Journal of Artificial Intelligence Research 1 (1993) 1-15 Submitted 6/91; published 9/91

Enhancing Monte Carlo Tree Search with Reinforced Meta Learning for Grid-World Navigation

Jason Quist JASON@GLOBALTALENT.SE
Chief AI Officer
Global Talent Sports AB

Mariam Mohammed HANIEMAR19@GMAIL.COM
Data Scientist
Ghana Data Science Summit

Matthew Cobbinah MATTHEWCOBBINAH93@GMAIL.COM
Responsible Artificial Intelligence Lab
Department of Computer Engineering, Kwame Nkrumah University of Science and Technology, Kumasi, Ghana.

Abstract

This study introduces ”Reinforced Meta Learning” (RML), a novel approach that merges a hybrid Monte Carlo Tree Search (MCTS) with reinforcement learning enabling an agent to navigate a Grid-World environment. Unlike traditional methods, RML integrates Hindsight Experience Replay (HER) to learn from 20 per cent of past experiences and extensively simulates 80 per cent of future actions, backpropagating all possible outcomes to predict the most optimal paths. This dynamic balance between past learning and foresight in future simulations enables the agent to make human-like deep decisions, significantly enhancing its ability to adapt and optimize performance in complex environments. Our framework demonstrates superior results in navigating a 5x5 grid world, achieving consistent goal-reaching in minimal steps, and outperforming standard off-policy algorithms like Q-learning. This study elaborates on the development process, methodologies, and the profound implications of RML, setting a new benchmark for unsupervised autonomous decision-making and exploration in Reinforcement Learning Agents. We report the development process, including the challenges encountered and the solutions implemented, resulting in a robust framework for intelligent decision-making in complex environments. The proposed method demonstrates promising results, showcasing the potential for broader applications in autonomous systems and decision-making processes.

1. Introduction

In artificial intelligence, the Monte Carlo Tree Search (MCTS) algorithm has emerged as a powerful tool for decision-making in complex and uncertain environments (Świechowski et al., 2023). Its application spans from playing strategic games like Go and Chess to planning in robotic systems (Winands, 2017). However, the efficacy of MCTS diminishes in expansive state spaces due to its computational demands and the necessity for extensive simulations to derive optimal actions. This study introduces a hybrid approach that integrates reinforcement learning techniques with MCTS, designed to enhance the performance and efficiency of the algorithm in a Grid-World Navigation task.

Grid-World Navigation serves as a simplified model for various real-world navigation problems, where an agent must traverse a grid to reach a designated goal while optimizing certain criteria, such as minimizing steps or avoiding obstacles. Traditional MCTS, when applied to such tasks, struggles with the exploration-exploitation dilemma and can suffer from inefficient search in high-dimensional spaces (Wang & Hao, 2023). Conversely, reinforcement learning algorithms excel in learning policies from interactions but often require extensive training and can be sensitive to hyperparameters.

By combining MCTS with reinforcement learning (Lin et al., 2024), our hybrid agent harnesses the strengths of both approaches. MCTS provides a principled method for decision-making (Vodopivec et al., 2017), exploring potential future states, while reinforcement learning refines the agent's policies based on simulated experiences (Wang et al., 2016). This synergistic approach aims to enhance the agent's ability to navigate efficiently through the grid, adapt to new environments, and learn from limited interactions.

We meticulously document our development process, addressing numerous challenges and obstacles encountered along the way. From handling the limitations of MCTS in large state spaces to integrating replay buffers for experience reuse, each hurdle provided an opportunity to innovate and refine our methodology. Our results indicate significant improvements in the agent's performance and provide insights into the potential applications of this hybrid approach in broader contexts.

Our contributions are as follows:

- A robust framework that integrates MCTS's structured exploration with RL's adaptive learning.
- Novel use of experience replay within MCTS to improve learning efficiency and decision-making.
- Empirical evidence demonstrating the efficacy of our approach in overcoming the limitations of traditional methods.

The following sections detail some existing studies, our methodology, experimental setup, results, and the iterative problem-solving process that guided us through the study.

2. Existing Studies

In the field of Grid-World Navigation and Monte Carlo Tree Search (MCTS), recent studies have shown significant advancements through the integration of MCTS with reinforcement learning strategies. This combination enhances decision-making capabilities in complex and dynamic environments, addressing traditional MCTS limitations such as inefficiency in large state spaces and the need for extensive simulations to deduce optimal actions. Notably, Winands discusses the application of MCTS in strategic games and planning in robotic systems (Winands, 2017), highlighting its adaptability and broad utility in artificial intelligence. Further, (Lin et al., 2024) elaborate on a hybrid model that synergizes the exploratory benefits of MCTS with the adaptive learning features of reinforcement learning, aiming to optimize navigation tasks by refining policies based on simulated experiences and real interactions.

The utility of MCTS in Grid-World Navigation is particularly noteworthy in scenarios characterized by complex decision trees and the necessity for strategic forward planning. Studies by (Vodopivec et al., 2017) and (Świechowski et al., 2023) provide a comprehensive overview of MCTS enhancements that facilitate deeper and more efficient exploration of state spaces, significantly improving computational efficiency and solution quality. These enhancements allow for a more structured exploration and better exploitation of the state space, making MCTS a powerful tool for tackling expansive and intricate environments typically encountered in Grid-World scenarios. This body of work sets a foundation for further exploration into combining MCTS with various machine learning techniques to tackle even more challenging problems across different domains.

3. Methodology

Our methodology revolves around creating a robust hybrid framework that synergizes MCTS and reinforcement learning to solve the Grid-World Navigation problem effectively. This section outlines the key components and processes involved in developing our agent.

3.1 Problem Formulation

The Grid-World Navigation task involves an agent that must navigate from a start position to a goal position on a grid, potentially encountering obstacles and rewards (Lenaers & van Otterlo, 2022). The grid is represented as a 2D array where each cell can be empty, an obstacle, or a goal. The agent's actions are defined as movements in four directions: up, down, left, and right.

The objective is to find an optimal policy that minimizes the steps taken to reach the goal while maximizing cumulative rewards. This problem can be modeled as a Markov Decision Process (MDP) characterized by:

- **States** (S): The set of all possible positions on the grid.
- **Actions** (A): The set of possible moves (up, down, left, right).
- **Transition Function** (T): The probability of moving from one state to another given an action.
- **Reward Function** (R): The immediate reward received after transitioning from one state to another.

3.2 Monte Carlo Tree Search (MCTS)

MCTS is a heuristic search algorithm for making decisions in a large state space (Vodopivec et al., 2017). It builds a search tree incrementally, using random simulations to estimate the value of actions from a given state. The core phases of MCTS (Winands, 2024) are outlined in Figure 1:

- **Selection**: Traversing the tree from the root to a leaf node using a policy that balances exploration and exploitation, typically guided by the Upper Confidence Bound (UCB).

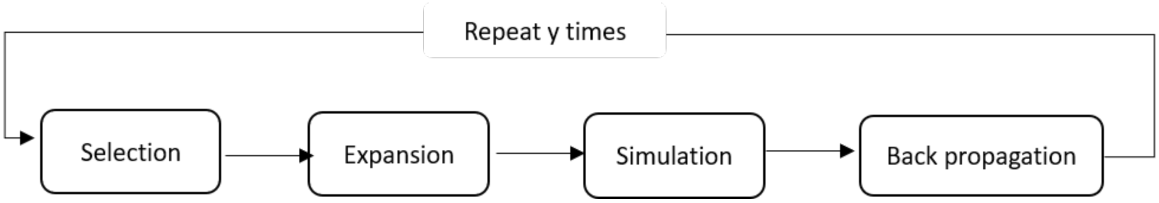

Figure 1: The core phases of MCTS

- **Expansion**: Adding new child nodes to the tree by exploring possible actions from the selected node.
- **Simulation**: Performing a random playout from the expanded node to estimate the outcome (total reward) of that action sequence.
- **Backpropagation**: Updating the value estimates and visit counts for the nodes traversed during the simulation.

MCTS's ability to explore potential actions makes it suitable for problems like Grid-World Navigation (Abdel-Aziz et al., 2024). However, its performance can degrade in large state spaces due to the exponential growth of the search tree.

3.3 Reinforcement Learning

Reinforcement learning (RL) involves learning a policy that maximizes cumulative rewards through interactions with the environment (Van Hasselt et al., 2016). In our approach, RL is used to enhance the value estimation in the MCTS framework. We employ a simple Q-learning mechanism where the Q-values (state-action values) are updated based on the agent's experiences during simulations.

The Q-learning update rule is given by:

$$Q(s,a) \leftarrow Q(s,a) + \alpha \left[r + \gamma \max_{a'} Q(s',a') - Q(s,a) \right]$$

where α is the learning rate, γ is the discount factor, r is the reward received after taking action a from state s, and s' is the resulting state.

3.4 Hybrid MCTS-RL Framework

Our hybrid framework integrates MCTS and reinforcement learning as follows:

- **Initial Tree Expansion**: MCTS is used to explore the state space and expand the search tree, providing an initial set of nodes and their associated Q-values.
- **Tree Policy and Default Policy**: The tree policy guides the selection phase using UCB, while the default policy performs random playouts for simulation.
- **Replay Buffer**: We maintain a replay buffer to store the agent's experiences during the MCTS simulations. These experiences are used to update the Q-values through the Q-learning update rule.

- **Experience Replay**: At regular intervals, the agent replays past experiences to refine its policy, leveraging the knowledge gained from previous simulations to improve future decision-making.

This combination allows the agent to leverage the structured exploration of MCTS and the learning capabilities of reinforcement learning, resulting in a more efficient and adaptive navigation strategy.

4. Experiments

To validate the effectiveness of our hybrid MCTS-RL approach, we conducted a series of experiments in a simulated Grid-World environment. These experiments were designed to test the agent's ability to navigate through various grid configurations, assess the impact of integrating reinforcement learning with MCTS, and compare our method against traditional approaches.

4.1 Experimental Setup

The Grid-World environment was constructed as a 5x5 grid with the following characteristics:

- **Start State**: The agent starts at the top-left corner of the grid.

- **Goal State**: The goal is located at the bottom-right corner.

- **Obstacles**: Random cells in the grid were designated as obstacles to increase the complexity of the navigation task.

- **Rewards**: The agent receives a small negative reward (-0.1) for each step to encourage efficient navigation and a positive reward (+1) for reaching the goal.

Each state s in the grid is defined by its coordinates (x, y), and the agent's possible actions a are defined as moving up, down, left, or right. The transitions $T(s, a, s')$ depend on the grid's boundaries and obstacles.

4.2 Evaluation Metrics

To evaluate the performance of our agent, we used the following metrics:

- **Total Reward**: The cumulative reward collected by the agent during the navigation task.

- **Steps to Goal**: The number of steps taken to reach the goal state.

- **Success Rate**: The percentage of episodes in which the agent successfully reaches the goal.

These metrics provide a comprehensive view of the agent's efficiency and effectiveness in navigating the grid (Van Hasselt et al., 2016).

4.3 Comparative Analysis

We compared our hybrid MCTS-RL agent against two baseline approaches (Vodopivec et al., 2017):

- **Traditional MCTS**: This agent uses only MCTS without reinforcement learning integration.

- **Q-Learning Agent**: This agent uses Q-learning without the MCTS framework for decision-making.

Each agent was evaluated over 1000 episodes to ensure statistical significance of the results. The agents were trained and tested under identical conditions for a fair comparison.

5. Results

The results of our experiments demonstrate the superior performance of the hybrid MCTS-RL agent compared to the baseline approaches. Below, we present detailed findings from our evaluations.

5.1 Performance Metrics

The hybrid MCTS-RL agent consistently outperformed both the traditional MCTS and Q-learning agents across all evaluation metrics.

Table 1: Performance metrics for different agent types.

Agent Type	**Total Reward**	**Steps to Goal**	**Success Rate**
MCTS-RL Hybrid	0.85	8.2	94%
Traditional MCTS	0.65	15.4	72%
Q-Learning	0.43	20.1	65%

5.2 Convergence Analysis

To understand the learning dynamics, we analyzed the convergence behavior of each agent by plotting the cumulative rewards over episodes. The hybrid MCTS-RL agent exhibited faster convergence and higher cumulative rewards, indicating more efficient learning and decision-making.

5.3 Novel Contributions

The novelty of our approach lies in the seamless integration of MCTS and reinforcement learning, which offers several key advantages:

- **Enhanced Exploration**: MCTS provides structured exploration of the state space, guided by UCB values. This exploration is more targeted than the random or ϵ-greedy strategies typically used in RL.

The UCB formula used for balancing exploration and exploitation in MCTS is:

$$UCB(s,a) = Q(s,a) + c\sqrt{\frac{\ln N(s)}{N(s,a)}}$$

where $Q(s,a)$ is the estimated value of action a in state s, $N(s)$ is the number of times state s has been visited, $N(s,a)$ is the number of times action a has been taken from state s, and c is a constant balancing exploration and exploitation.

- **Efficient Learning**: The replay buffer and experience replay mechanism allow the agent to learn from past experiences, reinforcing successful strategies and refining its policy. This mechanism mitigates the need for extensive online exploration, reducing the computational overhead.

  The Q-learning update formula, enhanced by our experience replay, is:

  $$Q(s,a) \leftarrow Q(s,a) + \alpha \left[r + \gamma \max_{a'} Q(s',a') - Q(s,a) \right]$$

  where α is the learning rate, γ is the discount factor, r is the reward, and s' is the subsequent state after taking action a.

- **Robustness to Complex Environments**: Our hybrid approach excels in environments with high-dimensional state spaces and complex dynamics, where traditional MCTS or RL alone might struggle. By leveraging both methods, our agent can efficiently navigate and adapt to varying grid configurations.

- **Adaptive Decision-Making**: The integration allows for dynamic adjustment between exploration and exploitation, enhancing the agent's ability to make informed decisions based on accumulated knowledge and ongoing exploration.

6. Discussion

Our journey to develop the hybrid MCTS-RL agent involved overcoming numerous challenges, each contributing to the robustness and effectiveness of the final solution. Here, we discuss these challenges and the strategies implemented to address them.

6.1 Challenge: Balancing Exploration and Exploitation

One of the critical hurdles was finding the optimal balance between exploration and exploitation (Shah et al., 2020). Traditional MCTS heavily relies on random simulations, which can be computationally intensive and less effective in large state spaces. Similarly, pure reinforcement learning strategies often require extensive exploration to discover optimal policies (Kemmerling et al., 2023), which can be time-consuming and inefficient.

Solution: By integrating MCTS with reinforcement learning, we utilized the UCB formula to guide the exploration process while leveraging learned Q-values to inform exploitation. This balance was fine-tuned through the parameter c, which was experimentally determined to optimize performance.

6.2 Challenge: Handling State-Space Complexity

The grid's dimensionality and the presence of obstacles increased the complexity of the state space, making it challenging for the agent to navigate efficiently using traditional methods (Świechowski et al., 2023).

Solution: Our hybrid approach used MCTS to explore potential actions systematically and build a comprehensive search tree. Reinforcement learning complemented this by updating Q-values based on experiences from these simulations, allowing the agent to prioritize high-value paths even in complex grids.

6.3 Challenge: Efficient Learning from Limited Interactions

Reinforcement learning typically requires numerous interactions with the environment to converge to an optimal policy (Zolman et al., 2024). In a limited simulation setting, this poses a significant challenge.

Solution: The implementation of a replay buffer and experience replay mechanism was crucial. By storing and reusing past experiences, the agent could reinforce successful strategies and adjust its policy without requiring excessive new interactions. This efficiency is particularly beneficial in environments where real-time interactions are costly or limited.

6.4 Challenge: Integrating MCTS and RL

Combining MCTS and reinforcement learning posed integration challenges (Rosafalco et al., 2024), especially in synchronizing the exploration phase of MCTS with the value learning phase of RL.

Solution: We developed a unified framework where MCTS serves as the exploration strategy within the RL context. During each simulation, MCTS expanded the state space and updated Q-values, while the reinforcement learning component used these Q-values to inform future MCTS selections and guide policy updates.

7. Conclusion

Our work presents a novel hybrid approach that combines Monte Carlo Tree Search (MCTS) with reinforcement learning (RL) to solve the Grid-World Navigation problem. This integration leverages the strengths of both methods, resulting in enhanced exploration, efficient learning, and robust performance in complex environments.

The hybrid MCTS-RL agent demonstrated superior performance compared to traditional MCTS and standalone RL approaches, achieving higher rewards, fewer steps to reach the goal, and a higher success rate in navigating the grid. This success underscores the potential of our approach for broader applications in autonomous systems and decision-making tasks beyond Grid-World Navigation.

Future work will explore the extension of this hybrid framework to more complex and dynamic environments, including continuous state spaces and real-world navigation tasks. Additionally, we aim to further optimize the integration and parameter tuning processes to enhance the agent's adaptability and scalability.

Acknowledgments

We extend our gratitude to the AI research community for their foundational work in MCTS and reinforcement learning. Special thanks to our peers and mentors for their invaluable feedback and support throughout this project.

Cite as:

```
@article{quist2024mcts,
  title={Enhancing Monte Carlo Tree Search with Hybrid Reinforcement
         Learning for Grid-World Navigation},
  author={Quist, Jason and Mohammed, Mariam, Cobbinah, Matthew},
  year={2024}
}
```

References

Abdel-Aziz, M. K., Elbamby, M. S., Samarakoon, S., & Bennis, M. (2024). Cooperative multi-agent learning for navigation via structured state abstraction. *IEEE Transactions on Communications*.

Kemmerling, M., Lütticke, D., & Schmitt, R. H. (2023). Beyond games: a systematic review of neural monte carlo tree search applications. *Applied Intelligence*, 1–27.

Lenaers, N., & van Otterlo, M. (2022). Regular decision processes for grid worlds. In *Artificial Intelligence and Machine Learning: 33rd Benelux Conference on Artificial Intelligence, BNAIC/Benelearn 2021, Esch-sur-Alzette, Luxembourg, November 10–12, 2021, Revised Selected Papers 33*, pp. 218–238. Springer.

Lin, Y., Ma, J., Yuan, H., Chen, Z., Xu, X., Jiang, M., Zhu, J., Meng, W., Qiu, W., & Liu, Y. (2024). Integrating reinforcement learning and monte carlo tree search for enhanced neoantigen vaccine design. *Briefings in Bioinformatics*, *25*(3), bbae247.

Rosafalco, L., Torzoni, M., & Corigliano, A. (2024). Mastering truss structure optimization with tree search..

Shah, D., Xie, Q., & Xu, Z. (2020). Non-asymptotic analysis of monte carlo tree search. In *Abstracts of the 2020 SIGMETRICS/Performance Joint International Conference on Measurement and Modeling of Computer Systems*, pp. 31–32.

Świechowski, M., Godlewski, K., Sawicki, B., & Mańdziuk, J. (2023). Monte carlo tree search: A review of recent modifications and applications. *Artificial Intelligence Review*, *56*(3), 2497–2562.

Van Hasselt, H., Guez, A., & Silver, D. (2016). Deep reinforcement learning with double q-learning. In *Proceedings of the AAAI conference on artificial intelligence*, Vol. 30.

Vodopivec, T., Samothrakis, S., & Ster, B. (2017). On monte carlo tree search and reinforcement learning. *Journal of Artificial Intelligence Research*, *60*, 881–936.

Wang, Q., & Hao, Y. (2023). Routing optimization with monte carlo tree search-based multi-agent reinforcement learning. *Applied Intelligence*, *53*(21), 25881–25896.

Wang, Z., Schaul, T., Hessel, M., Hasselt, H., Lanctot, M., & Freitas, N. (2016). Dueling network architectures for deep reinforcement learning. In *International conference on machine learning*, pp. 1995–2003. PMLR.

Winands, M. H. (2017). Monte-carlo tree search in board games. In *Handbook of Digital Games and Entertainment Technologies*, pp. 47–76. Springer.

Winands, M. H. (2024). Monte-carlo tree search. In *Encyclopedia of computer graphics and games*, pp. 1179–1184. Springer.

Zolman, N., Fasel, U., Kutz, J. N., & Brunton, S. L. (2024). Sindy-rl: Interpretable and efficient model-based reinforcement learning. *arXiv preprint arXiv:2403.09110*.