austin

Overview

- Build on Helios code release (*agent2d*)
- Model as Markov Decision Process (MDP)
- Learn to cooperate with past teammates
- Select policies online

Learning Agents Research Group
The University of Texas at Austin

MDP Formulation

$$\text{MDP} = \langle S, A, P, R \rangle$$

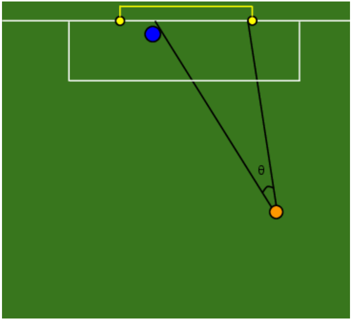
- S = State
- A = Actions
- P = transition function
- R = reward function

MDP Formulation

$$\text{MDP} = \langle S, A, P, R \rangle$$

State:

- Agent's x,y position and orientation
- Agent's goal opening angle
- Teammate's goal opening angle
- Distance to opponent
- Distance from teammate to opponent
- Pass opening angle
- Distance to teammate



MDP Formulation

$$\text{MDP} = \langle S, A, P, R \rangle$$

Actions with ball:

- High level
- Select from a many options using hand-coded evaluation
- 6 actions:
    - Shoot
    - Short dribble
    - Long dribble
    - Pass x3

MDP Formulation

$$\text{MDP} = \langle S, A, P, R \rangle$$

Actions away from ball:

- 7 actions:
  - Stay still
  - Towards ball
  - Towards goal
  - Towards teammate
  - Away from teammate
  - Towards opponent
  - Away from opponent

MDP Formulation

$$\text{MDP} = \langle S, A, P, R \rangle$$

Transition function:

- Gives resulting state after taking an action
- Given by the 2D RoboCup simulator
- Not explicitly modeled

MDP Formulation

$$\text{MDP} = \langle S, A, P, R \rangle$$

Reward function:

- Describes the value of a state
- 1,000 on win
- -1,000 on loss
- -1 per step

Collect Data

- Collect $\langle s, a, r, s' \rangle$
  - s – original state
  - a – action
  - r – reward
  - s' – next state
- In parallel

Q-Learning

- Iterate through stored experiences
- Estimates values of state-actions
- Update rule:
  $Q(s, a) = Q(s, a) + \alpha[r + \gamma \max'_a Q(s', a') - Q(s, a)]$

Q-Learning

- Iterate through stored experiences
- Estimates values of state-actions
- Update rule:
  $Q(s, a) = Q(s, a) + \alpha[r + \gamma \max'_a Q(s', a') - Q(s, a)]$
- Use Q(λ)
  - Incorporates elligibility traces
  - Aids in credit assignment

Function Approximation

- Generalize known values to neighbors
- Handles continuous space

Function Approximation

- Generalize known values to neighbors
- Handles continuous space
- CMAC Tile coding
  - $\hat{Q}(s, a) = \sum_i w_i f_i$
  - Perform parameter search over parameters

Learning Agents Research Group
The University of Texas at Austin

Different Teammates

- Learning a single policy does not work well for all teammates
- Instead, learn policy for each teammate

Different Teammates

- Learning a single policy does not work well for all teammates
- Instead, learn policy for each teammate
- Question: how do we know which policy to use for an unknown teammate?

How to select?

- Can treat as a multi-armed bandit problem
  - Slow to learn
  - One pull = 1 game of HFO

How to select?

- Can treat as a multi-armed bandit problem
  - Slow to learn
  - One pull = 1 game of HFO
- Compare observed teammate actions to past experiences
  - Normally distributed
  - Update using Bayes rule
  - Bound loss

Teammates

- Externally-created teammates
- Total of 7 teammate types

Teammates

- Externally-created teammates
- Total of 7 teammate types
- 6 of top 8 teams from the 2013 competition
  - aut
  - axiom
  - cyrus
  - gliders
  - helios
  - yushan
- Plus the agent2d code release

LARG
Learning Agents Research Group
The University of Texas at Austin

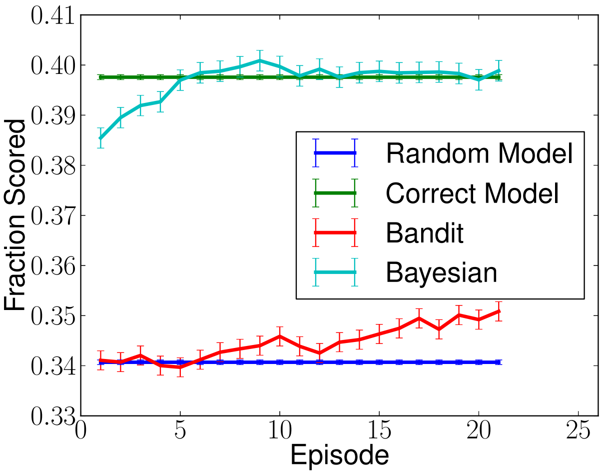
Setup

- 10,000 trials
- Randomly selected teammate
- Teammate is unknown to agent
- Baselines:
  - Randomly select a policy to use
  - Select the policy learned for that teammate

Problem Description

- Limited version of the game
- 2 offensive players
  - 1 teammate
- 2 defensive players

Results for 2v2



Results for 2v2

- Knowing your teammate's behavior helps
- Bandit algorithm is slow for this setting
  - 7 arms
  - Only 25 pulls

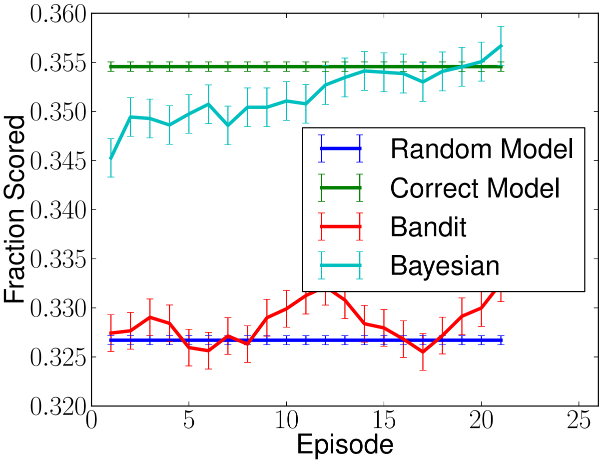
Results for 2v2

- Knowing your teammate's behavior helps
- Bandit algorithm is slow for this setting
  - 7 arms
  - Only 25 pulls
- Bayesian approach learns quickly
  - Outperforms bandit significantly

Problem Description

- 4 offensive players
  - 3 teammates
- 5 defensive players
- Harder task

Results for 4v5



Results for 4v5

- Less improvement by learning policy
  - Only 1 of 9 agents in play
- Slower to learn than 2v2 case
  - More noise from more agents

Learning Agents Research Group
The University of Texas at Austin

Results for 4v5

- Less improvement by learning policy
  - Only 1 of 9 agents in play
- Slower to learn than 2v2 case
  - More noise from more agents
- Can learn which policy to use
- Bayesian approach outperforms bandit significantly

Conclusions

- Handle a complex domain
- Can learn policy to cooperate with teammates

Conclusions

- Handle a complex domain
- Can learn policy to cooperate with teammates
- Can figure out how to cooperate with unknown teammates on the fly

LARG
Learning Agents Research Group
The University of Texas at Austin

Related Work

- P. Stone and S. Kraus. To teach or not to teach? Decision making under uncertainty in ad hoc teams. In *AAMAS '10*, May 2010
- P. Stone, G. A. Kaminka, and J. S. Rosenschein. Leading a best-response teammate in an ad hoc team. In *AMEC*. November 2010
- F. Wu, S. Zilberstein, and X. Chen. Online planning for ad hoc autonomous agent teams. In *IJCAI*, 2011

Related Work

- S. Liemhetcharat and M. Veloso. Modeling mutual capabilities in heterogeneous teams for role assignment. In *IROS '11*, pages 3638 –3644, 2011
- M. Bowling and P. McCracken. Coordination and adaptation in impromptu teams. In *AAAI*, pages 53–58, 2005
- J. Han, M. Li, and L. Guo. Soft control on collective behavior of a group of autonomous agents by a shill agent. *Journal of Systems Science and Complexity*, 19:54–62, 2006

Thank You!

- Can learn to cooperate with different teammates in the 2D RoboCup domain