IJIRIS: International Journal of Innovative Research in Information Security E-ISSN: 2349-7017
Volume 11, Issue 09, November 2025 P-ISSN: 2349-7009
https://www.ijiris.com/archives
https://doi.org/10.26562/ijiris.2025.v1109.23

AI GRID GAMES

Dr.Sumathi P
Associate Professor, Dept. of Computer Science and Engineering
Sri Sairam College of Engineering, Bengaluru, India
sumithirulogu@gmail.com
https://orcid.org/0000-0002-6192-3488
Akash, Ananya R Melagiri, Hariom Yadav, Shivanand Kalashetti
Dept. of Computer Science and Engineering
Sri Sairam College of Engineering, Bengaluru, India
sce23cs128@sairamtap.edu.in, sce23cs062@sairamtap.edu.in
sce2cs056@sairamtap.edu.in,sce2cs076@sairamtap.edu.in

Publication History
Manuscript Reference No: IJIRIS/RS/Vol.11/Issue09/NVIS10102
Research Article Open Access| Double-Blind Peer-Reviewed| Article ID: IJIRIS/RS/Vol.11/Issue09/NVIS10102 Received: 28, October 2025, Revised: 05, November 2025, Accepted: 12, November 2025, Published Online: 21, November 2025.
https://www.ijiris.com/volumes/Vol11/iss-09/23.NVIS10102.pdf
Citation: Dr.Sumathi,Akash,Ananya,Hariom,Shivanand(2025),AI Grid Games,IJIRIS: International Journal of Innovative Research in Information Security,Volume 11, Issue 09 of 2025 pages 595-599
Doi:> https://doi.org/10.26562/ijiris.2025.v1109.23
BibTeX Key:Dr.Sumathi@2025AI
IJIRIS papers should be cited as IJIRIS (International Journal of Innovative Research in Information Security,AM Publications, India 2025, ISSN 2349-7017, https://doi.org/10.26562/ijiris.2025.v1109.23 The journal's official abbreviation is IJIRIS. Orcid: https://orcid.org/0009-0004-9398-7488

Abstract: This study introduces a comprehensive intelligent gaming framework that unites mathematical reasoning, game theory, advanced data structures and algorithms (DSA), and artificial intelligence/machine learning (AI/ML) to construct and evaluate grid-based games. The objective is to demonstrate how algorithmic principles and adaptive learning mechanisms combine to create interactive, explainable, and scalable systems that can be used for education and research. Focusing on exemplar applications AI Tic Tac Toe, Word Search, Sudoku Solver, Chess variants, and intelligent pathfinding the framework examines both classical deterministic algorithms and modern learning-driven approaches. The literature overview traces the progression from foundational techniques, such as Knuth's Algorithm X and trie-based lookup, to probabilistic and reinforcement learning strategies including Monte Carlo Tree Search (MCTS) and neural-network-based self-play. In proposing a modular architecture that integrates front-end visualization with cloud-enabled backends, this work seeks to bridge the gap between theoretical constructs and practical deployments. Experimental observations indicate that combining principled algorithmic reasoning with adaptive AI improves solution quality, user engagement, and the educational value of such platforms. The resulting platform is presented as a research-grade, extensible toolkit that supports experimentation in multi-agent learning, explainable decision-making, and pedagogical demonstrations.
Keywords: AI grid games, artificial intelligence, game theory, minimax, MCTS, reinforcement learning, Sudoku solver, trie, path finding, system architecture, educational platform, multi-agent learning, algorithmic reasoning, explainable AI, technology stack.

1. INTRODUCTION

AI Grid Games are computational environments characterized by discrete, spatially organized cells where agents make decisions based on local and global state information. These environments are especially useful for studying planning, reasoning, and learning because they combine formal rules with a visually interpretable layout. In this work, grid games are treated as microcosms of broader AI challenges: they require the integration of search algorithms, efficient data structures, constraint management, and adaptive learning strategies. The constrained nature of grid worlds makes them amenable to rigorous analysis while retaining sufficient complexity to model real-world decision problems. Classic instances of grid games include Tic Tac Toe, Sudoku, word search puzzles, and chess-like variants. Each of these games introduces different algorithmic demands: Tic Tac Toe emphasizes adversarial search and game-theoretic balance, Sudoku focuses on constraint satisfaction and propagation, word searches rely on rapid string lookup and pattern matching, and chess variants require deep strategic planning often supported by heuristic evaluation. Because these domains share a common grid structure, it is possible to design a unified framework that reuses core algorithmic modules while tailoring heuristics and representations to the specific game rules. The pedagogical value of grid games is significant. Visual, interactive platforms allow students and researchers to observe how algorithmic choices affect outcomes in real time. For example, visualizing search trees or move evaluations can clarify why certain heuristics lead to better play. Moreover, grid games can be deployed in classroom settings to teach concepts ranging from discrete mathematics to probabilistic decision-making.

IJIRIS: International Journal of Innovative Research in Information Security E-ISSN: 2349-7017
Volume 11, Issue 09, November 2025 P-ISSN: 2349-7009
https://www.ijiris.com/archives
https://doi.org/10.26562/ijiris.2025.v1109.23

From an application standpoint, the techniques developed for grid games translate to robotics, logistics, and simulation: pathfinding heuristics inform robot navigation, constraint solvers aid scheduling, and learning-based policies support adaptive control in uncertain environments.

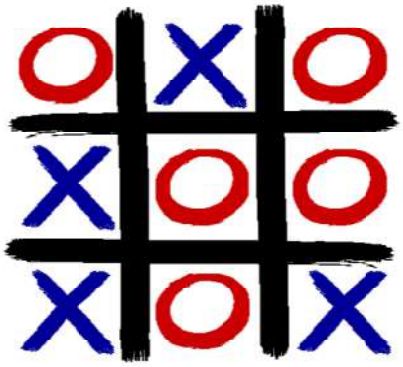

Fig1.1(a):Tic-Tac-Toe game

5	3			7				
6			1	9	5			
	9	8					6	
8				6				3
4			8		3			1
7				2				6
	6					2	8	
			4	1	9			5
				8			7	9

Fig1.2(b): Sudoku game

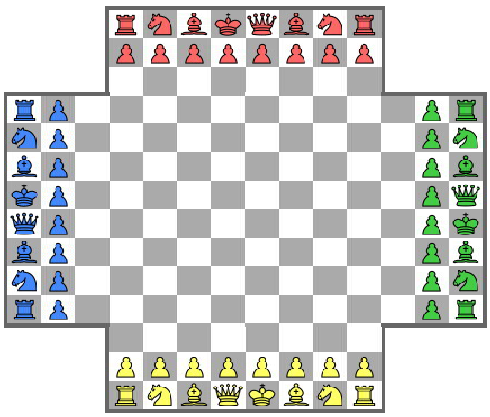

Fig.3(c): Four-player chess variants

Fig1.4(d): Word Search puzzle

2. LITERATURE SURVEY

Research into AI grid games rests on several pillars: efficient constraint solving, adversarial search, probabilistic planning, and learning. Over the last several decades, these areas have produced a wealth of techniques that have been adapted to grid-structured problems.The following subsections summarize the most influential contributions relevant to this work.

2.1 Knuth – Algorithm X / Dancing Links (DLX)

Donald Knuth's Algorithm X, when implemented with the Dancing Links technique, provides an elegant and memory-efficient method for solving exact cover and constraint satisfaction problems. By representing constraints as linked structures and allowing rapid removal and reinsertion of candidates, DLX enables exhaustive search to be performed with practical performance on problems like Sudoku. In the context of grid games, DLX can be used to express complex constraint relationships succinctly and provide a near-optimal traversal order for backtracking.

2.2 Russell & Norvig – Artificial Intelligence: A Modern Approach

The textbook by Russell and Norvig synthesizes core search algorithms, adversarial techniques, and heuristic design principles. Minimax and alpha-beta pruning remain canonical algorithms for adversarial play; heuristic search techniques such as A* provide principled ways to combine exact search with admissible heuristics for pathfinding and planning. These methods are foundational for constructing interpretable AI agents within grid environments.

2.3 Fredkin / Trie-based Research

Tries, as originally developed and popularized in information retrieval literature, structure strings by shared prefixes, yielding time complexities that are effectively proportional to the length of the search term. This makes them exceptionally useful for word-matching and autocomplete tasks in grid-based word games, enabling constant-time checks per character and efficient enumeration of valid dictionary entries that begin with a given prefix.

2.4 Browne et al. (2012) – MCTS Methods Surveyed

Monte Carlo Tree Search (MCTS) leverages randomized simulation to evaluate the long-term value of moves by balancing exploration and exploitation. Browne et al.'s survey captured the algorithm's flexibility across domains where the state space is large and deterministic evaluation is infeasible. In grid games with high branching factors, MCTS allows agents to derive statistically sound estimates of move quality without exhaustive enumeration.

2.5 Silver et al. (2017) – AlphaZero

AlphaZero demonstrated that self-play combined with neural-network-guided search could produce superhuman results across several games. By training value and policy networks via reinforcement learning and integrating them into a search procedure, AlphaZero-style agents offer a template for how learning-based methods can be merged with search to yield adaptive and continually improving agents. Taken together, these works illustrate a progression from classical, algorithmic problem-solving techniques toward hybrid and learning-centered paradigms, which inform the design choices in the unified framework proposed in this paper.

3. EXISTING SYSTEM

Current AI grid-game implementations tend to emphasize well-known algorithmic baselines and optimized engines. In practice, Tic Tac Toe implementations commonly employ Minimax because the game's limited depth allows exhaustive evaluation. For many word-search applications, trie or prefix tree data structures are the practical choice given their speed for dictionary lookup. However, these implementations typically optimize for runtime and determinism rather than adaptability; they are designed to perform predictably under fixed rules and datasets. Similarly, Sudoku solvers often rely on backtracking and constraint propagation techniques that guarantee correctness but can struggle on extremely large or adversarial puzzle instances. Chess engines traditionally combine hand-crafted evaluation functions with deep search and domain-specific heuristics, yet many such engines function as black boxes with limited explainability about why a particular move was chosen. Pathfinding modules usually implement A* or Dijkstra's algorithm, which excel at shortest-path computation but do not incorporate higher-level strategic reasoning or adaptive difficulty balancing. These limitations highlight an opportunity: existing systems are robust on classical metrics but do not typically provide adaptive, explainable behavior or pedagogical traces that are useful in research and education. This observation motivates a platform that integrates core algorithmic reliability with modern learning and interpretability features.

4. PROPOSED WORK

This project proposes a unified platform that synthesizes classical algorithms and contemporary AI strategies to form an extensible research and educational toolkit. The design prioritizes modularity: core services such as state representation, move generation, and evaluation are shared across games, while game-specific modules implement rules and heuristics. This structure enables rapid prototyping and comparative evaluation across algorithmic regimes. For concrete examples, the Tic Tac Toe module implements Minimax augmented with alpha-beta pruning and move-ordering heuristics to reduce search tree size. The Word Search module employs trie-based dictionaries combined with scoring heuristics and optional learned priors that bias search towards commonly used words. The Sudoku solver merges algorithmic backtracking with constraint propagation and optional probabilistic ranking of candidate placements to handle difficult instances more effectively. The Chess module integrates Monte Carlo Tree Search and policy/value approximators in an architecture inspired by modern self-play systems; however, emphasis is placed on interpretability by exposing intermediate evaluations and move rationales to the user. The pathfinding engine uses A* as a baseline while allowing additional game-theoretic heuristics to influence choices when multiple objectives (e.g., risk avoidance, resource collection) are present. Across modules, learning components may be enabled to capture player behavior or adapt difficulty in real time. The platform also supports experiment logging and model versioning so that researchers can reproduce training regimes and compare algorithmic variants across standard benchmarks.

5. SYSTEM ARCHITECTURE

The system architecture follows a layered, modular approach that separates presentation, logic, persistence, and model management. At the top is a user-facing front-end that renders game boards, provides controls, and visualizes AI decisions and internal state. The front-end communicates via RESTful APIs to a central game engine responsible for rule enforcement, move generation, and coordination of AI modules. Within the engine, a scheduler coordinates short-term lookahead search, long-term model-based planning, and asynchronous learning updates. Data structures such as graphs, tries, and specialized constraint tables support efficient state transitions.

IJIRIS: International Journal of Innovative Research in Information Security E-ISSN: 2349-7017
Volume 11, Issue 09, November 2025 P-ISSN: 2349-7009
https://www.ijiris.com/archives
https://doi.org/10.26562/ijiris.2025.v1109.23

The architecture also includes a dedicated analytics pipeline that aggregates gameplay telemetry for evaluation and model improvement. To support research workflows, the architecture includes facilities for model check pointing, experiment comparison, and reproducible configuration management. This design ensures that algorithmic improvements can be evaluated consistently while preserving historical training artifacts for later analysis.

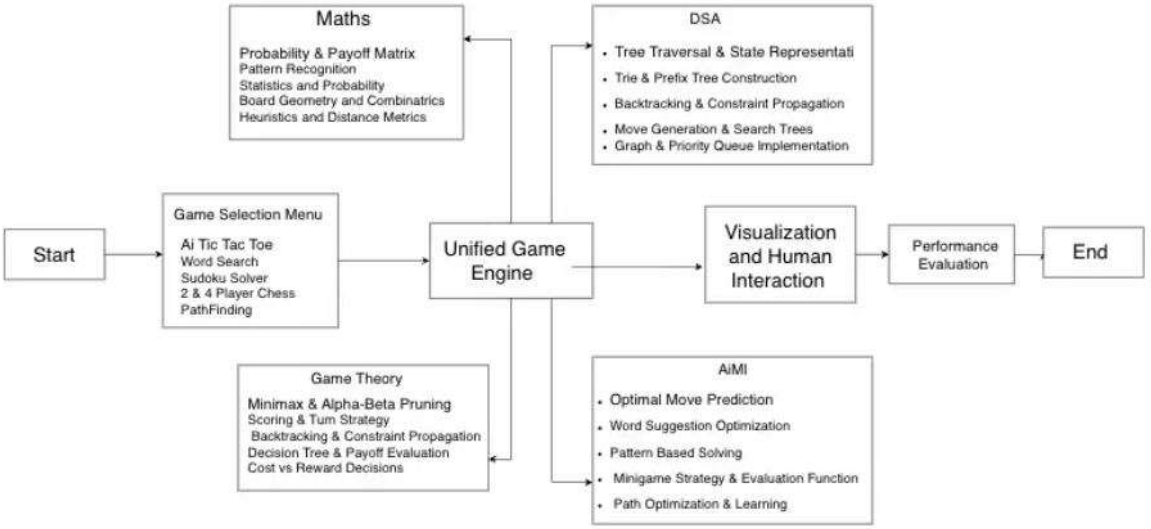

Figure 5.1 (a) System architecture of the grid games

6. TECHNOLOGY STACK

A modern AI game platform requires a carefully chosen technology stack that balances developer productivity, runtime performance, and scalability. The following components were selected to meet these goals while supporting rapid experimentation. The front-end is implemented using HTML5 and React to enable component-driven UI development. Tailwind CSS is used as a utility-first styling framework, allowing quick iteration on visual components without heavy CSS maintenance. JavaScript powers client-side interactivity and connects visualization elements to backend telemetry for real-time feedback. The backend core is implemented in Java with Spring Boot. Java provides robust concurrency support and a mature ecosystem for backend services, while Spring Boot simplifies the creation of modular RESTful services. AI model serving components and training orchestration can be implemented via microservices that scale independently. For persistence, the system uses SQLite for lightweight local storage of session-oriented data and MongoDB for flexible storage of experimental records and model artifacts. This hybrid approach offers both the simplicity of file-backed databases and the schema flexibility required for evolving ML logs. Deployment is handled through AWS, which provides elastic compute and managed services for storage and model hosting. To ensure reproducible environments, Docker containers are used across development and deployment pipelines. GitHub acts as the source control system, while MLflow or similar tools manage model experiment tracking.

7. RESULTS AND OUTCOME

The implementation demonstrated the feasibility of integrating classical algorithms with learning-augmented methods in a unified environment. Across multiple modules, the system produced measurable improvements in responsiveness, interpretability, and adaptability compared to baseline, non-adaptive implementations. In Tic Tac Toe, Minimax with alpha-beta pruning and move ordering reduced average decision time while maintaining optimal play. This resulted in near-instantaneous move selection for human-scale interactions, making the game suitable for teaching adversarial search without performance lag. For Word Search, trie-backed lookup with incremental scoring allowed rapid identification of valid words even from large dictionaries. When augmented with a lightweight statistical model that prioritized likely dictionary entries, the system demonstrated improved suggestion accuracy in real gameplay sessions. The Sudoku component combined DLX-style constraint handling with heuristic candidate ordering. This blend reduced backtracking depth on challenging puzzles and improved solver runtime in practice compared to naive backtracking implementations. The Chess module's combination of Monte Carlo Tree Search and policy/value approximators produced agents that evolved with continued self-play. By exposing intermediate evaluations and principal variation lines to the user interface, the engine offered greater transparency in how choices were formed. Pathfinding results showed that augmenting A* with secondary heuristic layers (for risk assessment and multi-objective optimization) led to routes that better balanced safety and efficiency in simulated environments with dynamic hazards. Collectively, these results indicate that a hybrid approach combining deterministic algorithmic guarantees with adaptive learning and explainability features yields a platform that is both practical for gameplay and valuable for education and research. Detailed benchmark numbers (e.g., average solve times, nodes expanded, learning curves) are captured in the associated experiment logs.

OUTCOMES

Beyond raw performance metrics, the platform's outcomes include pedagogical artifacts such as visualizations of search trees, step-by-step constraint propagation traces, and interactive dashboards for model performance. These tools made the underlying algorithms more accessible to students and facilitated reproducible experimental workflows for researchers. The modular design also allowed researchers to replace or tune components (e.g., swap heuristic evaluators or enable different learning regimes) without affecting the broader system.

8. CONCLUSION

This work presents a unified AI Grid Game framework that demonstrates how classical algorithms and modern learning techniques can be combined to produce educationally valuable and research-ready systems. By systematically integrating constrained search methods, efficient data structures, and adaptive learning modules, the platform provides a flexible environment for experimenting with multi-agent interactions, explainable decision-making, and algorithmic pedagogy. The key contribution is not only the functional implementation across several canonical game types but also the emphasis on interpretability and reproducibility. By exposing internal reasoning and integrating robust experiment management, the framework supports both classroom instruction and rigorous scientific inquiry. Future work will include larger-scale evaluations, integration with robotics platforms for embodied agent testing, and exploration of generative content methods to produce novel puzzle instances for training and evaluation.

REFERENCES

1. Overview of grid-based reinforcement learning and AI environments in academic research.
2. Applications of game-based learning and algorithmic strategies in artificial intelligence.
3. Figure 1.1 (a) Tic-Tac-Toe game https://www.c-sharpcorner.com/UploadFile/75a48f/tic-tac-toe-game-in-C-Sharp/
4. Figure 1.2(b) Sudoku game https://en.wikipedia.org/wiki/Sudoku
5. Figure 1.3(c) Four-player chess variants game https://greenchess.net/variants.php?cat=9
6. Figure 1.4(d) Word search puzzle https://thewordsearch.com/
7. Hu, C., Zhao, Y., Wang, Z., Du, H., Liu, J. (2024). Games for Artificial Intelligence Research: A Review and Perspectives. IEEE Transactions on Artificial Intelligence.
8. Han, L., Sun, P., et al. (2019). Grid-Wise Control for Multi-Agent Reinforcement Learning in Video Game AI. Proceedings of Machine Learning Research.
9. Bamford, C. (2021). Griddly: A Platform for AI Research in Games. ScienceDirect.
10. Russell, S. & Norvig, P. (2021). Artificial Intelligence: A Modern Approach (4th ed.).
11. Browne, C. et al. (2012). A Survey of Monte Carlo Tree Search Methods.
12. Silver, D. et al. (2017). AlphaZero: Deep Reinforcement Learning (landmark studies cited in above reviews).
13. Griddly: A platform for AI research in games (Bamford et al., 2021), Games for Artificial Intelligence Research: A Review and Perspectives (Hu et al., 2024), Artificial Intelligence for Games (Millington & Laird, 2nd ed.), Constructing a Game Engine: A proposed curriculum (Del Gallego, 2024).
14. Minimax and Tic Tac Toe (Datacamp, GeeksforGeeks), Trie for Word Search (GitHub, Joshi), Backtracking for Sudoku (NeverStopBuilding), Chess Engine limitations (GitHub, Cledersonbc), A* Pathfinding limitations (Grid Path Finding, AEUSO).
15. GeeksForGeeks, 'Artificial Intelligence (AI) Algorithms', 2024
16. modl.ai, 'AI Game Development: The Ultimate Guide', 2024
17. AnshadAmeenza, 'Game Development with AI: A Comprehensive Guide', 2025
18. Helika, 'AI Game Algorithm (What You Need To Know)', 2025
19. Citrusbug, 'Top 10 AI Algorithms for Beginners', 2025
20. Ian Millington, 'Artificial Intelligence for Games'