Actor-Critic Policy Learning in Cooperative Planning

Josh Redding, Alborz Geramifard
Han-Lim Choi and Jonathan P. How

Aerospace Controls Lab, MIT

August 22, 2011
Motivating Example

- Image
- Track Unknown Vehicle
- Escort
- Rover Recovery
- Rover Deployment
- Cooperative Search

A. Whitten, 2010
Challenges of Cooperative Planning

1. Cooperative planning uses models
   - E.g. vehicle dynamics, fuel use, rules of engagement, embedded strategies, desired behaviors, etc...
   - Models enable anticipation of likely events & prediction of resulting behavior

2. Models are approximated
   - Planning with stochastic models is time consuming → Model simplification
   - Un-modeled uncertainties, parameter uncertainties

3. Result is sub-optimal planner output
   - Sub-optimalities range from $\epsilon$ to catastrophic
   - Mismatch between actual and expected performance
Open Questions

1. How can current multi-agent planners balance between robustness and performance better?

2. How should the learning algorithms be formulated to best address the errors and uncertainties present in the multi-agent planning problem?

3. How can a learning algorithm be formulated to enable a more intelligent planner response, given stochastic models?
Focus

▶ How can a learning algorithm be formulated to enable a more intelligent planner response, given stochastic models?

Objectives

▶ Increase model fidelity to narrow the gap between expected and actual performance

▶ Increase cooperative planner performance over time
Two Worlds

- Cooperative Control
  - Provides *fast* solutions
  - *Sub-optimal*

- Online Learning Techniques
  - Handles *stochastic* system and unknown models
  - *High* sample complexity
  - Might *crash* the plane to learn!

- Can we take the best of the both worlds?
Best of the Both Worlds

- Cooperative control scheme that learns over time
  - Learning $\rightarrow$ Improve Sub-optimal Solutions
  - Fast Planning $\rightarrow$ Reduce Sample Complexity
  - Fast Planning $\rightarrow$ Avoid Catastrophic plans
A Framework for Planning + Learning

- Template architecture for multi-agent planning and learning
- A cooperative planner coupled with learning and analysis algorithms to improve future plans
  - Distinct elements cut combinatorial complexity of full integration and enable decentralized planning and learning
- Intelligent cooperative control architecture (iCCA)
Merging Point

- Deterministic → Stochastic
  - Plan (Trajectory) → Policy (Behavior)

- Import a plan into a policy
  - **Bias** the policy for those states on the planned trajectory
  - Need a method to **explicitly** represent the policy

- Avoid taking actions with unsustainable outcome
  - **Override** with the safe (planned) action
  - Provide a **virtual** negative feedback
**Scenario:** A small team of fuel-limited UAVs (triangles) in a simple, uncertain world cooperate to visit a set of targets (circles) with stochastic rewards

**Objective:** Maximize collective reward

**Key features:**
- Stochastic target rewards (probability shown in nearest cloud)
- Specific windows for target visit-times
Stochastic WTA Formulation under iCCA

- Apply iCCA template [Redding et al, 2010]
- **Cooperative Planner** ← Consensus-Based Bundle Algorithm (CBBA)
- **Learning Algorithm** ← Actor-Critic Reinforcement Learning
- **Performance Analysis** ← Risk Assessment
Problem Description

Scenario

Stochastic WTA Formulation under iCCA

- Apply iCCA template [Redding et al, 2010]
- **Cooperative Planner** ← Consensus-Based Bundle Algorithm (CBBA)
- **Learning Algorithm** ← Actor-Critic Reinforcement Learning
- **Performance Analysis** ← Risk Assessment
Problem Description

Cooperative Planner

Stochastic WTA Formulation under iCCA

Consensus-Based Bundle Algorithm (CBBA)

- CBBA is a deterministic planner
- Applying CBBA to a stochastic problem introduces sub-optimalities
- CBBA provides a “plan”, which seeds an initial policy $\pi_0$
- $\pi_0$ provides contingency actions
Current approach is inspired by the Consensus-Based Bundle Algorithm (CBBA) [Choi, Brunet, How TRO 2009]

- Key new idea: Focus on agreement of plans. Combines auction mechanism for decentralized task selection and consensus protocol for resolving conflicted selections.
- Note: auction without auctioneer.

Consensus on information & winning bids, winning agents:

- Situational awareness used to improve score estimates.
- Best bid for each task used to allocate tasks w/o conflicts.

\[
y_i(j) = \text{what agent } i \text{ thinks is the maximum bid on task } j
\]
\[
z_i(j) = \text{who agent } i \text{ thinks bid max value on task } j
\]

Distributed algorithm, but also provides a fast central solution.
Consensus Based Bundle Algorithm

- Distributed multi-task assignment algorithm: CBBA
  - Each agent carries a single bundle of tasks that is populated by greedy task selection process
  - Consensus on marginal score of each task not overall bundle score ⇒ suboptimal, but avoids bundle enumeration

- Phase 1: Bundle construction
  - Add task that gives largest marginal score improvement
  - Populate bundle to its full length $L_t$ (or feasibility)

- Phase 2: Conflict resolution – locally exchange $y, z, t_i$
  - Sophisticated decision map needed to account for marginal score dependency on previous selections
  - If an agent is outbid for a task in its bundle, it releases all tasks in bundle following that task
Reinforcement Learning

- **Value Function:**
  \[ Q^\pi(s, a) = E_\pi \left[ \sum_{t=0}^{\infty} \gamma^{t-1} r_t \middle| s_0 = s, a_0 = a, \right] \]

- **Temporal Difference TD Learning**
  \[ Q^\pi(s_t, a_t) = Q^\pi(s_t, a_t) + \alpha \delta_t(Q^\pi) \]
  \[ \delta_t(Q^\pi) = r_t + \gamma Q^\pi(s_{t+1}, a_{t+1}) - Q^\pi(s_t, a_t) \]
Stochastic WTA Formulation under iCCA

_actor-critic reinforcement learning_

- Combination of two popular RL thrusts
  - Policy search methods (Actor)
  - Value based techniques (Critic)
- Reduced variance of the policy gradient estimate
- Natural Actor Critic [Bhatnagar et al. 2007] - more reduced variance
- Convergence Guarantees
Problem Description

Learning Algorithm

Actor-Critic Reinforcement Learning

- Explore parts of world likely to lead to better system performance
- Actor-critic learning: $\pi(s)$ (actor) and $Q(s, a)$ (critic)

Actor handles the policy

- $\pi(s) = \frac{e^{P(s, a)/\tau}}{\sum_b e^{P(s, b)/\tau}}$
- $P(s, a)$: Preference of taking action $a$ from state $s$
- $\tau \in [0, \infty)$ acts as temperature (greedy $\rightarrow$ random action selection)
- $P(s, a) \leftarrow P(s, a) + \alpha Q(s, a)$
Actor-Critic Reinforcement Learning

- Explore parts of world likely to lead to better system performance
- Actor-critic learning: $\pi(s)$ (actor) and $Q(s, a)$ (critic)

**Critic handles the value function**

- Associates reward received with recent state/action pair
- Updates $Q(s, a)$ via Temporal-Difference (TD) algorithm
**Risk Analysis**

- Heuristic check of the candidate action $\pi(x)^a$, suggested by learner
- Rejects $\pi(x)^a$ if too “risky”, $\pi(x) \leftarrow \pi(x)^b$
- Reward $r(x)$ is virtual if $\pi(x)^a$ is too “risky”
Objective: Ensure the agent remains safely within its operational envelope and away from undesirable or catastrophic states.

Exploration can tend toward dangerous states as all information is valuable to learning algorithms - even negative information.

A virtual reward is introduced:
- Large negative value given to the learner for actions deemed too risky, where “risk” is defined according to domain-dependent rules.
- Learner is dissuaded from suggesting that action again due to its large negative value.
Simulation Setup

- Mixed Matlab C/C++ implementation

- Two stochastic WTA scenarios:
  1. 2 UAVs, 7 Targets
  2. 2 UAVs, 10 Targets

- Four test cases per scenario:
  1. Optimal: Dynamic programming
  2. CBBA only: No learning to augment the baseline plan
  3. Actor-Critic only: Learning not seeded with baseline plan.
  4. Actor-Critic + CBBA: Instance of iCCA framework
Simulation Setup II

- Parameter Initialization
  - \( P(s, a) = \begin{cases} 
  100 & \text{if } (s, a) \text{ is on the CBBA planned trajectory} \\
  0 & \text{otherwise} 
\end{cases} \)
  - \( Q(s, a) = 0, \tau \leftarrow 1 \)

- Risk Analyzer
  - Given \((s, a)\), calculate the shortest path from the successive state to the base.
  - If remaining fuel is not sufficient
    - Action \( a \) is replaced with CBBA solution ran from state \( s \).
    - Set virtual reward so that \( P(s, a) = -100 \).
Numerical Results

Scenario 1

2 UAVs, 7 Targets

- UAVs (triangles) and Targets (circles)
- Acceptable windows for target visit times in brackets, e.g. [2,3]
- Target visit rewards
- Probability of receiving reward shown in cloud
- ≈ 100 million state-action pairs

- iCCA and Actor-Critic test cases were run for 60 episodes
- CBBA was run on the deterministic version of the stochastic problem for 10,000 episodes
Numerical Results

Scenario 1

2 UAVs, 7 Targets: Simulation Results

Comparison of Collective Rewards

- (Black) Optimal as calculated via dynamic programming
- (Red) CBBA only
- (Blue) Actor-critic only
- (Green) Coupled CBBA + actor-critic via iCCA
2 UAVs, 10 Targets

- UAVs (triangles) and Targets (circles)
- Acceptable windows for target visit times in brackets, e.g. [2,3]
- Target visit rewards
- Probability of receiving reward shown in cloud
- ≈ 9 billion state-action pairs

- iCCA and Actor-Critic test cases were run for 30 episodes
- CBBA was run on the deterministic version of the stochastic problem for 10,000 episodes
2 UAVs, 10 Targets: Simulation Results

Comparison of Collective Rewards

- Optimal solution intractable
- (Red) CBBA only
- (Blue) Actor-critic only
- (Green) Coupled CBBA + actor-critic via iCCA
Conclusions

A reinforcement learning algorithm was implemented under iCCA to **improve planner response** under stochastic models.

A safe initial policy was incrementally adapted by a natural actor-critic learning algorithm to increase planner performance over time.

Approach successfully demonstrated in simulation with limited-fuel UAVs visiting stochastic targets.

**Current Work:**
- Extend to other *forms* of cooperative planners.
- Extend tabular representation to function approximation to improve scalability of problem formulation.
- Formally define the notion of “risk”.
- Implement virtual forward search for suggested actions.