Section Outline

- Introduction to RL
- The dimensions of transfer
  - task relatedness
  - transferred knowledge
  - learning algorithms
- Transfer between tasks with same state-action variables
  - From one source task to one target task
  - From many source tasks to one target task
  - Multitask learning: Learning a distribution of tasks
- Transfer between tasks with different state-action variables
  - No explicit mapping
  - Mapping state variables and actions between tasks
  - Learning the inter-task mapping

Problem-Independent Learning

- Konidaris and Barto
  - “Autonomous shaping: Knowledge transfer in reinforcement learning,” 2006
- Agent Space
  - Possibly non-Markovian
  - Capabilities: physical sensors, actuators
- Problem Space
  - Markovian
  - May change between tasks

Transfer between tasks with different SxA

- Problems Arise
  - How to handle new features or actions?
  - Ignore?
  - What if ordering of features/actions change?

- The idea: reason over agent-space and problem-space
- Task difference: problem-space
- Transferred knowledge: shaping reward / options
- Learning algorithm: sarsa
- Metric: learning speed
Konidaris and Barto (2006, 2007)

- Learn shaping reward in Agent-Space
- Learn agent-space options

Transfer in RRL

"Learning relational options for inductive transfer in relational reinforcement learning," (Croonenborghs et al., 2007)

- Explicit mapping not necessarily needed
- Actions
  - move(A,B)
- State
  - clear(A)
  - above(A,X)

RRL Transfer

- The idea: use relational representation
- Task difference: # of objects
- Transferred knowledge: options
- Learning algorithm: TGR
- Metric: learning speed

Croonenborghs et al., 2007

Option Learning in Blocksworld

1. use an RRL algorithm to learn to solve this task
2. create training examples of state-action pairs labeled as policy-based or not
3. use the TILDE system to learn a relational decision tree that predicts whether or not the action will be executed by the policy
4. extract a relational policy

Option: clear(S, A)

Skill: if (clear(S, A)) then
  \{ (S, good, clear(X)): (clear(S, A), clear(X), above(S, A, X) -> move(A, B)) \}
Section Outline

- Introduction to RL
- The dimensions of transfer
  - task relatedness
  - transferred knowledge
  - learning algorithms
- Transfer between tasks with same state-action variables
  - From one source task to one target task
  - From many source tasks to one target task
  - Multitask learning: Learning a distribution of tasks
- Transfer between tasks with different state-action variables
  - No explicit mapping
  - Mapping state variables and actions between tasks
  - Learning the inter-task mapping

Value Function Transfer

- The idea: transfer Q-values using inter-task mappings
- Task difference: state features and actions
- Transferred knowledge: Q-values
- Learning algorithm: sarsa
- Metric: learning speed+

Inter-Task Mappings (Taylor and Stone, 2005)

- \( \chi_x: s_{\text{target}} \rightarrow s_{\text{source}} \)
  - Given state variable in target task (some \( x \) from \( s=x_1, x_2, \ldots x_n \))
  - Return corresponding state variable in source task
- \( \chi_A: a_{\text{target}} \rightarrow a_{\text{source}} \)
  - Similar, but for actions
- Intuitive mappings exist in some domains (Oracle)
- Used to construct transfer functional

Value Function Transfer

- “Transfer learning via inter-task mappings for temporal difference learning,” Taylor et al, 2007
- Q not defined on \( S' \) and \( A' \)
- \( \rho(Q(S,A)) = Q'(S',A') \)
- Action-Value function transferred
- \( \rho \) is task-dependant:
  - relies on inter-task mappings
Keepaway Hand-coded $\chi_A$

Actions in 4 vs. 3 have “similar” actions in 3 vs. 2

- $\text{Hold}_{4v3} \rightarrow \text{Hold}_{3v2}$
- $\text{Pass1}_{4v3} \rightarrow \text{Pass1}_{3v2}$
- $\text{Pass2}_{4v3} \rightarrow \text{Pass2}_{3v2}$
- $\text{Pass3}_{4v3} \rightarrow \text{Pass2}_{3v2}$

Value Function Transfer Flexibility

- Different Function Approximators
- Different Actuators
- Different Keepaway Tasks
- Partial Mappings

<table>
<thead>
<tr>
<th>Transfer Functional</th>
<th># 3 vs. 2 Episodes</th>
<th>Avg. 4 vs. 3 Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>none</td>
<td>0</td>
<td>30.84</td>
</tr>
<tr>
<td>Full</td>
<td>100</td>
<td>17.71</td>
</tr>
<tr>
<td>Full</td>
<td>3000</td>
<td>9.12</td>
</tr>
<tr>
<td>Partial</td>
<td>100</td>
<td>18.50</td>
</tr>
<tr>
<td>Partial</td>
<td>3000</td>
<td>12.00</td>
</tr>
</tbody>
</table>

Value Function Transfer: Time to threshold in 4 vs. 3

- **The idea**: learn and transfer strategies
- **Task difference**: state features and actions
- **Transferred knowledge**: options
- **Learning algorithm**: sarsa
- **Metric**: learning speed
Strategy Transfer

“Relational macros for transfer in reinforcement learning,” (Torrey et al., 2007)

1. Structure Learning
   – ID sequences of successful actions
   – Compose into Finite State Machine

2. Ruleset Learning
   – Learn when action in a strategy should be taken
   – Aleph

3. Remap strategies via human

4. DemonstrateStrategy

5. Sarsa(λ) with SVM function approximation

Torrey et al., 2007: Example

- Relational Macro Transfer via Demonstration (RMT-D)

- Macro learned in 2-on-1 Breakaway

- Demonstrated in 3-on-2 Breakaway

Cross Domain Transfer?

- Series of mazes with different goals [Fernandez and Veloso, 2006]
- Mazes with different structures [Konidaris and Barto, 2007]
Related Transfer Work

- Series of mazes with different goals [Fernandez and Veloso, 2006]
- Mazes with different structures [Konidaris and Barto, 2007]
- Keepaway with different numbers of players [Taylor and Stone, 2005]
- Keepaway to Breakaway [Maclin et al, 2005]

Source Task: Ringworld

- 2 agents
- 3 actions
- 7 state variables
- Fully Observable
- Discrete State Space (2-table with ~8,100 s,a pairs)
- Stochastic Actions

Source Task: Knight’s Joust

- 3 vs. 2 Keepaway
  - Goal: Maintain possession of ball
  - 2 agents
  - 3 actions
  - 13 state variables
  - Partially Observable
  - Continuous State Space
  - Stochastic Actions

- The idea: learn and transfer learning-independent rules
- Task difference: state features and actions
- Transferred knowledge: rules
- Learning algorithm: any / sarsa
- Metric: learning speed
Rule Transfer Overview

1. Learn a policy ($\pi : S \rightarrow A$) in the source task
   - TD, Policy Search, Model-Based, etc.
2. Learn a decision list, $D_{source}$, summarizing $\pi$
3. Translate ($D_{source} \rightarrow D_{target}$) (applies to target task)
   - State variables and actions can differ in two tasks
4. Use $D_{target}$ to learn a policy in target task

Allows for different learning methods and function approximators in source and target tasks

Section Outline

- Introduction to RL
  - The dimensions of transfer
    - task relatedness
    - transferred knowledge
    - learning algorithms
  - Transfer between tasks with same state-action variables
    - From one source task to one target task
    - From many source tasks to one target task
    - Multitask learning: Learning a distribution of tasks
  - Transfer between tasks with different state-action variables
    - No explicit mapping
    - Mapping state variables and actions between tasks
    - Learning the inter-task mapping

“Cross-domain Transfer for Reinforcement Learning,” Taylor and Stone 2007

<table>
<thead>
<tr>
<th>Training Time (simulator hours)</th>
<th>Episode Duration (simulator seconds)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>10</td>
<td>10</td>
</tr>
<tr>
<td>15</td>
<td>15</td>
</tr>
<tr>
<td>20</td>
<td>20</td>
</tr>
<tr>
<td>25</td>
<td>25</td>
</tr>
<tr>
<td>30</td>
<td>30</td>
</tr>
<tr>
<td>35</td>
<td>35</td>
</tr>
<tr>
<td>40</td>
<td>40</td>
</tr>
</tbody>
</table>

“Keepaway Transfer Learning Results”

<table>
<thead>
<tr>
<th>No Transfer</th>
<th>Ringworld</th>
<th>Knight’s Joust</th>
</tr>
</thead>
<tbody>
<tr>
<td>Initial</td>
<td>7.3</td>
<td>11.0</td>
</tr>
<tr>
<td>Asymptotic</td>
<td>21.6</td>
<td>24.0</td>
</tr>
<tr>
<td>Accumulated</td>
<td>756.7</td>
<td>842.0</td>
</tr>
</tbody>
</table>

“Keepaway Transfer Learning Results”

“Knight’s Joust: 50,000 episodes”

“Ringworld: 20,000 episodes”

Learning Task Relationships

- Sometimes task relationships are unknown
- Necessary for Autonomous Transfer
- But finding similarities (analogies) can be very hard!
- Key ideas:
  1. Data Driven
     - Agents may generate data (experience) in both tasks
     - Leverage existing machine learning techniques
  2. Domain Analysis
     - Find similarities in problem structure
**MASTER Overview**

"Transferring instances for Model-Based Reinforcement Learning", Taylor et al., 2008

**Modeling Approximate State Transitions by Exploiting Regression**

- **Goals:**
  - Learn inter-task mapping between tasks
  - Minimize data complexity
  - No background knowledge needed

- **Overview:**
  1. Record data in source task
  2. Record small amount of data in target task
  3. Analyze data off-line to determine best mapping
  4. Use mapping in target task

### Observations

- **Pros:**
  - Very little target task data needed (sample complexity)
  - Analysis for discovering mappings is off-line

- **Cons:**
  - Exponential in # of state variables and actions

---

**MASTER Algorithm**

Record observed \((s_{\text{source}}, a_{\text{source}}, s'_{\text{source}})\) tuples in source task

Record small number of \((s_{\text{target}}, a_{\text{target}}, s'_{\text{target}})\) tuples in target task

Learn one-step transition model, \(T(S_T, A_T)\), for the target task:

\[
M(s_{\text{target}}, a_{\text{target}}) \rightarrow s'_{\text{target}}
\]

for every possible action mapping \(\chi_A\)

for every possible state variable mapping \(\chi_X\)

Transform recorded source task tuples

Calculate the error of the transformed source task tuples on the target task model:

\[
\sum (M(s_{\text{transformed}}, a_{\text{transformed}}) - s'_{\text{transformed}})^2
\]

**Transfer in 3D Mountain Car**

Hand-Coded

3D Mountain Car

Average Both
Other Data-Driven Methods

- **Classification**
  - Taylor et al., 2007

- **Implicit search via multiple experts**
  - Talvitie and Singh, 2007

- **Homomorphisms**
  - Soni and Singh, 2006
  - Sorg and Singh, 2009 at AAMAS-09

Domain Analysis Approach 1

- In **GGP**, complete model is given
- “Graph-based domain mapping for transfer learning in general games,” Kuhlmann and Stone, 2007
  - Produces **rule-graph** for source task(s)
  - In target task, find **isomorphic** source state

Domain Analysis Approach 2

- Qualitative Dynamic Bayes Networks (QDBN)
- “Value-function-based transfer for reinforcement learning using structure mapping,” Liu and Stone, 2006
- Use variant of the **Structure Mapping Engine**

Mapping Learning: Conclusions

- Method depends on **knowledge** available
- Never **guaranteed** to work
- Useful in the general case?