Complex Backup Strategies in Monte Carlo Tree Search

Piyush Khandelwal, Elad Liebman, Scott Niekum, and Peter Stone

University of Texas at Austin

ICML 2016
Monte Carlo Tree Search

**MDP**

- **Agent**
- **Environment**
- **Reward** $r_t$
- **Next State** $s_{t+1}$
- **Action** $a_t$

**Planning Start State**

- **MCTS**
- **Actions**
- **Next State** $s_{t+1}$
- **Action** $a_{t+1}$, $r_{t+1}$

**Backup Strategies in MCTS**
Monte Carlo Tree Search

4 stages in MCTS:
- Selection
- Expansion
- Simulation
- Backpropagation
Monte Carlo backup for single trajectory:

\[ R = \sum_{i=0}^{L-1} \gamma^i r_{t+i} \]

Across all trajectories:

\[ Q(s_t, a_t) = \mathbb{E} \left[ \sum_{i=0}^{L-1} \gamma^i r_{t+i} \right] \]

Can we do better?
Contribution:
➢ Formalize and analyze different on-policy/off-policy complex backup approaches from RL literature for MCTS planning.

Talk outline:
➢ Review complex backup strategies from RL in MCTS context.
➢ Empirical evaluation using IPC benchmarks.
➢ Explore relationship between domain structure and backup strategy performance.
n-step return (bias-variance tradeoff)

We can compute the return sample in many different ways!

1-step:

\[ R^{(1)} = r_t + \gamma Q(s_{t+1}, a_{t+1}), \]

n-step:

\[ R^{(n)} = \left[ \sum_{i=0}^{n-1} \gamma^i r_{t+i} \right] + \gamma^n Q(s_{t+n}, a_{t+n}) \]

Monte Carlo:

\[ R = \sum_{i=0}^{L-1} \gamma^i r_{t+i} \]

We have estimates for all Q values while performing backpropagation.
MCTS - Complex return

Complex return: \( R^C = \sum_{i=1}^{L} [w_{n,L} \cdot R^{(n)}] \)

\( \lambda \)-return/eligibility [Rummery 1995]:

\[ w_{n,L}^\lambda = \begin{cases} (1 - \lambda)\lambda^{n-1} & 1 \leq n < L \\ \lambda^L & n = L \end{cases} \]

⇒ MCTS(\( \lambda \))

\( \gamma \)-return weights [Konidaris et al. 2011]:

\[ w_{n,L}^\gamma = \frac{\left(\sum_{i=1}^{n} \gamma^{2(i-1)}\right)^{-1}}{\sum_{n=1}^{L} \left(\sum_{i=1}^{n} \gamma^{2(i-1)}\right)^{-1}} \]

⇒ MCTS\( \gamma \)
Complex return:

\[ R^C = \sum_{i=1}^{L} [w_{n,L} \cdot R^{(n)}] \]

\( \lambda \)-return/eligibility [Rummery 1995]:

\[ w_{n,L}^\lambda = \begin{cases} (1 - \lambda)\lambda^{n-1} & 1 \leq n < L \\ \lambda^n & n = L \end{cases} \]

\( \Rightarrow \) MCTS(\( \lambda \))

- Easier to implement.
- Assumes n-step return variances increase @ \( \lambda^{-1} \).

\( \gamma \)-return weights [Konidaris et al. 2011]:

\[ w_{n,L}^\gamma = \frac{\sum_{i=1}^{n} \gamma^{2(i-1)}}{\sum_{n=1}^{L} (\sum_{i=1}^{n} \gamma^{2(i-1)})^{-1}} \]

\( \Rightarrow \) MCTS\( \gamma \)

- Parameter free.
- Assumes n-step return variances are highly correlated.
MaxMCTS - Off-policy style returns

Backup using best known action:

\[ R^{(1)} = r_t + \gamma \max_a Q(s_{t+1}, a) \]
\[ R^{(n)} = \sum_{i=0}^{n-1} \gamma^i r_{t+i} + \gamma^n \max_a Q(s_{t+n}, a) \]

Intuition:

➢ Don’t penalize exploratory actions.
➢ Reinforce previously seen better trajectories instead.

Equivalent to Peng’s Q(\(\lambda\)) style updates.

MaxMCTS(\(\lambda\)) and MaxMCTS\(\gamma\)
Experiments

- 4 variants:
  - On-policy: MCTS(\(\lambda\)) and MCTS\(\_\gamma\)
  - Off-policy: MaxMCTS(\(\lambda\)) and MaxMCTS\(\_\gamma\)

- Test performance in IPC domains
  - Limited planning time (10,000 rollouts per step).

- Grid-world experiments to explore dependency between domain structure and backup strategy performance.
IPC - Random action selection

\[ \text{Monte Carlo} \]

Elevators

Reward

\[ \lambda \]

0 0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9 1

Recon

Skill Teaching

Reward
IPC - Random action selection

![Graph showing reward vs. λ for Elevators and Recon]

- Monte Carlo
- MaxMCTS_γ
- MaxMCTS(λ)

![Graph showing reward vs. λ for Skill Teaching]

- Recon
- Skill Teaching
IPC - UCB1 action selection

Elevators

Recon

Skill Teaching

Monte Carlo
MaxMCTS_\gamma
MaxMCTS(\lambda)
Monte Carlo + UCT
MaxMCTS_\gamma + UCT
MaxMCTS(\lambda) + UCT

Reward vs \lambda

Piyush Khandelwal (UT Austin)
Computational Time Comparison

![Chart showing computational time comparison for different domains and strategies]

- **MaxMCTS**
- **Monte Carlo**
- MaxMCTS(λ)

Domain names: AA, CT, Elev, GOL, Nav, Recon, ST, Sys, Tam, Tra, TT, WP
Grid World Domain

- 90% chance of moving in intended direction.
- 10% chance of moving to any neighbor randomly.
Grid World Domain

Start

Variable number of 0 Reward Terminal States

Goal +100

<table>
<thead>
<tr>
<th>#0-Term</th>
<th>0</th>
<th>3</th>
<th>6</th>
<th>15</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\lambda = 1$</td>
<td>90.4</td>
<td>11.3</td>
<td>0.9</td>
<td>-2.2</td>
</tr>
<tr>
<td>$\lambda = 0.8$</td>
<td>90.2</td>
<td>28.0</td>
<td>10.7</td>
<td>-1.4</td>
</tr>
<tr>
<td>$\lambda = 0.6$</td>
<td>89.5</td>
<td>62.8</td>
<td>45.3</td>
<td>8.5</td>
</tr>
<tr>
<td>$\lambda = 0.4$</td>
<td>88.7</td>
<td><strong>85.1</strong></td>
<td>77.6</td>
<td>24.1</td>
</tr>
<tr>
<td>$\lambda = 0.2$</td>
<td>87.7</td>
<td>82.6</td>
<td><strong>78.1</strong></td>
<td>28.4</td>
</tr>
<tr>
<td>$\lambda = 0$</td>
<td>84.5</td>
<td>79.8</td>
<td>74.1</td>
<td><strong>31.8</strong></td>
</tr>
</tbody>
</table>
Related Work

- $\lambda$-return has been applied previously for planning:
  - TEXPLORE used a slightly different version of MaxMCTS($\lambda$) [Hester 2012].
  - Dyna2 used eligibility traces [Silver et al. 2008].

- Other backpropagation strategies:
  - MaxMCTS($\lambda=0$) is equivalent to MaxUCT [Keller, Helmert 2012].
  - Coulom analyzed hand-designed backpropagation strategies in 9x9 Computer Go [Coulom 2007].

- Planning Horizon:
  - Dependence of planning horizon on performance [Jiang et al. 2015].
Conclusions

➢ In some domains, selecting the right complex backup strategy is important.

➢ $\text{MaxMCTS}_\gamma$ is a parameter-free approach that always performs better than/equivalent to Monte Carlo.

➢ $\text{MaxMCTS}(\lambda)$ performs best if $\lambda$ can be selected appropriately.

➢ Backup strategy performance related to number of trajectories with high rewards.
Multi-robot coordination

[Khandelwal et al. 2015]

- 84 discrete and continuous factors
- 100-500 actions per state (10-50 after heuristic reduction).