Good Afternoon, Colleagues

Are there any questions?
Logistics

- Executable teams due next Tuesday
Logistics

- Executable teams due next Tuesday
- Final reports due on Thursday
Logistics

• Executable teams due next Tuesday

• Final reports due on Thursday

• Final tournament: Wednesday, May 7th, 10am, TAY 3.128
Machine Learning

**Hypothesis space:** set of possible functions

**Training examples:** the data

**Learning method:** training examples $\mapsto$ hypothesis
Machine Learning

**Hypothesis space:** set of possible functions

**Training examples:** the data

**Learning method:** training examples $\mapsto$ hypothesis

Agent Learning

**Policy:** how to **act** (generate training examples)
Machine Learning

**Hypothesis space:** set of possible functions

**Training examples:** the data

**Learning method:** training examples $\mapsto$ hypothesis

Agent Learning

**Policy:** how to act (generate training examples)

neural network training, decision tree training, clustering, genetic algorithms, genetic programming, reinforcement learning...
3 vs. 2 Keepaway (joint with Rich Sutton)

- **Play in a small area** (20m × 20m)
- **Keepers** try to keep the ball
- **Takers** try to get the ball

**Episode:**
- Players and ball reset randomly
- Ball starts near a keeper
- Ends when taker gets the ball or ball goes out

**Performance measure:** *average possession duration*

**Use CMUnited-99 skills:**
- HoldBall, PassBall($k$), GoToBall, GetOpen
Available Skills (from CMUnited-99)

**HoldBall()**: Remain stationary while keeping possession of the ball.

**PassBall(k)**: Kick the ball directly to keeper $k$.

**GoToBall()**: Intercept a moving ball or move directly towards a stationary ball.

**GetOpen()**: Move to a position that is free from opponents and open for a pass from the ball’s current position (using SPAR (Veloso et al., 1999))

**BlockPass(k)**: Get in between the ball and keeper $k$
The Keepers’ Policy Space

- GetOpen
- Ball not kickable
- Ball kickable
- GoToBall
- \{HoldBall, PassBall(k)\}
  (k is another keeper)

Teammate with ball or can get there faster
The Keepers’ Policy Space

- Teammate with ball or can get there faster
- GetOpen
- Ball not kickable
- GoToBall
- {HoldBall, PassBall(k)}
  (k is another keeper)

Example Policies

**Random:** HoldBall or PassBall(k) randomly

**Hold:** Always HoldBall

**Hand-coded:**

- If no taker within 10m: HoldBall
- Else If there’s a good pass: PassBall(k)
- Else HoldBall
Mapping Keepaway to RL

**Discrete-time, episodic, distributed RL**

- Simulator operates in discrete time steps, $t = 0, 1, 2, \ldots$, each representing 100 msec

- **Episode:** $s_0, a_0, r_1, s_1, \ldots, s_t, a_t, r_{t+1}, s_{t+1}, \ldots, r_T, s_T$

- $a_t \in \{\text{HoldBall}, \text{PassBall}(k), \text{GoToBall}, \text{GetOpen}\}$

- $r_t = 1$

- $V^\pi(s) = E\{T \mid s_0 = s\}$

- **Goal:** Find $\pi^*$ that maximizes $V$ for all $s$
Representation

Full soccer state

Huge binary feature vector
(about 400 1’s and 40,000 0’s)

Sparse, coarse, tile coding

Few continuous state variables
(13)

Linear map
Action values
$s$: 13 Continuous State Variables

- 11 distances among players, ball, and center
- 2 angles to takers along passing lanes
Function Approximation: Tile Coding

- Form of sparse, coarse coding based on CMACS (Albus, 1981)

- Tiled state variables individually (13)
Policy Learning

• Learn $Q^\pi(s, a)$: Expected possession time
Policy Learning

- Learn $Q^\pi(s, a)$: Expected possession time
- Linear Sarsa($\lambda$) — each agent learns independently
  - On-policy method: advantages over e.g. Q-learning
  - Not known to converge, but works (e.g. (Sutton, 1996))
Main Result

1 hour = 720 5-second episodes
## Varied Field Size

<table>
<thead>
<tr>
<th>Keepers</th>
<th>Testing Field Size</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>15x15</td>
</tr>
<tr>
<td>Trained on field</td>
<td>15x15</td>
</tr>
<tr>
<td>of size</td>
<td>20x20</td>
</tr>
<tr>
<td>25x25</td>
<td>6.3</td>
</tr>
<tr>
<td>Benchmarks</td>
<td>Hand</td>
</tr>
<tr>
<td></td>
<td>Hold</td>
</tr>
<tr>
<td></td>
<td>Random</td>
</tr>
</tbody>
</table>

- Single runs
- Learning specific to fields
  - Mechanism generalizes better than policies
4 vs. 3 Keeper Learning

- Preliminary: taker learning successful as well
What’s new in Keepaway?

• 5 vs. 4
What’s new in Keepaway?

- 5 vs. 4
- Transfer learning (Taylor, Liu)
What’s new in Keepaway?

• 5 vs. 4

• Transfer learning (Taylor, Liu)

• Evolutionary learning (Taylor and Whiteson)
What’s new in Keepaway?

• 5 vs. 4

• Transfer learning (Taylor, Liu)

• Evolutionary learning (Taylor and Whiteson)

• Half field offense (Kalyanakrishnan)
What’s new in Keepaway?

- 5 vs. 4
- Transfer learning (Taylor, Liu)
- Evolutionary learning (Taylor and Whiteson)
- Half field offense (Kalyanakrishnan)
  - Communication updates when others have the ball
What’s new in Keepaway?

- 5 vs. 4
- Transfer learning (Taylor, Liu)
- Evolutionary learning (Taylor and Whiteson)
- Half field offense (Kalyanakrishnan)
  - Communication updates when others have the ball
- Any coevolution?
Genetic algorithms

- Keep a population of individuals

- Each generation
  - Evaluate their fitness
  - Throw out the bad ones
  - Change the good ones randomly
  - Repeat
Genetic algorithms

- Keep a population of individuals

- Each generation
  - Evaluate their fitness
  - Throw out the bad ones
  - Change the good ones randomly
  - Repeat

The fitness function matters

- Playing against top-notch competition → no info

- Playing against a single foe → too brittle
Rosin and Belew

- Co-evolve 2 populations: gives software and test suites
  - item “New genotypes arise to defeat old ones”
- Why not self play?
Rosin and Belew

- Co-evolve 2 populations: gives software and test suites item “New genotypes arise to defeat old ones”
  - Why not self play?

- 2 techniques to keep diversity
Rosin and Belew

- Co-evolve 2 populations: gives software and test suites
  item “New genotypes arise to defeat old ones”
  - Why not self play?

- 2 techniques to keep diversity
  - Fitness sharing: prevent extinctions
  - Opponent sampling: use range of opponents to test
Rosin and Belew

- Co-evolve 2 populations: gives software and test suites
  - New genotypes arise to defeat old ones
  - Why not self play?

- 2 techniques to keep diversity
  - Fitness sharing: prevent extinctions
  - Opponent sampling: use range of opponents to test

- Test on TTT, Nim (and go)
  - Millions of generations
  - Worse than perfect play
Rosin and Belew

- Co-evolve 2 populations: gives software and test suites
  item “New genotypes arise to defeat old ones”
  - Why not self play?

- 2 techniques to keep diversity
  - Fitness sharing: prevent extinctions
  - Opponent sampling: use range of opponents to test

- Test on TTT, Nim (and go)
  - Millions of generations
  - Worse than perfect play
  - Why compare against old methods?
Collaborative Co-Evolution

- Learn collaborative behaviors simultaneously
Collaborative Co-Evolution

- Learn collaborative behaviors simultaneously
- Applied in pursuit domain among others
Collaborative Co-Evolution

- Learn **collaborative** behaviors simultaneously
- Applied in pursuit domain among others
- Simultaneous learning by teammates could be thought of in this way as well.