Incomparable rewards

Georgia Tech Research Institute

Christopher Simpkins chris.simpkins@gatech.edu
Advisor: Charles Isbell isbell@cc.gatech.edu
AAAI-DC Mentor: Sven Koenig, USC

My Thesis

Endowing an arbitrator with its own learner and global knowledge/reward signal can overcome critical problems in multiple-goal reinforcement learning to enable truly modular agent software engineering.

Modularity in Reinforcement Learning

Modularity is essential to real-world software engineering.

- Decomposition: break big problem into smaller problems.
- Reuse: apply solutions to smaller problems in new contexts.
- Existing approaches address decomposition.
- Hierarchical Reinforcement Learning (Temporal Decomposition)
  - Options (Sutton et al. 1999)
  - (Probabilistic) Hierarchical Abstract Machines (ALisp) (Parr & Russell 1998)
  - MAXQ (Dietterich 1998)
- Multiple-goal reinforcement learning (Concurrent Decomposition)
  - Q-Decomposition (Russell & Zimdars 2003)
  - GM-Sarsa (Sprague & Ballard 2003)

But reuse is a challenge ...

Our Focus: Multiple Goal Reinforcement Learning

A Learning agent, M, decomposed into n subagents, M = \{M\}\n
(Bhat et al. 2006), (Sprague & Ballard 2003), (Russell & Zimdars 2003). Each subagent shares an action set but has distinct state spaces and rewards. Q(s, a) is the aggregate Q-value for the overall agent. Arbitrator’s job is to find Q, and use it to select actions.

- Greatest Mass Q-Learning (GM-Q): Q = \sum_{s} Q(s, a).
- Top Q-Learning (Top-Q): Q = max, Q(s, a).
- Negotiated W-Learning (Neg-W): subagent with the most lose selects action.

Problem: R(s, a) = \sum_{s',N} R(s, a) must hold, i.e., rewards must be comparable.

A Simple Example: Furry Creature

FindFood Rewards

FindFood Values and Resulting Policy

Example world adapted from (Russell & Norvig 2003).

No action selection logic. What separated from how (Simpkins et al. 2008).

The Difficulty of Multi-Goal RL (Bhat et al. 2006)

Command arbitration is Arrow’s Paradox (Arrow 1963). An agent is a “society” of subagents, and command arbitration is social choice.

We want arbitrator to have following properties:

- Universality: the ability to handle any possible set of subagents.
- Unanimity: guarantee that if every subagent prefers action A, so will the arbitrator.
- Independence of Irrelevant Alternatives: each subagent’s preference for actions A and B are independent of the availability of any other action C. Prevents any particular subagent from affecting the global action choice by dishonestly reporting its own preference ordering.
- Scale Invariance: ability to scale any subagent’s Q-values without affecting the arbitrator’s choice. This is the crucial property that allows separately authored subagents with incomparable reward signals.
- Non-Dictatorship: no subagent gets its way all the time.

Problem: If |A| > 3, then there does not exist an arbitration function that satisfies each of the properties listed above. So even our simple furry creature is too complex for ideal arbitration.

The Solution: Intelligent Arbitration

Arbitrator performs command arbitration (Brooks 1986). Arbitrator is a special subagent, A0, with composite state space S0 = S1 × S2 × ... × Sn, an action set that represents choosing a subagent, A0 = 1 ... n, and a reward signal that represents the “greater good.”

The agent’s policy is defined indirectly by the arbitrator’s policy, Q0(s, a), which assigns probabilities to selection of each subagent’s preferred action for each state. We have relaxed the non-dictatorship requirement of ideal arbitration. By Arrow’s theorem, other properties will still hold.

In practical terms, means adding an additional subagent to Furry Creature, e.g., LiveLongProsper.

- “Greater good” - why avoid predator, why eat?
  - Encodes the tradeoffs between subagents.
  - Could be hand-authored, or could be another RL agent.
  - For the small cost of authoring a reward signal that represents the “greater good” you get true modularity, that is, the ability to combine separately authored subagents with incomparable rewards. Problem. Solved.

Validation

GM-Sarsa:

Comparables

Incompreciable

Our arbitrator:

Comparables

Incompreciable

Contributions and Issues

- An arbitration algorithm for modular reinforcement learning.
- A language specification and implementation that integrates modular reinforcement learning. (First a Scala lib, then DSL.)
- Code reuse versus policy reuse (Under what conditions can a policy be reused in another context?).
- Language/software engineering issues: debugging, inheritance (e.g., what does it mean to be a subclass of another agent?).
- Reward authoring.
Integrating Reinforcement Learning into a Programming Language

Christopher Simpkins  chris.simpkins@gatech.edu
Advisor: Charles Isbell  isbell@cc.gatech.edu
AAAI-DC Mentor: Sven Koenig, USC