DOI 10.31891/2307-5732-2024-341-5-52
УДК 004.94

NOVIKOV ARTEM
Karazin Kharkiv National University
https://orcid.org/0009-0004-5914-7098
email: artem.slick@gmail.com
YANOVSKY VOLODYMYR
"Institute for Single Crystals" of National Academy of Sciences
https://orcid.org/0000-0003-0461-749X
e-mail: yanovsky@isc.kharkov.ua

EXPLORING THE LIMITS OF MCTS IN PAC-MAN: MAZE SIZE, SIMULATIONS, AND PERFORMANCE

This paper explores the performance and limitations of Monte Carlo Tree Search (MCTS) when applied to the Pac-Man game, with a particular focus on how maze complexity and the number of simulations affect the agent's decision-making process. The game serves as a dynamic environment where the MCTS agent must navigate mazes filled with rewards (capsules and food) and avoid adversarial agents (ghosts), creating a challenging testbed for decision-making algorithms.

The primary objective of this work is to assess the efficiency of MCTS across mazes of varying sizes and configurations, from small, optimized layouts to large, non-optimized ones. We aim to understand the trade-offs between computational resources (e.g., number of simulations) and the agent's overall performance, particularly in terms of score, win rate, and decision-making time. The experiments were conducted using different numbers of simulations per move, allowing the MCTS agent to build decision trees that guide its actions. This enabled us to observe how performance metrics evolve as the complexity of the environment increases.

The findings indicate that MCTS performs effectively in smaller, optimized mazes where paths are clearer, and the decision space is more manageable. However, as maze complexity grows—particularly in non-optimized environments filled with obstacles and unpredictable paths—the agent's performance deteriorates. A key insight is that increasing the number of simulations does improve decision quality to an extent, but only up to a point; beyond that, additional simulations incur a computational overhead that does not yield proportional gains in performance. Moreover, the experiments reveal that optimized maze designs allow the MCTS agent to make more informed and efficient decisions, while non-optimized mazes exacerbate the agent's struggle with unpredictability and more complex decision spaces. These findings underscore the limitations of MCTS in handling dynamic and irregular environments.

Key conclusions include the necessity of enhancing MCTS for complex scenarios by incorporating more advanced heuristic functions to evaluate game states more accurately, along with adaptive simulation strategies to better manage computational resources. Additionally, combining MCTS with reinforcement learning or neural networks could offer more robust solutions for tackling the growing complexity of both in-game and real-world environments. This research highlights several promising avenues for future exploration, particularly in applying MCTS to more intricate AI challenges such as autonomous navigation and real-time decision-making in robotics.

Keywords: mcts, pacman, maze complexity, decision-making, ai agents, modelling

НОВІКОВ АРТЕМ
Харківський національний університет ім. В.Н. Каразіна
ЯНОВСЬКИЙ ВОЛОДИМИР
«Інститут Монокристалів» Національної академії наук

ДОСЛІДЖЕННЯ МЕЖ MCTS У PAC-MAN: РОЗМІР ЛАБІРИНТУ, СИМУЛЯЦІЇ ТА ПРОДУКТИВНІСТЬ

Ця стаття досліджує продуктивність та обмеження алгоритму пошуку дерева Монте-Карло (MCTS) у грі Pac-Man, особливо звертаючи увагу на вплив складності лабіринту та кількості симуляцій на процес ухвалення рішень. Основною метою роботи є оцінка ефективності MCTS в умовах різних за розміром та конфігурацією лабіринтів, з аналізом компромісів між обчислювальною вартістю та результативністю агента. Дослідження показали, що MCTS добре працює в малих та оптимізованих лабіринтах, але його продуктивність значно знижується у великих і неструктурованих середовищах. Пропонується покращити алгоритм шляхом застосування адаптивних симуляцій та комбінування MCTS з іншими методами штучного інтелекту.

Ключові слова: мктс, пакман, складність лабіринту, прийняття рішень, ші агенти, моделювання

Problem Statement

In dynamic environments, decision-making agents must navigate complex scenarios with multiple adversaries, changing conditions, and limited information. The Pac-Man game, with its combination of adversarial ghosts, rewards, and penalties, serves as an ideal testbed for exploring the performance of decision-making algorithms. Monte Carlo Tree Search (MCTS), a widely used algorithm for making decisions under uncertainty, has shown promise in various AI applications but presents challenges in environments with increasing complexity and size.

One of the main challenges in deploying MCTS effectively is balancing exploration (searching for new strategies) and exploitation (using known successful strategies) while maintaining computational efficiency. Additionally, the performance of MCTS can degrade in larger, more complex environments due to the increasing number of potential states and actions.

The main goal of this paper is to evaluate the efficiency and performance of an MCTS agent in Pac-Man mazes of varying complexity and size, analyzing the impact of increasing the number of simulations and exploring how different maze configurations affect the agent's decision-making capabilities. Through these experiments, we aim to

identify scalability issues and propose potential improvements to enhance the MCTS algorithm's performance in complex environments.

Analysis of Recent Sources

In recent years, Monte Carlo Tree Search (MCTS) has gained significant traction in the field of artificial intelligence (AI), particularly for solving decision-making problems under uncertainty. Originally introduced in the domain of game playing, MCTS has been employed in a variety of complex environments where agents must explore vast decision spaces efficiently. One of its most notable applications has been in the game of Go, where MCTS, combined with deep neural networks, led to breakthroughs such as the development of AlphaGo. The strength of MCTS lies in its ability to balance exploration and exploitation by conducting numerous simulations to evaluate potential future states before making a decision.

In their research, Ou et al. conducted a comprehensive survey on the theories and applications of MCTS. They highlighted its growing success in decision-making scenarios like Go but also pointed out the ongoing challenges in real-time applications with imperfect information. Their work emphasizes the need for further development to extend MCTS to broader, more dynamic environments [1].

Similarly, in a study by Jiang, the success of MCTS in the game of Go is further exemplified through its integration with deep learning techniques in AlphaGo. Jiang's research demonstrates how MCTS, in combination with deep neural networks, significantly advanced the AI's ability to manage large decision spaces, surpassing traditional brute-force methods and even outperforming human players in one of the most complex board games in the world [2].

In another important study, Steinmetz and Gini explored the impact of parallelizing MCTS for more time-constrained scenarios. Their research focused on comparing various parallelization techniques, including root parallelization, where multiple independent trees are built simultaneously. Their findings showed that root parallelization provides better performance improvements under limited time constraints compared to other methods. This study offers valuable insights into optimizing MCTS for situations where time is a critical factor [3].

Overall, these studies illustrate the versatility and power of MCTS while also addressing its limitations, particularly in real-time and imperfect information scenarios. The continued research and optimization of MCTS, particularly through parallelization and integration with deep learning, hold promise for expanding its applicability beyond game environments and into more complex, real-world decision-making systems.

In the context of the Pac-Man game, MCTS offers an interesting opportunity to study decision-making under dynamic adversarial conditions. Pac-Man’s environment features elements such as moving adversaries (ghosts), static and dynamic rewards (food and capsules), and the challenge of navigating complex mazes. These characteristics make Pac-Man a perfect testbed for analyzing how decision-making algorithms, like MCTS, cope with uncertainty and complexity.

In their research, Pepels et al. explored the application of MCTS for controlling the Pac-Man character in the real-time game Ms. Pac-Man. They introduced several enhancements to adapt MCTS to this fast-paced environment, where decisions must be made within a strict 40 ms time limit. Their work highlighted key modifications, such as implementing a variable-depth tree, incorporating simulation strategies for both Pac-Man and the ghost teams, and reusing the search tree with a decay factor. These changes significantly improved the agent's ability to navigate the game by balancing survival and maximizing points, even when facing diverse ghost behaviors [4].

Further enhancing the capabilities of MCTS in the Pac-Man game, Nguyen and Thawonmas developed an MCTS-based strategy for controlling ghost teams. They combined rule-based control with MCTS to improve coordination among ghosts, enhancing their ability to predict Pac-Man’s movements. Their approach led to significant improvements in the ghosts' performance, as demonstrated by their success in winning competitions against other ghost teams [5].

Another study by Liu et al. demonstrated the effectiveness of using MCTS in combination with an artificial neural network (ANN) to control non-player characters (NPCs) in Pac-Man. The researchers showed how integrating ANN with MCTS reduced computational load, allowing for more efficient decision-making processes while maintaining competitive performance. This combination of MCTS and ANN proved successful in controlling both Pac-Man and the ghost teams, offering a robust approach to developing intelligent game opponents [6].

Beyond Pac-Man, the relevance of decision-making in dynamic environments extends to a range of real-world applications. In particular, 2D terrains in robotics and drone navigation often resemble the maze-like environments of Pac-Man. In these cases, agents (e.g., drones or robots) must traverse a complex space while avoiding obstacles (analogous to ghosts) and completing specific tasks (similar to eating capsules or food). The vulnerability of drones to certain threats—represented by the ghosts' scared state in the game—mirrors real-world scenarios where agents may be temporarily neutralized by external factors such as signal jamming or environmental conditions. For example, Bartolini et al. explored scenarios where drones inspect areas under uncertainty about the time and location of target events, building dynamic probabilistic maps for trajectory planning and collision avoidance [7]. This approach highlights the complexity and unpredictability drones face in real-world missions, much like Pac-Man navigating mazes under ghost threat.

Additional research by Pairet et al. addressed online motion planning under uncertainty, focusing on building an uncertainty-aware representation of the environment while ensuring safe navigation. Their method incrementally maps the surroundings and replans trajectories in real-time, ensuring drones adapt to unknown and changing environments [8]. Similarly, the work by Sandino et al. demonstrated how autonomous drones could efficiently plan their paths under detection uncertainty, a challenge that mirrors the unpredictability in Pac-Man's environment [9]. In

related research, Thatavarthy et al. tackled urban drone navigation, presenting a system that allows drones to navigate amongst high-rises using visual-inertial SLAM maps, further emphasizing the need for robust decision-making algorithms in uncertain environments [10]. These studies exemplify how AI models tested in games like Pac-Man can be extended to real-world drone navigation challenges, where uncertainty and real-time adaptation are crucial. In addition studies on drones and robots navigating uncertain terrains [11, 12] provide valuable insights into handling unpredictable environments.

The decision-making problem itself is a central theme in AI research. Numerous studies have focused on the difficulties agents face when making decisions in adversarial environments where other agents are actively working against them. Cheraghi et al., for instance, developed algorithms based on Depth First Search (DFS) and Breadth First Search (BFS) for autonomous robots to efficiently cover 2D terrains. Their study showed how robot swarms can self-organize and share data, much like how Pac-Man must navigate while avoiding ghosts that dynamically alter their strategies [13]. These challenges are particularly pronounced in the Pac-Man game, where ghosts actively seek out Pac-Man while the agent must simultaneously focus on maximizing its score. This balancing act between survival and optimization mirrors broader problems in AI, such as multi-agent systems where multiple actors with competing objectives interact in a shared environment. Similar adversarial dynamics have been explored in the field of multi-agent systems, such as in the work by Andreas et al., where adversary decision-making was modeled using Markov models to influence outcomes strategically [14].

Moreover, the dynamic nature of adversarial agents (ghosts) in Pac-Man is akin to real-world problems involving mobile threats. In applications like security patrolling or autonomous delivery, adversarial agents (e.g., intruders or obstacles) dynamically change positions, requiring decision-making algorithms to adapt in real-time. Similar to how ghosts adjust their behavior based on Pac-Man's location, adversaries in real-world scenarios often adapt their tactics in response to external cues. This dynamic decision-making process has been a focus of studies such as Deng and Jiang's work on modeling adversarial decision-making using fuzzy sets, game theory, and D number theory to handle conflict and uncertainty [15]. Likewise, Talebi et al. demonstrated how adversaries' decision-making can be influenced in complex dynamical systems using game theory and Q-learning techniques [16]. Such real-time adaptability and adversarial strategy adjustments are mirrored in Pac-Man's gameplay, where both Pac-Man and the ghosts must constantly reevaluate their tactics as the environment changes.

Furthermore, Yang and Parasuraman demonstrated how game theory could be used to model multi-agent decision-making in adversarial environments, allowing agents to cooperate or compete based on the situation [17]. This directly parallels Pac-Man's adversarial context, where both ghosts and Pac-Man must make strategic decisions based on the ever-changing state of the game. Real-world applications like autonomous drone navigation or urban planning benefit from similar strategies, where decision-making must balance competition and cooperation while adapting to dynamic environments.

In addition, research on decision-making in adversarial environments [18] shows how agents must adapt strategies based on opposing actions, much like Pac-Man adapting to ghost behaviors.

In conclusion, MCTS offers a versatile framework for solving decision-making problems not only in games like Pac-Man but also in more global real-world applications. Whether it is navigating a robot through a hazardous terrain or guiding a drone to avoid adversarial agents, the challenges faced by an MCTS agent in Pac-Man can be closely tied to these real-world scenarios. By understanding how MCTS interacts with complex environments, such as different maze configurations in Pac-Man, we gain valuable insights into the limitations and potential of AI in larger, more complex domains.

Purpose of the Article

The article aims to achieve the following key objectives to contribute to the understanding of MCTS's applicability in real-world problems involving decision-making under uncertainty:

- **Evaluate** the effectiveness of MCTS in different maze configurations, ranging from small to large and optimized vs. non-optimized layouts.
- **Analyze** the scalability of MCTS when increasing the number of simulations and the resulting trade-offs between computational cost and performance.
- **Identify** the limitations of MCTS in complex and adversarial environments, particularly in the context of dynamic agents like ghosts.
- **Propose** potential improvements for enhancing MCTS agent performance in larger, more complex scenarios.

Main Material

The experiments were conducted using a variety of Pac-Man maze configurations, which play a key role in determining the complexity and difficulty faced by the MCTS agent. The goal of the experiments remained consistent: Pac-Man must eat all the capsules while avoiding the ghosts. The scoring system and key game mechanics are as follows:

- **Time penalty:** Pac-Man loses 1 point per move, discouraging inefficient or overly defensive play.
- **Capsule reward:** Eating a capsule grants +500 points and temporarily puts ghosts into a vulnerable "scared" state.
- **Ghost reward:** While in the scared state, Pac-Man can eat ghosts for +200 points per ghost.
- **Food reward:** Each food dot eaten grants +10 points.
- **Scared timer:** After eating a capsule, ghosts remain vulnerable for 10 moves.

The "time" metric used in the results measures the average decision-making time (in seconds) for the MCTS agent, separate from the in-game move count.

A range of maze layouts was used to simulate different levels of complexity and challenge. The mazes were designed to test the MCTS agent's ability to navigate various spatial constraints and adversarial dynamics:

Small Mazes (7x20): These compact mazes feature narrow corridors and fewer paths, creating simpler decision spaces for the agent but less room for error. The small maze configurations are designed with only 2 ghosts—Blinky and Pinky—creating a focused environment where the agent has to balance immediate threat avoidance and capsule collection. Two types of small mazes were used (See Fig. 1):

1. **Optimized Small Maze:** Carefully structured with clear paths to the capsules and limited dead ends, allowing for more straightforward decision-making.
2. **Non-Optimized Small Maze:** Featuring more obstacles and dead ends, this version challenges the agent to find efficient routes while navigating complex layouts.

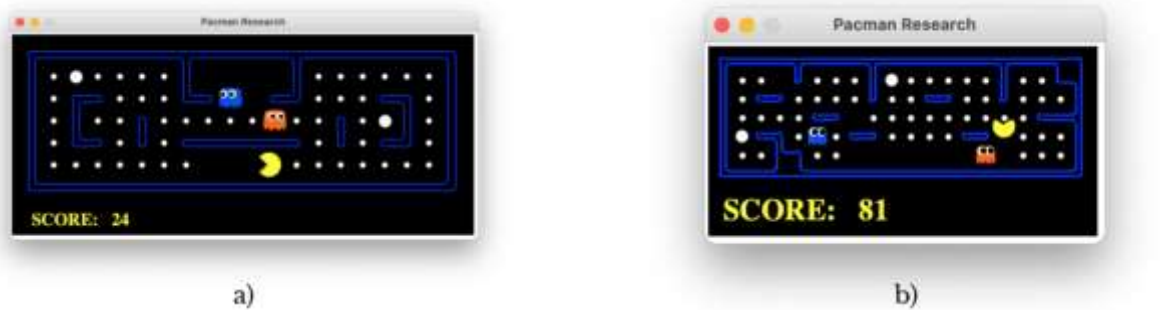

Fig.1 Small Mazes displaying in the game: a) - Optimized Small Maze; b) - Non-Optimized Small Maze

Medium Mazes (11x20): These medium-sized mazes introduce more corridors and junctions, offering multiple pathways to the capsules. The agent must consider more strategic routes while balancing ghost avoidance. Two ghost agents, Blinky and Pinky, are still used, but their behavior in a more open layout increases the decision space (See Fig. 2).

1. **Optimized Medium Maze:** Structured to offer multiple routes to each capsule, making it easier for Pac-Man to escape ghosts while pursuing capsules.
2. **Non-Optimized Medium Maze:** Featuring more dead ends and maze complexity, this layout forces the agent to make riskier decisions when navigating toward capsules or escaping ghosts.

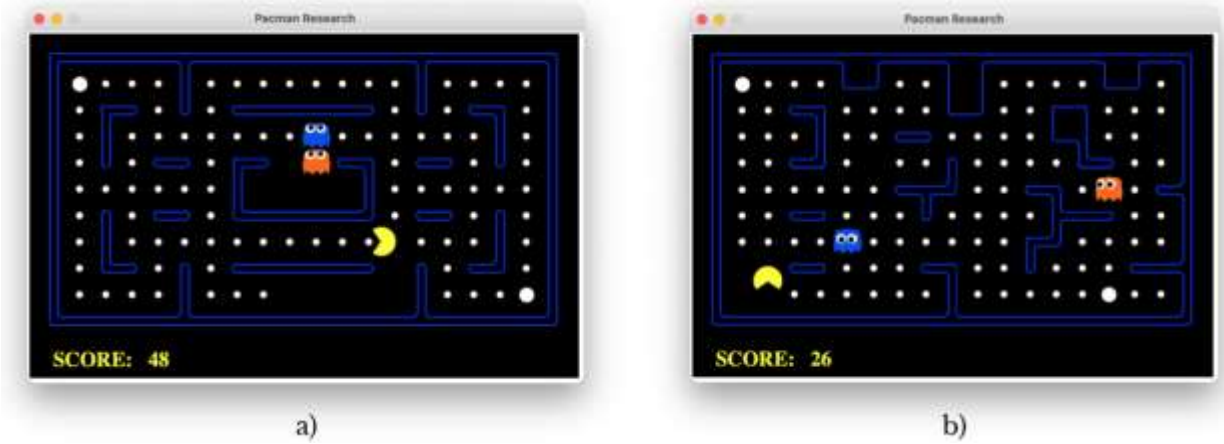

Fig.2 Medium Mazes displaying in the game: a) - Optimized Medium Maze; b) - Non-Optimized Medium Maze

Large Mazes (27x28): The large mazes represent the most complex environments, featuring multiple intersections, long corridors, and more opportunities for the ghosts to corner Pac-Man. These mazes introduce all four ghost agents—Blinky, Pinky, Inky, and Clyde—creating additional challenges. Larger mazes require the agent to navigate more complex paths while tracking the positions of all four ghosts (See Fig. 3).

1. **Optimized Large Maze:** The layout offers several escape routes and more strategic pathways to capsules, but the larger space gives the ghosts more room to surround Pac-Man.
2. **Non-Optimized Large Maze:** With irregular pathways and numerous obstacles, this maze significantly increases the difficulty, as the agent must navigate a far more complex environment where ghost behavior becomes harder to predict.

In the larger mazes, the diversity of ghost behaviors makes decision-making more complex. Blinky continues to chase Pac-Man directly, Pinky predicts future movements, Inky takes tactical positions based on Pac-Man's and Blinky's locations, and Clyde introduces an element of randomness, oscillating between chasing Pac-Man and moving unpredictably. These factors combine to create a multi-layered challenge, where the agent must make decisions under high uncertainty.

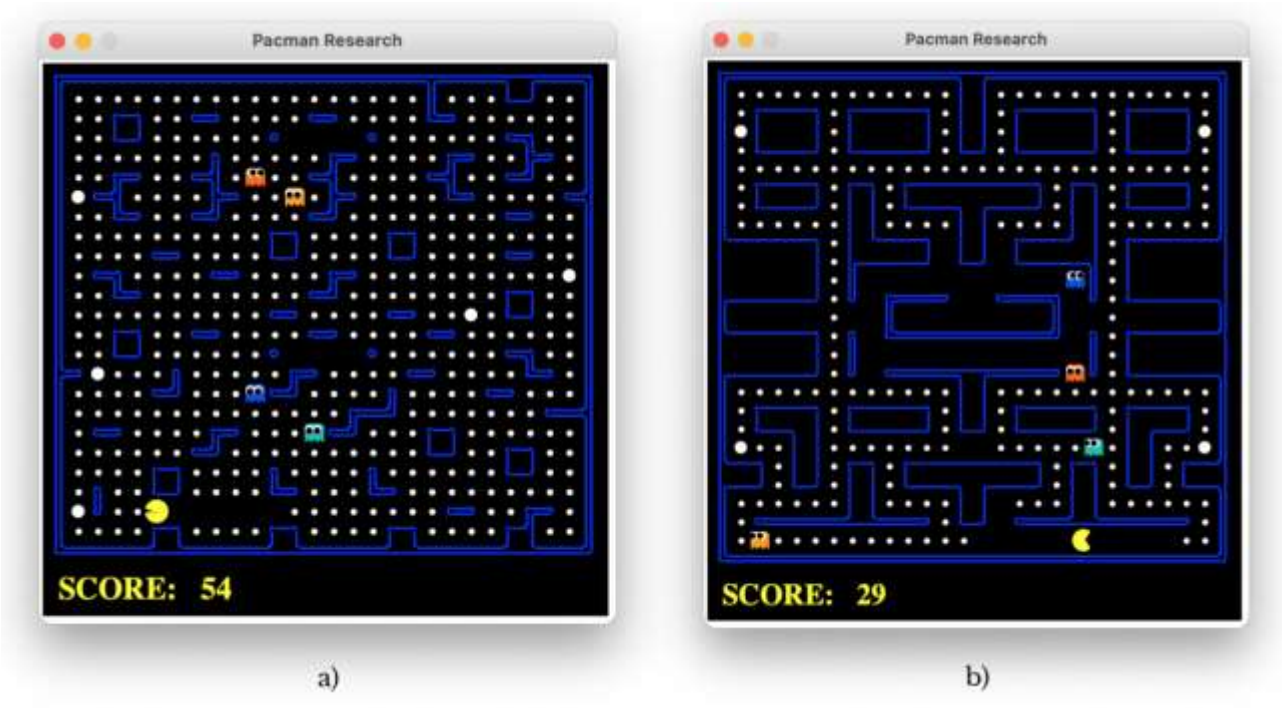

Fig.3 Large Mazes displaying in the game: a) - Optimized Large Maze; b) - Non-Optimized Large Maze

The four ghost agents each bring unique behavior to the game:

- **Blinky ("Chaser")**: Targets Pac-Man's current location. Blinky's straightforward strategy provides a consistent threat, making him the most immediate danger to Pac-Man at all times.
- **Pinky ("Ambusher")**: Anticipates Pac-Man's future position based on his movement direction. Pinky adds complexity by trying to cut off Pac-Man's escape routes rather than chasing him directly.
- **Inky ("Tactical")**: Inky's behavior is more complex, relying on both Pac-Man's position and Blinky's location to calculate its movement. Inky's unpredictability makes it difficult for Pac-Man to predict his future moves.
- **Clyde ("Scatter")**: Clyde behaves erratically, switching between chasing Pac-Man and wandering to random corners of the maze. His sporadic behavior adds an additional layer of uncertainty, forcing Pac-Man to react to both predictable and random threats.

The combination of these ghosts creates a constantly shifting set of threats, requiring the MCTS agent to balance immediate survival with long-term planning. In smaller mazes, the agent has fewer paths to escape, increasing the difficulty of managing Blinky's direct pursuit. In larger mazes, the unpredictability of Inky and Clyde requires the agent to adopt more adaptive strategies.

The experiments were conducted using the aforementioned mazes, with the following key parameters:

- **Maze complexity**: Small, medium, and large mazes were used to test how the MCTS agent handles environments of increasing complexity. The optimized and non-optimized variants allowed for testing the impact of layout design on agent performance.
- **Ghost count**: The number of ghosts was varied between 2 (Blinky and Pinky) in small and medium mazes, and 4 (Blinky, Pinky, Inky, and Clyde) in large mazes.
- **Simulations**: The MCTS agent was tested with varying numbers of simulations, ranging from 101 to 500 per decision. Each simulation run builds a tree of possible game states, with more simulations typically improving decision quality but increasing computation time.
- **Performance metrics**: The performance of the MCTS agent was evaluated based on:
    - **Average score**: Reflecting how effectively the agent collected food, capsules, and ghosts.
    - **Decision-making time**: Average computation time per move, indicating the trade-off between simulation count and agent speed.
    - **Win rate**: Percentage of games won out of 100 trials, used to measure the overall effectiveness of the agent under each condition.

Below, the results of the experiments are shown in individual tables for each maze configuration, detailing how the MCTS agent performed with different simulation counts.

The performance of the MCTS agent in this small, optimized maze shows incremental improvements with increasing simulations (see Table 1). However, the win rate fluctuates, indicating a limit to the effectiveness of additional simulations.

Table 1

Small Maze 1 (Optimized) results

Number of Simulations	Average Score	Average Time (sec)	Win Rate (%)
101	585.18	7.59	12
200	611.07	9.68	13
300	569.33	11.14	13
400	569.33	12.99	18
500	666.89	14.99	17

In the non-optimized small maze (see Table 2), the agent's average score improves with more simulations, but the win rate remains low, suggesting the increased complexity of the maze hinders the agent's ability to capitalize on more simulations.

Table 2

Small Maze 2 (Non-Optimized results

Number of Simulations	Average Score	Average Time (sec)	Win Rate (%)
101	784.41	9.28	9
200	717.09	10.81	10
300	732.33	12.98	9
400	797.49	16.16	7
500	900.73	21.22	12

In the optimized medium maze (see Table 3), the agent shows a steady improvement in both score and win rate, particularly with 500 simulations, where the win rate reaches 41%. The increased complexity of the medium maze provides the agent with more opportunities to optimize its path and collect rewards.

Table 3

Medium Maze 1 (Optimized) results

Number of Simulations	Average Score	Average Time (sec)	Win Rate (%)
101	1093.71	17.95	34
200	1103.57	23.71	33
300	1122.53	39.19	34
400	1126.94	49.39	36
500	1188.56	60.29	41

In the non-optimized medium maze (see Table 4), the results are more erratic, with both the score and win rate remaining low, even as the number of simulations increases. The complexity of the maze hinders the MCTS agent's ability to consistently perform well, indicating that additional simulations are less effective in highly irregular environments.

Table 4

Medium Maze 2 (Non-Optimized) results

Number of Simulations	Average Score	Average Time (sec)	Win Rate (%)
101	766.24	15.31	5
200	773.37	25.85	6
300	746.08	24.26	6
400	813.50	56.12	12
500	789.85	73.79	9

In the large optimized maze (see Table 5), the MCTS agent struggles with increased complexity, achieving only marginal improvements in score and win rate, even with more simulations. This highlights the diminishing returns in large mazes, where the complexity overwhelms the computational power available to the agent.

Table 5

Large Maze 1 (Optimized) results

Number of Simulations	Average Score	Average Time (sec)	Win Rate (%)
50	1255.41	36.23	1
101	1190.48	41.32	2
200	1369.72	60.83	2
300	1480.29	91.49	0

In the non-optimized large maze (see Table 6), the agent performs surprisingly well, achieving a high average score and win rate. However, starting at 300 simulations, the performance drops significantly, with computation time skyrocketing. This suggests that while more simulations initially benefit the agent, too many lead to a performance bottleneck, where computational costs outweigh the benefits of additional simulations.

Table 6

Large Maze 2 (Non-Optimized) results

Number of Simulations	Average Score	Average Time (sec)	Win Rate (%)
50	3109.16	37.12	29
101	3129.72	49.7	30
200	3138.23	85.54	32
300	2836.98	906.89	22

The results of the experiments reveal several key trends:

1. **Maze Size and Complexity Impact**: Larger and more complex mazes pose significant challenges to the MCTS agent. While the agent performs well in smaller, optimized mazes, it struggles in larger or non-optimized environments, where the decision space becomes exponentially more difficult to manage.
2. **Diminishing Returns of Simulations**: As the number of simulations increases, there is a point where additional simulations fail to improve the agent's performance. This is particularly evident in large mazes where, after 200–300 simulations, the agent's performance plateaus or even declines due to the high computational cost (see Table 5 and Table 6).
3. **Optimization vs. Non-Optimization**: The optimized mazes tend to provide the MCTS agent with clearer paths and fewer dead ends, improving both scores and win rates. In contrast, non-optimized mazes add unpredictability and complexity, making it harder for the agent to make effective decisions, especially with limited simulations.

These findings suggest that while MCTS is effective in simpler environments, additional strategies, such as better heuristics or hybrid approaches, may be needed to handle more complex mazes efficiently.

Conclusions

This research has demonstrated the complexities involved in applying Monte Carlo Tree Search (MCTS) to Pac-Man, particularly when navigating mazes of varying sizes and levels of complexity. The results have highlighted several critical challenges and opportunities for refining the MCTS approach.

As the mazes grew in size and complexity, the MCTS agent faced increasingly difficult decision spaces. In smaller mazes, particularly those designed with optimized paths, the agent navigated efficiently, capitalizing on clearer routes to capsules and ghosts. However, the more complex environments—especially those with non-optimized layouts—proved far more challenging. The unpredictability and additional obstacles in these mazes pushed the MCTS algorithm beyond its capacity to manage decisions effectively with limited computational resources.

Another important observation from the experiments was the diminishing returns of increasing the number of simulations. While more simulations generally led to better decisions in simpler mazes, there was a clear point, especially in larger environments, where additional simulations no longer translated into better performance. In fact, the computational overhead started to reduce efficiency, underscoring the need for smarter resource management within the algorithm.

The structure of the mazes—whether optimized or non-optimized—played a decisive role in shaping the agent's performance. Optimized mazes allowed the MCTS agent to make more informed and effective decisions, taking advantage of less cluttered paths and more predictable patterns. In contrast, non-optimized mazes presented more difficult decisions, requiring the agent to constantly adapt to irregular and unpredictable structures. These findings

suggest that improvements in both algorithm efficiency and environment design could lead to more robust decision-making.

To better equip the MCTS agent for complex environments, several enhancements are necessary. First, integrating more advanced heuristics tailored to complex maze navigation could help the agent prioritize key decision points more effectively. Adaptive simulation techniques, where the number of simulations is dynamically adjusted based on the current state complexity, could also optimize computational effort. Finally, combining MCTS with reinforcement learning or neural networks may enable the agent to handle dynamic, adversarial environments more fluidly.

Further research could explore the application of these findings to larger, more intricate mazes or entirely new environments where the principles of MCTS may be useful. Additionally, the potential for hybrid decision-making models—integrating MCTS with other AI approaches—presents a promising avenue for developing more versatile and adaptive agents, not just for Pac-Man, but for real-world decision-making problems as well.

Література

17. Ou T., Lu Y., Wu X., Cao J. Monte Carlo Tree Search: A Survey of Theories and Applications / T. Ou, Y. Lu, X. Wu, J. Cao // 2022 3rd International Conference on Big Data, Artificial Intelligence and Internet of Things Engineering (ICBAIE). – 2022. – P. 388–396.

18. Jiang W. Application of Monte Carlo Tree Search algorithm in Go playing / W. Jiang // Applied and Computational Engineering. – 2024. – n. pag.

19. Steinmetz E. S., Gini M. L. More Trees or Larger Trees: Parallelizing Monte Carlo Tree Search / E. S. Steinmetz, M. L. Gini // IEEE Transactions on Games. – 2021. – Vol. 13. – P. 315–320.

20. Pepels T., Winands M. H. M., Lanctot M. Real-Time Monte Carlo Tree Search in Ms Pac-Man / T. Pepels, M. H. M. Winands, M. Lanctot // IEEE Transactions on Computational Intelligence and AI in Games. – 2014. – Vol. 6. – P. 245–257.

21. Nguyen K. Q., Thawonmas R. Monte Carlo Tree Search for Collaboration Control of Ghosts in Ms. Pac-Man / K. Q. Nguyen, R. Thawonmas // IEEE Transactions on Computational Intelligence and AI in Games. – 2013. – Vol. 5. – P. 57–68.

22. Liu X., Li Y., He S., Fu Y., Yang J., Ji D., Chen Y. To Create Intelligent Adaptive Game Opponent by Using Monte-Carlo for the Game of Pac-Man / X. Liu, Y. Li, S. He, Y. Fu, J. Yang, D. Ji, Y. Chen // 2009 Fifth International Conference on Natural Computation. – 2009. – Vol. 5. – P. 598–602.

23. Bartolini N., Coletta A., Maselli G. SIDE: Self Driving Drones Embrace Uncertainty / N. Bartolini, A. Coletta, G. Maselli // IEEE Transactions on Mobile Computing. – 2023. – Vol. 22. – P. 3650–3664.

24. Pairet É., Hernández J. D., Carreras M., Pétillot Y. R., Lahijanian M. Online Mapping and Motion Planning Under Uncertainty for Safe Navigation in Unknown Environments / É. Pairet, J. D. Hernández, M. Carreras, Y. R. Pétillot, M. Lahijanian // IEEE Transactions on Automation Science and Engineering. – 2020. – Vol. 19. – P. 3356–3378.

25. Sandino J., Maire F., Caccetta P., Sanderson C., Gonzalez F. Drone-Based Autonomous Motion Planning System for Outdoor Environments under Object Detection Uncertainty / J. Sandino, F. Maire, P. Caccetta, C. Sanderson, F. Gonzalez // Remote Sensing. – 2021. – Vol. 13. – P. 4481.

26. Thatavarthy A. S., Harithas S. S., Singh G., Singh A. K., Krishna K. M. UrbanFly: Uncertainty-Aware Planning for Navigation Amongst High-Rises with Monocular Visual-Inertial SLAM Maps / A. S. Thatavarthy, S. S. Harithas, G. Singh, A. K. Singh, K. M. Krishna // 2023 American Control Conference (ACC). – 2022. – P. 557–563.

27. Patle B. K., Chen S. L., Patel B., Kashyap S. K., Sanap S. Topological drone navigation in uncertain environment / B. K. Patle, S. L. Chen, B. Patel, S. K. Kashyap, S. Sanap // Industrial Robot. – 2021. – Vol. 48. – P. 423–441.

28. Silva V. D. S. D., Dos Santos M. G. S., Vieira M. F. N., Matos V. S., Queiroz Í. N., Lima R. T. Autonomous navigation strategy in quadruped robots for uneven terrain using 2D laser sensor / V. D. S. D. Silva, M. G. S. Dos Santos, M. F. N. Vieira, V. S. Matos, Í. N. Queiroz, R. T. Lima // 2023 Latin American Robotics Symposium (LARS), 2023 Brazilian Symposium on Robotics (SBR), and 2023 Workshop on Robotics in Education (WRE). – 2023. – P. 290–295.

29. Cheraghi A. R., Abdelgalil A., Graffi K. Universal 2-Dimensional Terrain Marking for Autonomous Robot Swarms / A. R. Cheraghi, A. Abdelgalil, K. Graffi // 2020 5th Asia-Pacific Conference on Intelligent Robot Systems (ACIRS). – 2020. – P. 24–32.

30. Andreas E., Dorismond J., Gamarra M. A. Adversary decision-making using Markov models / E. Andreas, J. Dorismond, M. A. Gamarra // Defense + Commercial Sensing. – 2023.

31. Deng X., Jiang W. D number theory based game-theoretic framework in adversarial decision making under a fuzzy environment / X. Deng, W. Jiang // International Journal of Approximate Reasoning. – 2019. – Vol. 106. – P. 194–213.

32. Cullen A. C., Alpcan T., Kalloniatis A. Adversarial Decisions on Complex Dynamical Systems using Game Theory / A. C. Cullen, T. Alpcan, A. Kalloniatis // ArXiv abs/2201.12355. – 2022. – n. pag.

33. Yang Q., Parasuraman R. A Game-Theoretic Utility Network for Cooperative Multi-Agent Decisions in Adversarial Environments / Q. Yang, R. Parasuraman // arXiv: Multiagent Systems. – 2020. – n. pag.

34. Talebi S., Simaan M. A., Qu Z. Decision-Making in Complex Dynamical Systems of Systems With One

Opposing Subsystem / S. Talebi, M. A. Simaan, Z. Qu // 2019 18th European Control Conference (ECC). – 2019. – P. 2789–2795.

References

1. Ou T., Lu Y., Wu X., Cao J. Monte Carlo Tree Search: A Survey of Theories and Applications / T. Ou, Y. Lu, X. Wu, J. Cao // 2022 3rd International Conference on Big Data, Artificial Intelligence and Internet of Things Engineering (ICBAIE). – 2022. – P. 388–396.

2. Jiang W. Application of Monte Carlo Tree Search algorithm in Go playing / W. Jiang // Applied and Computational Engineering. – 2024. – n. pag.

3. Steinmetz E. S., Gini M. L. More Trees or Larger Trees: Parallelizing Monte Carlo Tree Search / E. S. Steinmetz, M. L. Gini // IEEE Transactions on Games. – 2021. – Vol. 13. – P. 315–320.

4. Pepels T., Winands M. H. M., Lanctot M. Real-Time Monte Carlo Tree Search in Ms Pac-Man / T. Pepels, M. H. M. Winands, M. Lanctot // IEEE Transactions on Computational Intelligence and AI in Games. – 2014. – Vol. 6. – P. 245–257.

5. Nguyen K. Q., Thawonmas R. Monte Carlo Tree Search for Collaboration Control of Ghosts in Ms. Pac-Man / K. Q. Nguyen, R. Thawonmas // IEEE Transactions on Computational Intelligence and AI in Games. – 2013. – Vol. 5. – P. 57–68.

6. Liu X., Li Y., He S., Fu Y., Yang J., Ji D., Chen Y. To Create Intelligent Adaptive Game Opponent by Using Monte-Carlo for the Game of Pac-Man / X. Liu, Y. Li, S. He, Y. Fu, J. Yang, D. Ji, Y. Chen // 2009 Fifth International Conference on Natural Computation. – 2009. – Vol. 5. – P. 598–602.

7. Bartolini N., Coletta A., Maselli G. SIDE: Self Driving Drones Embrace Uncertainty / N. Bartolini, A. Coletta, G. Maselli // IEEE Transactions on Mobile Computing. – 2023. – Vol. 22. – P. 3650–3664.

8. Pairet É., Hernández J. D., Carreras M., Pétillot Y. R., Lahijanian M. Online Mapping and Motion Planning Under Uncertainty for Safe Navigation in Unknown Environments / É. Pairet, J. D. Hernández, M. Carreras, Y. R. Pétillot, M. Lahijanian // IEEE Transactions on Automation Science and Engineering. – 2020. – Vol. 19. – P. 3356–3378.

9. Sandino J., Maire F., Caccetta P., Sanderson C., Gonzalez F. Drone-Based Autonomous Motion Planning System for Outdoor Environments under Object Detection Uncertainty / J. Sandino, F. Maire, P. Caccetta, C. Sanderson, F. Gonzalez // Remote Sensing. – 2021. – Vol. 13. – P. 4481.

10. Thatavarthy A. S., Harithas S. S., Singh G., Singh A. K., Krishna K. M. UrbanFly: Uncertainty-Aware Planning for Navigation Amongst High-Rises with Monocular Visual-Inertial SLAM Maps / A. S. Thatavarthy, S. S. Harithas, G. Singh, A. K. Singh, K. M. Krishna // 2023 American Control Conference (ACC). – 2022. – P. 557–563.

11. Patle B. K., Chen S. L., Patel B., Kashyap S. K., Sanap S. Topological drone navigation in uncertain environment / B. K. Patle, S. L. Chen, B. Patel, S. K. Kashyap, S. Sanap // Industrial Robot. – 2021. – Vol. 48. – P. 423–441.

12. Silva V. D. S. D., Dos Santos M. G. S., Vieira M. F. N., Matos V. S., Queiroz Í. N., Lima R. T. Autonomous navigation strategy in quadruped robots for uneven terrain using 2D laser sensor / V. D. S. D. Silva, M. G. S. Dos Santos, M. F. N. Vieira, V. S. Matos, Í. N. Queiroz, R. T. Lima // 2023 Latin American Robotics Symposium (LARS), 2023 Brazilian Symposium on Robotics (SBR), and 2023 Workshop on Robotics in Education (WRE). – 2023. – P. 290–295.

13. Cheraghi A. R., Abdelgalil A., Graffi K. Universal 2-Dimensional Terrain Marking for Autonomous Robot Swarms / A. R. Cheraghi, A. Abdelgalil, K. Graffi // 2020 5th Asia-Pacific Conference on Intelligent Robot Systems (ACIRS). – 2020. – P. 24–32.

14. Andreas E., Dorismond J., Gamarra M. A. Adversary decision-making using Markov models / E. Andreas, J. Dorismond, M. A. Gamarra // Defense + Commercial Sensing. – 2023.

15. Deng X., Jiang W. D number theory based game-theoretic framework in adversarial decision making under a fuzzy environment / X. Deng, W. Jiang // International Journal of Approximate Reasoning. – 2019. – Vol. 106. – P. 194–213.

16. Cullen A. C., Alpcan T., Kalloniatis A. Adversarial Decisions on Complex Dynamical Systems using Game Theory / A. C. Cullen, T. Alpcan, A. Kalloniatis // ArXiv abs/2201.12355. – 2022. – n. pag.

17. Yang Q., Parasuraman R. A Game-Theoretic Utility Network for Cooperative Multi-Agent Decisions in Adversarial Environments / Q. Yang, R. Parasuraman // arXiv: Multiagent Systems. – 2020. – n. pag.

18. Talebi S., Simaan M. A., Qu Z. Decision-Making in Complex Dynamical Systems of Systems With One Opposing Subsystem / S. Talebi, M. A. Simaan, Z. Qu // 2019 18th European Control Conference (ECC). – 2019. – P. 2789–2795.