Reinforcement Learning for Sequential Decision Making

Peter Stone*

Director, Learning Agents Research Group (LARG)
Department of Computer Science
The University of Texas at Austin

Joint work with members of LARG past and present

Also, Cogitai Inc.
Who Am I?

To what degree can autonomous intelligent agents learn in the presence of teammates and/or adversaries in real-time, dynamic domains?
Who Am I?

To what degree can autonomous intelligent agents learn in the presence of teammates and/or adversaries in real-time, dynamic domains?

Research Areas

• Autonomous agents
• Robotics
• Multiagent systems
Who Am I?

To what degree can autonomous intelligent agents learn in the presence of teammates and/or adversaries in real-time, dynamic domains?

Research Areas
- Autonomous agents
- Robotics
- Multiagent systems
- Machine learning
Who Am I?

To what degree can autonomous intelligent agents learn in the presence of teammates and/or adversaries in real-time, dynamic domains?

Research Areas
- Autonomous agents
- Robotics
- Multiagent systems
- Machine learning
  - Reinforcement Learning
Who Am I?

To what degree can autonomous intelligent agents learn in the presence of teammates and/or adversaries in real-time, dynamic domains?

Research Areas
- Autonomous agents
- Robotics
- Multiagent systems
- Machine learning
  - Reinforcement Learning
AlphaGo [Silver et al., ’15]: An AI milestone
AlphaGo [Silver et al., ’15]: An AI milestone

- Integrated two **existing** AI technologies
AlphaGo [Silver et al., ’15]: An AI milestone

- Integrated two existing AI technologies
  - Supervised learning (Deep learning)
AlphaGo [Silver et al., ’15]: An AI milestone

- Integrated two **existing** AI technologies
  - Supervised learning (Deep learning)
  - **Reinforcement learning**
AlphaGo [Silver et al., ’15]: An AI milestone

- Integrated two existing AI technologies
  - Supervised learning (Deep learning)
  - Reinforcement learning
- Does not solve AI

Peter Stone (UT Austin) Reinforcement Learning 3
AlphaGo [Silver et al., ’15]: An AI milestone

- Integrated two existing AI technologies
  - Supervised learning (Deep learning)
  - Reinforcement learning
- Does not solve AI
  - The end of the line for:
AlphaGo [Silver et al., ’15]: An AI milestone

- Integrated two **existing** AI technologies
  - Supervised learning (Deep learning)
  - Reinforcement learning
- Does **not** solve AI
  - The end of the line for:

  **2-player zero-sum discrete finite deterministic games of perfect information**
A brief introduction to Go

- Black and white take turns to place down stones
- Once played, a stone cannot move
- The aim is to surround the most territory
- Usually played on 19x19 board
Capturing

- The lines radiating from a stone are called *liberties*
- If a connected group of stones has all of its liberties removed then it is captured
- Captured stones are removed from the board
Capturing

- The lines radiating from a stone are called *liberties*.
- If a connected group of stones has all of its liberties removed then it is captured.
- Captured stones are removed from the board.
2-player zero-sum discrete finite deterministic games of perfect information

What do these terms mean?

• **Two player:** Duh!

• **Zero-sum:** In any outcome of any game, Player A’s gains equal player B’s losses. ( Doesn’t mean fairness: “On average, two equal players will win or lose equal amounts” not necessary for zero-sum.)

• **Discrete:** All game states and decisions are discrete values.

• **Finite:** Only a finite number of states and decisions.

• **Deterministic:** No chance (no die rolls).

• **Games:** See next page

• **Perfect information:** Both players can see the state, and each decision is made sequentially (no simultaneous moves).

Slide created by Andrew Moore
Which of these are: 2-player zero-sum discrete finite deterministic games of perfect information

- Two player: Duh!
- Zero-sum: In any outcome of any game, Player A’s gains equal player B’s losses.
- Discrete: All game states and decisions are discrete values.
- Finite: Only a finite number of states and decisions.
- Deterministic: No chance (no die rolls).
- Games: See next page
- Perfect information: Both players can see the state, and each decision is made sequentially (no simultaneous moves).
Which of these are: 2-player zero-sum discrete finite deterministic games of perfect information

- Two player: Duh!
- Zero-sum: In any outcome of any game, Player A’s gains equal player B’s losses.
- Discrete: All game states and decisions are discrete values.
- Finite: Only a finite number of states and decisions.
- Deterministic: No chance (no die rolls).
- Games: See next page
- Perfect information: Both players can see the state, and each decision is made sequentially (no simultaneous moves).

Not finite
Stochastic
Hidden Information
Multiplayer
One player
Involves Improbable Animal Behavior

Slide created by Andrew Moore
Supervised Learning

Training Info = desired (target) outputs

Error = (target output - actual output)

Inputs  Supervised Learning System  Outputs
Reinforcement Learning

Training Info = evaluations ("rewards" / "penalties")

Inputs ➔ Reinforcement Learning System ➔ Outputs ("actions")

Objective: get as much reward as possible
Key Features of RL

- Learner is not told which actions to take
- Trial-and-Error search
- Possibility of delayed reward
  - Sacrifice short-term gains for greater long-term gains
- The need to *explore* and *exploit*
- Considers the whole problem of a goal-directed agent interacting with an uncertain environment
Reinforcement Learning

Supervised learning mature [WEKA]
Reinforcement Learning

Supervised learning mature [WEKA]

For agents, reinforcement learning most appropriate
Reinforcement Learning

Supervised learning mature [WEKA]

For agents, reinforcement learning most appropriate
Reinforcement Learning

Supervised learning mature [WEKA]

For agents, reinforcement learning most appropriate
Reinforcement Learning

Supervised learning mature [WEKA]

For agents, reinforcement learning most appropriate

- Foundational theoretical results
- Applications require innovations to scale up
RL Theory

Success story: **Q-learning** converges to $\pi^*$ [Watkins, 89]
RL Theory

Success story: Q-learning converges to $\pi^*$ [Watkins, 89]

- Table-based representation
- Visit every state infinitely often
Function Approximation

In practice, visiting every state impossible

\[ s[t-1], a[t-1] \]

\[ s[t], r[t] \]

\[ s, a \]

\[ Q(s,a) \]

\[ a[t] \]
Function Approximation

In practice, visiting every state impossible

Function approximation of value function

Theoretical guarantees harder to come by
In practice, often experience is scarce

Batch Methods
Batch Methods

In practice, often experience is scarce

\[ s[t-1], a[t-1], Q(s,a), r[t], a[t-1], s_a \]

Save transitions:

\[ <r[i], s[i], a[i]> \text{ for } i=0 \text{ to } t-1 \]
Applications: Towards a Useful Tool

- Backgammon [Tesauro, ’94]
- Helicopter control [Ng et al., ’03]
Applications: Towards a Useful Tool

- Backgammon [Tesauro, ’94]
- Helicopter control [Ng et al., ’03]

- Adaptive treatment of epilepsy [Pineau et al., ’08]

- Invasive species management, wildfire suppression [Dietterich et al., ’13]
Applications: Towards a Useful Tool

- Backgammon [Tesauro, ’94]
- Helicopter control [Ng et al., ’03]
- Adaptive treatment of epilepsy [Pineau et al., ’08]
- Invasive species management, wildfire suppression [Dietterich et al., ’13]
- Google DeepMind beats human go champion, [Silver et al., ’16]
Computer Go AI – Definition

\[ s \ (\text{state}) \]

\[ d = 1 \]

\[
\begin{pmatrix}
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\end{pmatrix}
\]

(e.g. we can represent the board into a matrix-like form)

\* The actual model uses other features than board positions as well
Computer Go AI – Definition

Given $s$, pick the best $a$
Computer Go AI – An Implementation Idea?

How about simulating all possible board positions?
Computer Go AI – An Implementation Idea?

d = 1  d = 2  d = 3
Computer Go AI – An Implementation Idea?

Process the simulation until the game ends, then report win / lose results.
Computer Go AI – An Implementation Idea?

Process the simulation until the game ends, then report win/lose results.

Example: it wins 13 times if the next stone gets placed here.

- 37,839 times
- 431,320 times

Choose the “next action/stone” that has the most win-counts in the full-scale simulation.
This is NOT possible; it is said the possible configurations of the board exceeds the number of atoms in the universe.
Key: To Reduce Search Space
Reducing Search Space

1. Reducing “action candidates” (Breadth Reduction)

IF there is a model that can tell you that these moves are not common / probable (e.g. by experts, etc.) ...
Reducing Search Space

1. Reducing “action candidates” (Breadth Reduction)

Remove these from search candidates in advance (breadth reduction)
Reducing Search Space

2. Position evaluation ahead of time (Depth Reduction)

Instead of simulating until the maximum depth ..
Reducing Search Space

2. Position evaluation ahead of time (Depth Reduction)

IF there is a function that can measure: $V(s)$: “board evaluation of state $s$”
Reducing Search Space

1. Reducing “action candidates” (Breadth Reduction)

2. Position evaluation ahead of time (Depth Reduction)
1. Reducing “action candidates”

Learning: $P(\text{next action} \mid \text{current state}) = P(a \mid s)$
1. Reducing “action candidates”

(1) Imitating expert moves (supervised learning)

Data: Online Go experts (5~9 dan)
160K games, 30M board positions
1. Reducing “action candidates”

(1) Imitating expert moves (supervised learning)
1. Reducing “action candidates”

(1) Imitating expert moves (supervised learning)

There are $19 \times 19 = 361$ possible actions (with different probabilities)
1. Reducing “action candidates”

(1) Imitating expert moves (supervised learning)
1. Reducing “action candidates”

(1) Imitating expert moves (supervised learning)

\[ s \rightarrow p(a|s) \rightarrow \text{argmax} \rightarrow a \]
1. Reducing “action candidates”

(1) Imitating expert moves (supervised learning)

$s \rightarrow \arg \max_{\text{Next Action}} p(a | s)$
1. Reducing “action candidates”

(1) Imitating expert moves (supervised learning)

| Current Board | 0 0 0 0 0 0 0 0 | 0 0 0 0 0 0 1 0 | 0 0 0 1 0 0 0 0 | 0 0 0 0 1 0 0 0 | 0 0 0 0 0 0 0 0 | 0 0 0 0 0 0 0 0 |
| Next Action | 0 0 0 0 0 0 0 0 | 0 0 0 0 0 0 0 0 | 0 0 0 0 0 0 0 0 | 0 0 0 0 0 0 0 0 | 0 0 0 0 0 0 0 0 | 0 0 0 0 0 0 0 0 |

\[ s \rightarrow p(a | s) \rightarrow \text{argmax} \rightarrow a \]

Deep Learning (13 Layer CNN)

\[ s \rightarrow g: s \rightarrow p(a | s) \rightarrow \text{argmax} \rightarrow a \]
**Go:** abstraction is the key to win

**CNN:** abstraction is its *forte*
1. Reducing “action candidates”

(1) Imitating expert moves (supervised learning)

```
Training:
\Delta \sigma \propto \frac{\partial \log p_\sigma(a|s)}{\partial \sigma}
```
1. Reducing “action candidates”

(2) Improving through self-plays (reinforcement learning)

Improving by playing against itself

Expert Moves
Imitator Model (w/ CNN) vs
Expert Moves
Imitator Model (w/ CNN)
1. Reducing “action candidates”

(2) Improving through self-plays (reinforcement learning)

Return: board positions, win/lose info
1. Reducing “action candidates”

(2) Improving through self-plays (reinforcement learning)

\[
\text{Board position} \quad \begin{array}{c}
\text{Expert Moves Imitator Model (w/ CNN)} \\
\text{(w/ CNN)}
\end{array} \quad \text{win/loss} \quad \text{Loss} \\
\text{Training: } \Delta \rho \propto \frac{\partial \log p_\rho(a_t|s_t)}{\partial \rho} z_t
\]

\[z = -1\]
1. Reducing “action candidates”

(2) Improving through self-plays (reinforcement learning)

Expert Moves Imitator Model (w/ CNN)

Training: \[ \Delta \rho \propto \frac{\partial \log p_{\rho}(a_t|s_t)}{\partial \rho} z_t \]

Board position

Win

Win

z = +1
1. Reducing “action candidates”

(2) Improving through self-plays (reinforcement learning)

Updated Model ver 1.1 vs. Updated Model ver 1.3

It uses the same topology as the expert moves imitator model, and just uses the updated parameters

Return: board positions, win/lose info
1. Reducing “action candidates”

(2) Improving through self-plays (reinforcement learning)

Updated Model ver 1.3 VS Updated Model ver 1.7

Return: board positions, win/lose info
1. Reducing “action candidates”

(2) Improving through self-plays (reinforcement learning)

Updated Model ver 1.5 vs Updated Model ver 2.0

Return: board positions, win/lose info
1. Reducing “action candidates”

(2) Improving through self-plays (reinforcement learning)

Updated Model ver 3204.1

Updated Model ver 46235.2

VS

Return: board positions, win/lose info
1. Reducing “action candidates”

(2) Improving through self-plays (reinforcement learning)

- Expert Moves
- Imitator Model

VS

- Updated Model
- ver 1,000,000

The final model wins 80% of the time when playing against the first model
2. Board Evaluation
2. Board Evaluation

Updated Model ver 1,000,000

Value Prediction Model (Regression)

Win / Loss

Win (0~1)

Board Position

Add a regression layer to the model
Predicts values between 0~1
Close to 1: a good board position
Close to 0: a bad board position

\[ \Delta \theta \propto \frac{\partial v_\theta(s)}{\partial \theta}(z - v_\theta(s)) \]
Reducing Search Space

1. Reducing “action candidates” (Breadth Reduction)
   - Policy Network

2. Board Evaluation (Depth Reduction)
   - Value Network
Looking ahead (w/ Monte Carlo Search Tree)

Action Candidates Reduction (Policy Network)

Board Evaluation (Value Network)

(Rollout): Faster version of estimating \( p(a | s) \)

\( \rightarrow \) uses shallow networks (3 ms \( \rightarrow \) 2\( \mu \)s)
Results

Elo rating system

Performance with different combinations of AlphaGo components
Takeaways

Use the networks trained for a certain task (with different loss objectives) for several other tasks.
AlphaGo [Silver et al., ’15]: An AI milestone

- Integrated two existing AI technologies
  - Supervised learning (Deep learning)
  - Reinforcement learning
- Does not solve AI
  - The end of the line for:

  2-player zero-sum discrete finite deterministic games of perfect information
Perception Uncertainty
Selected RL Contributions

- Human interaction
Selected RL Contributions

- Human interaction
  - Advice, Demonstration
Selected RL Contributions

- Human interaction
  - Advice, Demonstration
  - Positive/Negative Feedback

[Knox & Stone, ’09]
Selected RL Contributions

• Human interaction
  – Advice, Demonstration
  – Positive/Negative Feedback

• Transfer learning for RL

[Knox & Stone, ’09]
[Taylor & Stone, ’07]
Selected RL Contributions

- Human interaction
  - Advice, Demonstration
  - Positive/Negative Feedback
  [Knox & Stone, ’09]

- Transfer learning for RL
  [Taylor & Stone, ’07]

- Adaptive/hierarchical representations
  [Whiteson & Stone, ’05], [Jong & Stone, ’08]
Selected RL Contributions

- Human interaction
  - Advice, Demonstration
  - Positive/Negative Feedback
  [Knox & Stone, ’09]

- Transfer learning for RL
  [Taylor & Stone, ’07]

- Adaptive/hierarchical representations
  [Whiteson & Stone, ’05], [Jong & Stone, ’08]

- TEXPLORE for Robot RL
  [Hester & Stone, ’13]
Selected RL Contributions

- Human interaction
  - Advice, Demonstration
  - Positive/Negative Feedback

- Transfer learning for RL

- Adaptive/hierarchical representations

- **TEXPLORE** for Robot RL
  - Sample efficient; real-time

[Knox & Stone, ’09]
[Taylor & Stone, ’07]

[Whiteson & Stone, ’05], [Jong & Stone, ’08]

[Hester & Stone, ’13]
Selected RL Contributions

- Human interaction
  - Advice, Demonstration
  - Positive/Negative Feedback

- Transfer learning for RL

- Adaptive/hierarchical representations

- TEXPLORE for Robot RL
  - Sample efficient; real-time
  - Continuous state; delayed effects

[Knox & Stone, ’09]
[Taylor & Stone, ’07]
[Whiteson & Stone, ’05], [Jong & Stone, ’08]

[Hester & Stone, ’13]
UT Austin Villa 2014
RoboCup 3D Simulation League Champion via Overlapping Layered Learning

Patrick MacAlpine, Mike Depinet, and Peter Stone
Layered Learning

- For domains too complex for tractably mapping state features $S \mapsto O$
- Hierarchical subtask decomposition given: $\{L_1, L_2, \ldots, L_n\}$
Layered Learning

- For domains too complex for tractably mapping state features $S \mapsto O$
- Hierarchical subtask decomposition given: $\{L_1, L_2, \ldots, L_n\}$
- Machine learning: exploit data to train, adapt
Layered Learning

- For domains too complex for tractably mapping state features $S \rightarrow$ outputs $O$
- Hierarchical subtask decomposition given: $\{L_1, L_2, \ldots, L_n\}$
- Machine learning: exploit data to train, adapt
- **Synthesis:** Learning in one layer feeds into next layer
Layered Learning

- For domains too complex for tractably mapping state features $S \rightarrow$ outputs $O$
- Hierarchical subtask decomposition given: $\{L_1, L_2, \ldots, L_n\}$
- Machine learning: exploit data to train, adapt
- Synthesis: Learning in one layer feeds into next layer

Diagram:
- High Level Goals
- Adversarial Behaviors
- Team Behaviors
- Multi-Agent Behaviors
- Individual Behaviors
- World State
- Environment
- Machine Learning Opportunities
Layered Learning in Practice

First applied in *simulated* robot soccer [Stone & Veloso, ’97]

<table>
<thead>
<tr>
<th></th>
<th>Strategic Level</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>$L_1$</td>
<td>individual</td>
<td>ball interception</td>
</tr>
<tr>
<td>$L_2$</td>
<td>multiagent</td>
<td>pass evaluation</td>
</tr>
<tr>
<td>$L_3$</td>
<td><strong>team</strong></td>
<td>pass selection</td>
</tr>
</tbody>
</table>
Layered Learning in Practice

First applied in *simulated* robot soccer [Stone & Veloso, ’97]

<table>
<thead>
<tr>
<th>Strategic Level</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>$L_1$</td>
<td>individual ball interception</td>
</tr>
<tr>
<td>$L_2$</td>
<td>multiagent pass evaluation</td>
</tr>
<tr>
<td>$L_3$</td>
<td><em>team</em> pass selection</td>
</tr>
</tbody>
</table>

Later applied on *real robots* [Stone, Kohl, & Fidelman, ’06]

<table>
<thead>
<tr>
<th>Strategic Level</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>$L_1$</td>
<td>individual fast walking</td>
</tr>
<tr>
<td>$L_2$</td>
<td>individual ball control</td>
</tr>
</tbody>
</table>
Layered Learning Paradigms

DESCRIPTIONS:

**Sequential Layered Learning:** Freeze parameters of layer after learning before learning of the next layer

**Concurrent Layered Learning:** Keep parameters of layer open during learning of the next layer
**Layered Learning Paradigms**

**PROBLEMS:**

**Sequential Layered Learning:** Can be too limiting in the joint layer policy search space

**Concurrent Layered Learning:** The increased dimensionality can make learning harder or intractible
Layered Learning Paradigms

SOLUTION:

**Overlapping Layered Learning:** Tradeoff between freezing or keeping open previous learned layers

Optimizes “seam” or overlap between behaviors: keeps some parts of previously learned layers open during subsequent learning
Overlapping Layered Learning

Sequential Layered Learning (SLL)

Concurrent Layered Learning (CLL)

Overlapping Layered Learning

Combining Independently Learned Behaviors (CILB)

Partial Concurrent Layered Learning (PCLL)

Previous Learned Layer Refinement (PLLRR)
Overlapping Layered Learning

Combining Independently Learned Behaviors: Behaviors learned independently and then combined by relearning subset of behaviors’ parameters
Combining Independently Learned Behaviors: Behaviors learned independently and then combined by relearning subset of behaviors’ parameters

Partial Concurrent Layered Learning: Part, but not all, of a previously learned layer’s behaviors are left open
Overlapping Layered Learning

Combining Independently Learned Behaviors: Behaviors learned independently and then combined by relearning subset of behaviors’ parameters

Partial Concurrent Layered Learning: Part, but not all, of a previously learned layer’s behaviors are left open

Previous Learned Layer Refinement: After a pair of layers is learned, part or all of the initial layer is unfrozen
RoboCup 3D Simulation Domain

- Teams of 11 vs 11 autonomous robots play soccer
- Realistic physics using Open Dynamics Engine (ODE)
- Simulated robots modeled after Aldebaron Nao robot
- Robot receives noisy visual information about environment
- Robots can communicate over limited bandwidth channel
RoboCup Champions 2011, 2012
RoboCup Champions 2011, 2012
Humanoid Walk Learning via Layered Learning and CMA-ES
  • Parameterized double linear inverted pendulum model
RoboCup Champions 2011, 2012

Humanoid Walk Learning via Layered Learning and CMA-ES

- Parameterized *double linear inverted pendulum model*

- Stochastic, derivative-free, numerical optimization method
- Candidates sampled from *multidimensional Gaussian*

CMA-ES
[Hansen, ’09]
RoboCup Champions 2011, 2012

Humanoid Walk Learning via Layered Learning and CMA-ES

- Parameterized double linear inverted pendulum model

CMA-ES
[Hayanne, ’09]

- Stochastic, derivative-free, numerical optimization method
- Candidates sampled from multidimensional Gaussian
  - Mean maximizes likelihood of previous successes
  - Covariance update controls search step sizes

Peter Stone (UT Austin) Reinforcement Learning
RoboCup Champions 2011, 2012

Humanoid Walk Learning via Layered Learning and CMA-ES

- Parameterized **double linear inverted pendulum model**

CMA-ES
[Hansen, ’09]

- Stochastic, derivative-free, numerical optimization method
- Candidates sampled from multidimensional Gaussian
  - Mean maximizes likelihood of previous successes
  - Covariance update controls search step sizes

<table>
<thead>
<tr>
<th>Initial walk</th>
<th>No layered learning</th>
<th>2 layers</th>
</tr>
</thead>
<tbody>
<tr>
<td>3 layers</td>
<td>Final walk</td>
<td>Champs*2</td>
</tr>
</tbody>
</table>
Learned Layers

- **19 learned behaviors** for standing up, walking, and kicking
  - CILB, PCLL, PLLR
- **Over 500 parameters optimized** during the course of learning
  - frozen, passed, seeded
Dribbling and Kicking the Ball in the Goal

- Four different walk parameter sets
  - Target/sprint/position + **approach ball to kick**
- Learn **fixed kick**
- Combine **kick with walk**: combine independent layers (CILB)
  - Overlap kick parameters for positioning
- Final **walk and kick**
Scoring on a Kickoff

- Kickoffs **indirect** (2 players must touch to score)
- Learn **fixed kick**
- Learn **touch** behavior **interferes**
- Combine **kick with touch**
  - Relearn position patterns: combine independent layers (CILB)
  - Learn new timing parameter: partial concurrent (PCLL)

Peter Stone (UT Austin) Reinforcement Learning 26
Impact of Overlapping Layered Learning

1000 games vs. top 3 teams from 2013
Impact of Overlapping Layered Learning

1000 games vs. top 3 teams from 2013

<table>
<thead>
<tr>
<th>Opponent</th>
<th>Full Team</th>
<th>No Kickoff</th>
<th>Dribble Only</th>
</tr>
</thead>
<tbody>
<tr>
<td>apollo3d</td>
<td>2.703 (0.041)</td>
<td>2.062 (0.038)</td>
<td>1.861 (0.034)</td>
</tr>
<tr>
<td>UTAustinVilla2013</td>
<td>1.589 (0.036)</td>
<td>1.225 (0.033)</td>
<td>0.849 (0.025)</td>
</tr>
<tr>
<td>fcportugal3d</td>
<td>3.991 (0.051)</td>
<td>3.189 (0.048)</td>
<td>1.584 (0.030)</td>
</tr>
</tbody>
</table>

No Kickoff: On kickoff, kick ball deep into opponent’s end
Dribble Only: No kicking
Repetition on Different Robot Types

Type 0: Standard Nao model
Type 1: Longer legs and arms
Type 2: Quicker moving legs
Type 3: Wider hips and longest legs and arms
Type 4: Added toes to foot
Repetition on Different Robot Types

Type 0: Standard Nao model
Type 1: Longer legs and arms
Type 2: Quicker moving legs
Type 3: Wider hips and longest legs and arms
Type 4: Added toes to foot

<table>
<thead>
<tr>
<th>Opponent</th>
<th>Type 0</th>
<th>Type 1</th>
<th>Type 2</th>
<th>Type 3</th>
<th>Type 4</th>
</tr>
</thead>
<tbody>
<tr>
<td>apollo3d</td>
<td>1.787</td>
<td>1.819</td>
<td>1.820</td>
<td>1.543</td>
<td>2.827</td>
</tr>
<tr>
<td>UTAustinVilla2013</td>
<td>0.992</td>
<td>0.892</td>
<td>1.276</td>
<td>0.573</td>
<td>1.141</td>
</tr>
<tr>
<td>fcportugal3d</td>
<td>2.423</td>
<td>3.025</td>
<td>3.275</td>
<td>2.678</td>
<td>4.033</td>
</tr>
</tbody>
</table>
Repetition on Different Robot Types

Type 0: Standard Nao model
Type 1: Longer legs and arms
Type 2: Quicker moving legs
Type 3: Wider hips and longest legs and arms
Type 4: Added toes to foot

<table>
<thead>
<tr>
<th>Opponent</th>
<th>Type 0</th>
<th>Type 1</th>
<th>Type 2</th>
<th>Type 3</th>
<th>Type 4</th>
</tr>
</thead>
<tbody>
<tr>
<td>apollo3d</td>
<td>1.787</td>
<td>1.819</td>
<td>1.820</td>
<td>1.543</td>
<td>2.827</td>
</tr>
<tr>
<td>UTAustinVilla2013</td>
<td>0.992</td>
<td>0.892</td>
<td>1.276</td>
<td>0.573</td>
<td>1.141</td>
</tr>
<tr>
<td>fcportugal3d</td>
<td>2.423</td>
<td>3.025</td>
<td>3.275</td>
<td>2.678</td>
<td>4.033</td>
</tr>
</tbody>
</table>

Computation per type

\[ \approx 700k \text{ parameter sets evaluated} \]
\[ \approx 1.5 \text{ years compute time (} \approx 50 \text{ hours on condor cluster)} \]
RoboCup 2014

Won competition with **undefeated** record: outscored opps 52–0

<table>
<thead>
<tr>
<th>Opponent</th>
<th>Avg. Goal Diff.</th>
<th>Record (W-L-T)</th>
<th>Goals (F/A)</th>
<th>KO Score %</th>
</tr>
</thead>
<tbody>
<tr>
<td>BahiaRT</td>
<td>2.075 (0.030)</td>
<td>990-0-10</td>
<td>2092/17</td>
<td>96.2</td>
</tr>
<tr>
<td>FCPortugal</td>
<td>2.642 (0.034)</td>
<td>986-0-14</td>
<td>2748/106</td>
<td>83.4</td>
</tr>
<tr>
<td>magmaOffenburg</td>
<td>2.855 (0.035)</td>
<td>990-0-10</td>
<td>2864/9</td>
<td>88.3</td>
</tr>
<tr>
<td>RoboCanes</td>
<td>3.081 (0.046)</td>
<td>974-0-26</td>
<td>3155/74</td>
<td>69.4</td>
</tr>
<tr>
<td>FUT-K</td>
<td>3.236 (0.039)</td>
<td>998-0-2</td>
<td>3240/4</td>
<td>96.3</td>
</tr>
<tr>
<td>SEU_Jolly</td>
<td>4.031 (0.062)</td>
<td>995-0-5</td>
<td>4034/3</td>
<td>87.6</td>
</tr>
<tr>
<td>KarachiKoalas</td>
<td>5.681 (0.046)</td>
<td>1000-0-0</td>
<td>5682/1</td>
<td>87.5</td>
</tr>
<tr>
<td>ODENS</td>
<td>7.933 (0.041)</td>
<td>1000-0-0</td>
<td>7933/0</td>
<td>92.1</td>
</tr>
<tr>
<td>HfutEngine</td>
<td>8.510 (0.050)</td>
<td>1000-0-0</td>
<td>8510/0</td>
<td>94.7</td>
</tr>
<tr>
<td>Mithras3D</td>
<td>8.897 (0.041)</td>
<td>1000-0-0</td>
<td>8897/0</td>
<td>90.4</td>
</tr>
<tr>
<td>L3M-SIM</td>
<td>9.304 (0.043)</td>
<td>1000-0-0</td>
<td>9304/0</td>
<td>93.7</td>
</tr>
</tbody>
</table>

Peter Stone (UT Austin) Reinforcement Learning
RoboCup 2014

Won competition with **undefeated** record: outscored opps 52–0

<table>
<thead>
<tr>
<th>Opponent</th>
<th>Avg. Goal Diff.</th>
<th>Record (W-L-T)</th>
<th>Goals (F/A)</th>
<th>KO Score %</th>
</tr>
</thead>
<tbody>
<tr>
<td>BahiaRT</td>
<td>2.075 (0.030)</td>
<td>990-0-10</td>
<td>2092/17</td>
<td>96.2</td>
</tr>
<tr>
<td>FCPortugal</td>
<td>2.642 (0.034)</td>
<td>986-0-14</td>
<td>2748/106</td>
<td>83.4</td>
</tr>
<tr>
<td>magmaOffenburg</td>
<td>2.855 (0.035)</td>
<td>990-0-10</td>
<td>2864/9</td>
<td>88.3</td>
</tr>
<tr>
<td>RoboCanes</td>
<td>3.081 (0.046)</td>
<td>974-0-26</td>
<td>3155/74</td>
<td>69.4</td>
</tr>
<tr>
<td>FUT-K</td>
<td>3.236 (0.039)</td>
<td>998-0-2</td>
<td>3240/4</td>
<td>96.3</td>
</tr>
<tr>
<td>SEU_Jolly</td>
<td>4.031 (0.062)</td>
<td>995-0-5</td>
<td>4034/3</td>
<td>87.6</td>
</tr>
<tr>
<td>KarachiKoalas</td>
<td>5.681 (0.046)</td>
<td>1000-0-0</td>
<td>5682/1</td>
<td>87.5</td>
</tr>
<tr>
<td>ODENS</td>
<td>7.933 (0.041)</td>
<td>1000-0-0</td>
<td>7933/0</td>
<td>92.1</td>
</tr>
<tr>
<td>HfutEngine</td>
<td>8.510 (0.050)</td>
<td>1000-0-0</td>
<td>8510/0</td>
<td>94.7</td>
</tr>
<tr>
<td>Mithras3D</td>
<td>8.897 (0.041)</td>
<td>1000-0-0</td>
<td>8897/0</td>
<td>90.4</td>
</tr>
<tr>
<td>L3M-SIM</td>
<td>9.304 (0.043)</td>
<td>1000-0-0</td>
<td>9304/0</td>
<td>93.7</td>
</tr>
</tbody>
</table>

- **After: 11,000 games:** won all by 67 (no losses)
RoboCup 2014

Won competition with **undefeated** record: outscored opps 52–0

<table>
<thead>
<tr>
<th>Opponent</th>
<th>Avg. Goal Diff.</th>
<th>Record (W-L-T)</th>
<th>Goals (F/A)</th>
<th>KO Score %</th>
</tr>
</thead>
<tbody>
<tr>
<td>BahiaRT</td>
<td>2.075 (0.030)</td>
<td>990-0-10</td>
<td>2092/17</td>
<td>96.2</td>
</tr>
<tr>
<td>FCPortugal</td>
<td>2.642 (0.034)</td>
<td>986-0-14</td>
<td>2748/106</td>
<td>83.4</td>
</tr>
<tr>
<td>magmaOffenburg</td>
<td>2.855 (0.035)</td>
<td>990-0-10</td>
<td>2864/9</td>
<td>88.3</td>
</tr>
<tr>
<td>RoboCanes</td>
<td>3.081 (0.046)</td>
<td>974-0-26</td>
<td>3155/74</td>
<td>69.4</td>
</tr>
<tr>
<td>FUT-K</td>
<td>3.236 (0.039)</td>
<td>998-0-2</td>
<td>3240/4</td>
<td>96.3</td>
</tr>
<tr>
<td>SEU_Jolly</td>
<td>4.031 (0.062)</td>
<td>995-0-5</td>
<td>4034/3</td>
<td>87.6</td>
</tr>
<tr>
<td>KarachiKoalas</td>
<td>5.681 (0.046)</td>
<td>1000-0-0</td>
<td>5682/1</td>
<td>87.5</td>
</tr>
<tr>
<td>ODENS</td>
<td>7.933 (0.041)</td>
<td>1000-0-0</td>
<td>7933/0</td>
<td>92.1</td>
</tr>
<tr>
<td>HfutEngine</td>
<td>8.510 (0.050)</td>
<td>1000-0-0</td>
<td>8510/0</td>
<td>94.7</td>
</tr>
<tr>
<td>Mithras3D</td>
<td>8.897 (0.041)</td>
<td>1000-0-0</td>
<td>8897/0</td>
<td>90.4</td>
</tr>
<tr>
<td>L3M-SIM</td>
<td>9.304 (0.043)</td>
<td>1000-0-0</td>
<td>9304/0</td>
<td>93.7</td>
</tr>
</tbody>
</table>

- **After:** **11,000 games:** won all by 67 (**no losses**)
- **Highlights** from Final vs. RoboCanes (University of Miami)
RoboCup 2014

Won competition with **undefeated** record: outscored opps 52–0

<table>
<thead>
<tr>
<th>Opponent</th>
<th>Avg. Goal Diff.</th>
<th>Record (W-L-T)</th>
<th>Goals (F/A)</th>
<th>KO Score %</th>
</tr>
</thead>
<tbody>
<tr>
<td>BahiaRT</td>
<td>2.075 (0.030)</td>
<td>990-0-10</td>
<td>2092/17</td>
<td>96.2</td>
</tr>
<tr>
<td>FCPortugal</td>
<td>2.642 (0.034)</td>
<td>986-0-14</td>
<td>2748/106</td>
<td>83.4</td>
</tr>
<tr>
<td>magmaOffenburg</td>
<td>2.855 (0.035)</td>
<td>990-0-10</td>
<td>2864/9</td>
<td>88.3</td>
</tr>
<tr>
<td>RoboCanes</td>
<td>3.081 (0.046)</td>
<td>974-0-26</td>
<td>3155/74</td>
<td>69.4</td>
</tr>
<tr>
<td>FUT-K</td>
<td>3.236 (0.039)</td>
<td>998-0-2</td>
<td>3240/4</td>
<td>96.3</td>
</tr>
<tr>
<td>SEU_Jolly</td>
<td>4.031 (0.062)</td>
<td>995-0-5</td>
<td>4034/3</td>
<td>87.6</td>
</tr>
<tr>
<td>KarachiKoalas</td>
<td>5.681 (0.046)</td>
<td>1000-0-0</td>
<td>5682/1</td>
<td>87.5</td>
</tr>
<tr>
<td>ODENS</td>
<td>7.933 (0.041)</td>
<td>1000-0-0</td>
<td>7933/0</td>
<td>92.1</td>
</tr>
<tr>
<td>HfutEngine</td>
<td>8.510 (0.050)</td>
<td>1000-0-0</td>
<td>8510/0</td>
<td>94.7</td>
</tr>
<tr>
<td>Mithras3D</td>
<td>8.897 (0.041)</td>
<td>1000-0-0</td>
<td>8897/0</td>
<td>90.4</td>
</tr>
<tr>
<td>L3M-SIM</td>
<td>9.304 (0.043)</td>
<td>1000-0-0</td>
<td>9304/0</td>
<td>93.7</td>
</tr>
</tbody>
</table>

- **After:** **11,000 games:** won all by 67 (no losses)
- **Highlights** from Final vs. RoboCanes (University of Miami)
- More info: www.cs.utexas.edu/~AustinVilla/sim/3dsimulation/
**Practical RL**

- **Representation**
  - Selecting the Algorithm: parameterized domains [K.&S., MLJ 2011]
  - Adapting Representation: NEAT+Q [Whiteson & S., JMLR 2006]

- **Interaction**
  - With adversaries: CMLEs [Chakraborty & S., ICML 2010]
  - With ad hoc teammates: PLASTIC [Barrett, thesis 2014]
  - With people: TAMER [Knox & S., AAMAS 2010]

- **Synthesis**

- **Mortality**
  - Leverage the Past: Transfer Learning [Taylor, S., & Liu, JMLR 2007]
  - Acknowledge a Finite Future: TEXPLORE [Hester & S., MLJ 2013]
Making Friends on the Fly: Advances in Ad Hoc Teamwork

Samuel Barrett, Katie Genter, and Peter Stone
Ad Hoc Teamwork [Stone et al., AIJ 2013]

- Only in control of a single agent or subset of agents
- Unknown teammates
- Shared goals
- No pre-coordination
Ad Hoc Teamwork [Stone et al., AIJ 2013]

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

Examples in humans:
- Pick up soccer
- Accident response
PLASTIC: Planning and Learning to Adapt Swiftly to Teammates to Improve Cooperation

Peter Stone (UT Austin) Reinforcement Learning 33
Testbed Domains

- Agent replaces single teammate in otherwise coherent team
- Adapts based on knowledge learned from previous teammates
Other Ad Hoc Teamwork

- Introduced as AAAI Challenge Problem [Stone et al. ’10]

Peter Stone (UT Austin) Reinforcement Learning 35
Other Ad Hoc Teamwork

- Introduced as AAAI Challenge Problem [Stone et al. ’10]
- Theory: repeated games, bandits [Stone et al., ’11]
Other Ad Hoc Teamwork

- Introduced as **AAAI Challenge Problem**  
  [Stone et al. ’10]
- Theory: repeated games, bandits  
  [Stone et al., ’11]
- Experiments: **flocking**  
  [Genter & Stone, ’12]
Other Ad Hoc Teamwork

- Introduced as AAAI Challenge Problem
- Theory: repeated games, bandits
- Experiments: flocking
- RoboCup experiments

[Stone et al. ’10] [Stone et al., ’11] [Genter & Stone, ’12] [Genter et al., ’15]
Other Ad Hoc Teamwork

- Introduced as **AAAI Challenge Problem** [Stone et al. ’10]
- Theory: repeated games, bandits [Stone et al., ’11]
- Experiments: **flocking** [Genter & Stone, ’12]
- **RoboCup experiments** [Genter et al., ’15]
- AAAI Workshops, JAAMAS special issue
Acknowledgements

Nicholas Jong
Matthew Taylor
Shimon Whiteson
Doran Chakraborty
Todd Hester
Shivaram Kalyanakrishnan
Brad Knox
Samuel Barrett
Patrick MacAlpine
Daniel Urieli
Katie Genter

LARG (past and present), NSF, ONR, FHWA, DARPA, IBM, Google